C, C++

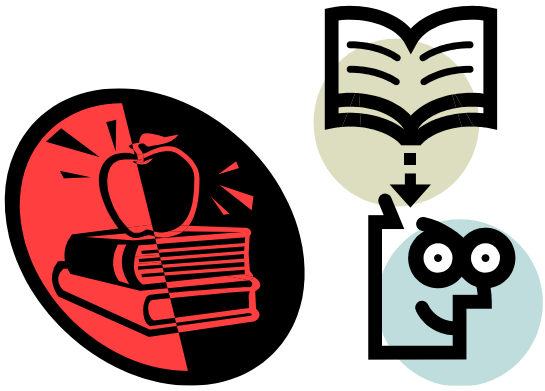
JAVA, PYTHON, PHP, JAVASCRIPT AND LINUX FOR BEGINNERS

BY

MANJUNATH.R

C, C++, Java, Python, PHP, JavaScript and Linux For Beginners

(A Step-by-Step Guide to Coding)



"The only true wisdom is in knowing you know nothing."

− Socrates

Manjunath.R

#16/1, 8th Main Road, Shivanagar, Rajajinagar, Bangalore560010, Karnataka, India

***Email:** manjunath5496@gmail.com

Disclaimer

Despite my best efforts to assure the accuracy of the material in this book, I do not accept and hereby disclaim any liability to any party for any loss, damage, or disruption caused by mistakes or omissions, whether caused by negligence, accident, or any other cause.

For any suggestions or concerns, please write to me: **manjunath5496@gmail.com**

Dedication

I **dedicate this book** to every individual, programmer, teacher, educational institutions and enterprise corporations in every country of the world for their immense contributions towards the process of creating, designing, deploying and supporting software…

Acknowledgements

Without the amazing work of some renowned programmers, their creativity, and their inventiveness in the field of software programming, this book would not have been accomplished. I would like to use this opportunity to thank my dearest friend and well-wisher "**Lawrence**" for his unwavering support during the **COVID crisis** and for giving me access to all the resources I needed to finish this book. I want to express my gratitude to my family for their support and encouragement as I wrote this book, especially to my **mother**, who has been a tremendous source of inspiration in my life. I owe a lot of gratitude to my mother for teaching me how to be perseverant and strong in life. Finally, I want to emphasize how crucial patience is when writing a book or taking on any other project in life.

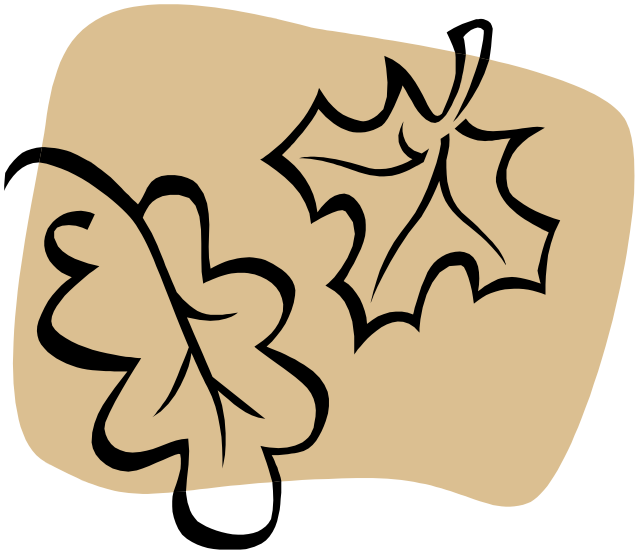
Foreword

I'm neither the proprietor of a well-known publishing house or a top IT firm with hundreds of in-house programmers who could easily produce anything I needed. I am a self-employed software engineer who is passionate about what I do, and believe me when I say that a lot of work and effort went into compiling this **comprehensive edition**. I'll be overjoyed if it helps even a few others reach their ideal positions in their professions.

Thank You

− Manjunath.R

An Enjoyable Introduction to Coding



Introduction

Today's devices are mostly powered by software: almost everyone uses Facebook, WhatsApp and Twitter to communicate, many phones have internet-connected desktops, and the majority of office work requires using a computer to do tasks. As a response, there is a huge increase in demand for programmers. **Numerous books**, interactive websites, and programmer training courses make the bold claim that they can turn ambitious novices into software engineers earning six figures. This book is for all programmers, whether you are a novice or an experienced pro. Its numerous examples and **well paced discussions** will be especially beneficial for beginners. Those who are already experienced with programming will probably gain more from this book, of course. You will be at a modest level of programming proficiency when you have finished this book, from where you can take yourself to next levels so that you can automate simple tasks such as:

- Making a file backup
- Get rid of the irritating emails
- Completing online forms

This book will make an amazing complement to any tutorial and serve as a source of information to your specific inquiries if you are just learning what kind of animals C, C+ +, Java, PHP, Python, and JavaScript are. Even if your career has nothing to do with computers, the skills you learn from programming can be valuable at school and at work. **Programming** is a pleasant, occasionally difficult and perhaps frustrating activity. Creativity, logic, and problem-solving are all enhanced through programming.

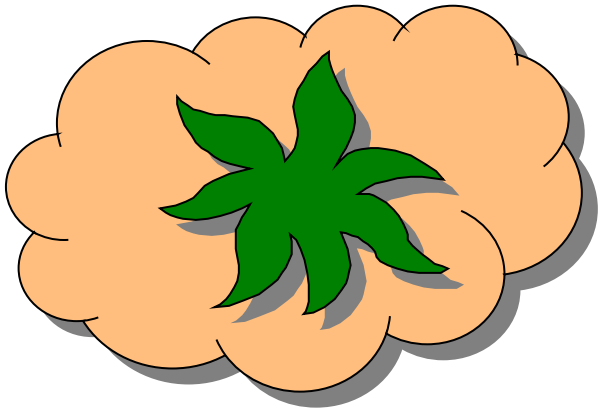
- Educational institutions are teaching it
- Corporate societies are employing it
- Pupils need it
- (Pedagogues desire it... ;)
- (Coders perceive it... :)

Have Fun!

As you progress through this book, keep in mind that programming can be enjoyable. Do not consider of this as work. Consider programming as a means to develop entertaining games or software applications that you can show off to others or your friends. Programming is a tremendous brain workout and is essential today because so much of our everyday world is automated. But above all, you have access to the quick-paced, creative world that depends on machine connections.

"The only way to learn a new programming language is by writing programs in it."

– Dennis Ritchie



Note:

- **Linux version used:** CentOS Linux release 7.3.1611 (Core)
- **Python version used:** 3.7.3

The Basic Programming Principles That Every Programmer Should Know:

1. Always be aware of the purpose of your software program before beginning to write it.
2. Programming is not the solution; it is merely a means to achieve a solution.
3. Consider the problem rather than just the solution.
4. Always try to make things simpler; anyone can come up with a complicated answer to a problem. To make a solution simple while remaining consistent, it requires extra work and consideration.
5. Reduce Deeply Nested Ifs or Loops: When your software program is deeply nested, your program becomes complicated and disorganized.
6. Delete Unnecessary Code. Make sure your software program is safe, secure, reliable, testable and clear to read.
7. Give code reviews some attention so you can spot bugs early, before they cause serious problems in your software application.
8. Reduce complexity. Software programs must have clear explanations.
9. Generalize your software program. Make sure your software program is documented. Understanding the function of a certain component of the software application is greatly aided by the documentation and comments.

10. Fancy algorithms and data structures are more difficult to implement. Use simple, efficient, appropriate algorithms and data structures.
11. Refractor your software program frequently to improve its internal software attributes in terms of upkeep, testing and comprehension.
12. Each time you make a change to your software program: check it, build it and test it.
13. Before being released, all software codes must pass each and every unit test.
14. Always use caution when using someone else's code. Maintain a standardized, orderly and generally consistent coding style.
15. Avoid implementing a code style that is too hard to understand.
16. Because it makes the code more difficult to maintain, duplication is seen adversely in software programming.
17. Look for bugs and flaws and fix them. Divide your software program into Brief, Concise Units.
18. Avoid overdesigning. Focus your software design on the requirements of the clients.
19. Program defensively. Functions should be simple and do a distinct, defined task.
20. Create reusable functions and Keep the functions as simple, immutable and manageable as possible.
21. When naming your variables and functions, choose names that are meaningful and descriptive.

22. Put your software program's structure on view by using indentation.
23. Delete any unused variables and functions; do not comment them.
24. If you feel that a part of the software program is excessively unorganized, regroup and modify it, or even split it up into different portions.
25. Avoid using GOTO statements because they cause the software program to be unstructured, which makes it harder to understand and makes debugging more complex.
26. Avoid using the same identifier more than once.
27. The length of functions shouldn't be excessively long.
28. Think Twice, Code Once: Encourage yourself to consider the problem more before coming up with a solution.
29. The very first step in making a software program readable by humans is to add comments. Comments should be detailed explanations of a software program.
30. White space should be utilized regularly to increase code readability even though it has little significance to compilers.
31. Coding standards must be followed while formatting code.
32. Avoid security pitfalls and Keep your software code portable.
33. All software design is redesign. Take advice from others' experience.
34. The writing of software program should make it simple for a future software developer to correct errors or modify its functionality.
35. Never compromise clarity for a false sense of efficiency.

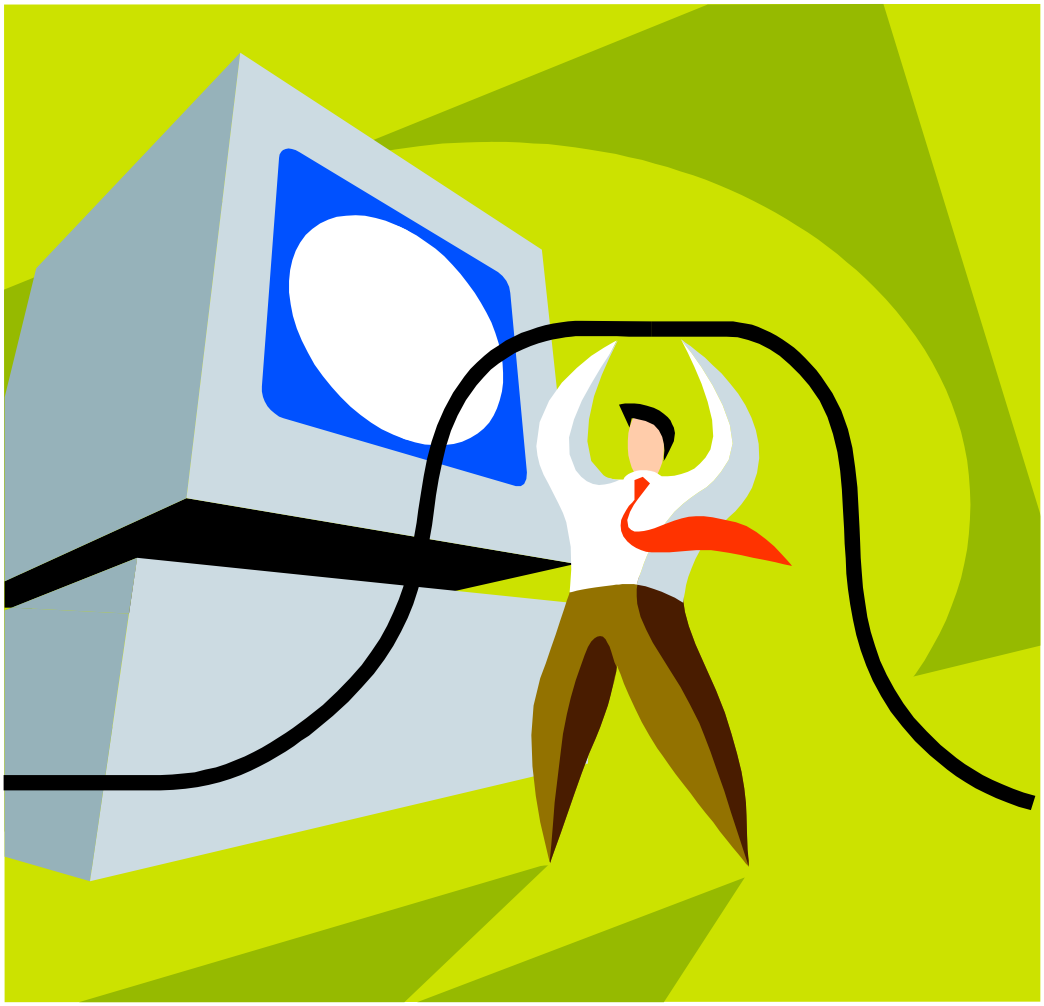
> "More computing sins are committed in the name of efficiency (without necessarily achieving it) than for any other single reason − including blind stupidity."
>
> **− W.A. Wulf**

36. Enhance the appearance of software program by avoiding excessively long names or ambiguous acronyms
37. Look for a method that employs a loop rather than duplicating lines. Compared to 100 individual blocks of code, one loop that can handle 100 repetitions is simpler to debug.

You're not coding to amaze strangers. You're in this profession to find ways to resolve problems.

"The computer programmer is a creator of universes for which he alone is the lawgiver. No playwright, no stage director, no emperor, however powerful, has ever exercised such absolute authority to arrange a stage or field of battle and to command such unswervingly dutiful actors or troops."

– Joseph Weizenbaum



Top 10 Programming Languages and Their Applications

Python	Artificial Intelligence, Deep learning and Machine Learning
JavaScript	Rich Interactive Web Development
Java	Enterprise Application Development
R	Data Analysis
C/C++	Operating Systems and System Tools
Golang	Server-Side Programming
C#	Application and Web Development Using .NET
PHP	Web Development
SQL	Database Management
Swift	For Mobile Application Development on iOS

How much time does it take to become a good programmer?

A skilled coder can identify the best solution to any problem and solve even the most challenging issues. Being a good programmer requires constant knowledge upkeep and the acquisition of new skills. A PhD isn't always necessary to become a skilled programmer, but discipline and determination are. Being a successful programmer demands you to be one step ahead, while becoming a respectable coder takes years of hard effort.

Image Credit: Wikipedia.org

Contents

When you innovate, you've got to be prepared for everyone telling you you're nuts.

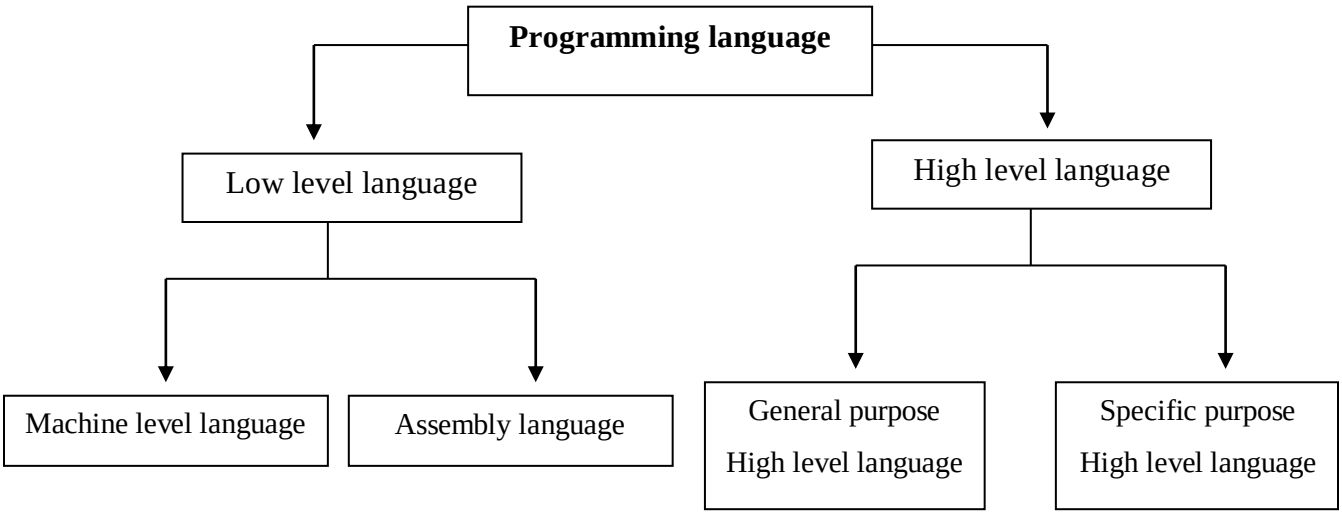
- Larry Ellison

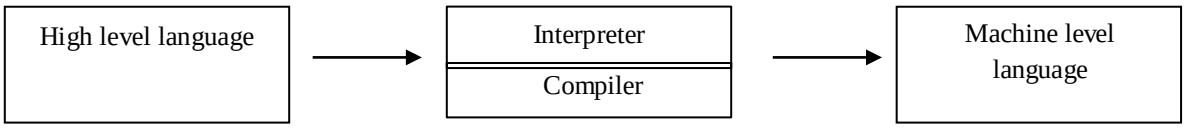
C++ Exercises	1331
Java Exercises	1385
Python Exercises	1461

Whether you want to uncover the secrets of the universe, or you just want to pursue a career in the 21st century, basic computer programming is an essential skill to learn.

– Stephen Hawking

(One of the most brilliant theoretical physicists in human history)





C Language Basic Syntax Rules

```
#include<stdio.h>
int main()
{
// Hello World Program
printf("Hello,world!");

return 0; /* indicate that program ended successfully */
}
```

Including header files

main() function must be there

Single line comment

Semicolon after each statement

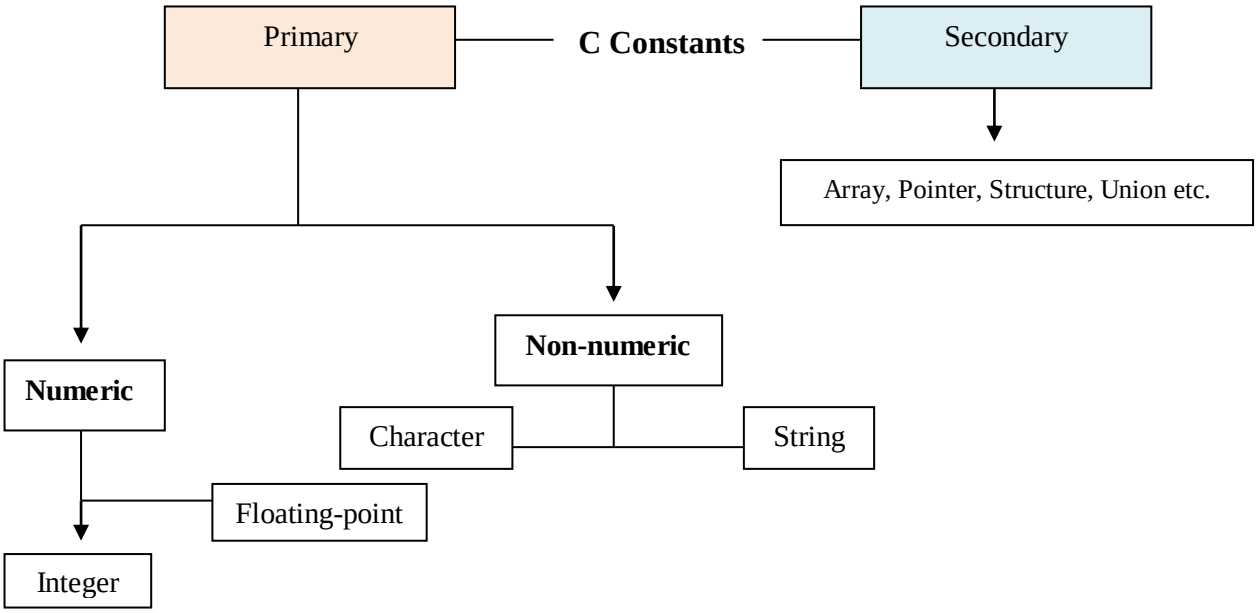
Program enclosed by curly braces

C Keywords

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
continue	for	signed	void
do	if	static	while
default	goto	sizeof	volatile
const	float	short	unsigned

Special Characters in C Programming

,	<	>	.	_
(	)	;	$	:
%	[	]	#	?
'	&	{	}	"
^	!	*	/	|
-	\	~	+	



- **Integer : 246, 0, -3679, +35, 9777, -36026 etc.**
- **Floating-point : 246.23, 0000.23, -36.79, +35.56, 9.777, -360.216 etc.**
- **Character : 'b', '?', '@', '#' etc.**
- **String : "C", "Java", "Python", "JavaScript" etc.**
- **Boolean : True, False**
- **Nothing : Null**

Data types:

Types	Size in bytes	Keyword
Integer	2	int
Floating-point	4	float
Double	8	double
Character	1	char

Escape Sequences:

Sequence	Meaning
\a	System alarm (bell)
\b	Backspace
\f	Form feed
\n	Newline
\r	Carriage Return
\t	Horizontal Tab
\v	Vertical Tab
\\	Backslash
\'	Single quote
\"	Double quote
\?	Question mark
\0	End of string

Input output functions:

	Input	Output
Formatted	scanf()	printf()
Unformatted	getchar() gets()	putchar() puts()

```c
#include <stdio.h> // This is a preprocessor directive
int main() // This identifies the function main()
{ // This marks the beginning of main()
printf("Albert!!\n\a"); // This line outputs a quotation
return 0; // This returns control to the operating system
} // This marks the end of main()

# Output: Albert!!
```

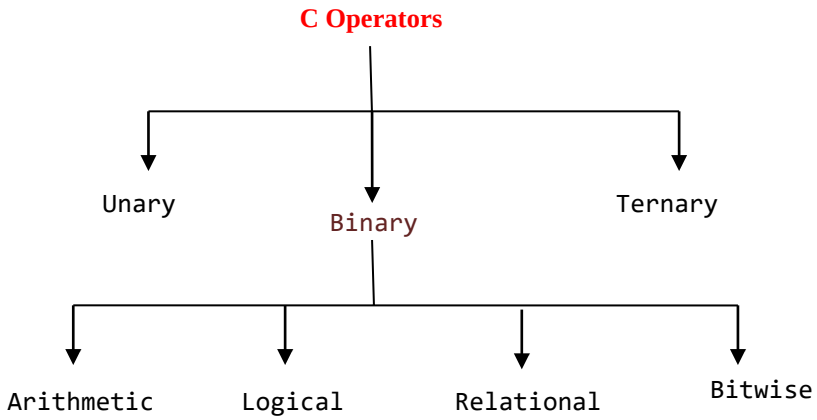
```c
// C Program to Find Square of a Number

#include<stdio.h> // preprocessor directive
/* function main begins program execution */
int main() {
int x; /* number to be input by user */
int square; /* variable in which square will be stored */
printf("Enter any number: "); /* prompt */
scanf("%d", &x); /* read an number */
square = x * x; /* assign product to square */
printf("The square of the number = %d", square); /* print square */
return 0; /* indicate that program ended successfully */
} /* end function main */
```

Output:

```
Enter any number: 2 # entered number
The square of the number = 4
```

Format specifiers in C

Format Specifier	Meaning
%c	Read a Single Character
%d	Read a Decimal integer
%e	Read a Floating-point number
%f	Read a Floating-point number
%g	Read a Floating-point number
%h	Read a short integer
%i	Read a Decimal or hexadecimal or octal number
%o	Read an octal number
%p	Read a pointer
%s	Read a string
%u	Read an Unsigned integer
%x	Read a hexadecimal number
%n	Prints nothing
%%	Prints % character



- **Arithmetic operators**

Operations	Operator
Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus	%

- **Relational operators**

Operator	Meaning
<	Lesser than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
!=	Not equal to

- **Logical operators**

Operator	Meaning
&&	Logical AND
||	Logical OR
!	Logical NOT

- **Bitwise operators**

Operator	Meaning
&	Bitwise AND
|	Bitwise OR
^	Exclusive –OR (XOR)
~	1's complement
<<	Left shifting of bits
>>	Right shifting of bits

Shorthand operators

Operator	Example	Equivalent construct
+	a=a+b	a+=b
-	a=a-b	a-=b
*	a=a*b	a*=b
/	a=a/b	a/=b
%	a=a%b	a%=b
&	a=a&b	a&=b
|	a=a|b	a|=b
^	a=a^b	a^=b
<<	a=a<<b	a<<=b
>>	a=a>>b	a>>=b

Mathematical Functions

Function	Description
sqrt(x)	Return the square root of x
exp(x)	Return exponential (e^x)
log(x)	Return natural logarithm of x (base e)
log10(x)	Return logarithm of x (base 10)
fabs(x)	Return absolute value of x
abs(x)	Return absolute value of x
floor(x)	Return a value rounded to the next lower integer
pow(x,y)	Return x raised to power y (x^y)
fmod(x,y)	Return floating-point remainder of x/y (with same sign of x)
sin(x)	Return the sine of x
cos(x)	Return the cosine of x
tan(x)	Return the tangent of x
acos(x)	Return arc cosine of x [$\cos^{-1}(x)$]
asin(x)	Return arc sine of x [$\sin^{-1}(x)$]
atan(x)	Return arc tangent of x [$\tan^{-1}(x)$]

```c
#include<stdio.h>
#include <math.h>
int main() {
  // ceil() function rounds a number upwards to its nearest integer
  printf("%f\n", ceil(2.6));
  // Output: 3.000000

  // floor() method rounds a number downwards to its nearest integer
  printf("%f\n", floor(2.6));
  // Output: 2.000000
  return 0;
}
```

```c
/*
 * Multi-line
 * Comment.
*/

/*****************************************
 * Multi-line *
 * Comment. *
 *****************************************/
```

Multi-line comments

```c
#include<stdio.h>
int main(void) {
 printf("\"Albert Einstein was a German-born theoretical \"\nphysicist\n");
 return 0;
}
```

Output:

```
"Albert Einstein was a German-born theoretical "
physicist
```

C Header files

Header file	Description
stdio.h	Input and Output functions
conio.h	Console Input and Output functions
stdlib.h	Some standard library functions
math.h	Mathematical functions
string.h	String manipulation functions
ctype.h	Character handling functions
time.h	Time computing functions
malloc.h	Memory allocation and deallocation functions
graphics.h	Graphical functions
dos.h	Function linking DOS routines
wctype.h	Functions to classify and transform individual wide characters
limits.h	Functions define various symbolic names
float.h	Functions define set of various platform-dependent constants related to floating point values

Preprocessor Directives

Preprocessor directive	Use
#define	To define a macro
#include	Inserts a particular header from another file
#if	To test whether a compile-time condition is true
#undef	To undefine a macro
#else	To specify the alternative action if a test fails
#endif	To end the preprocessor condition

```c
#include<stdio.h>
#define NUMBER_OF_APPLES 24

int main(void) {
printf("There are %d apples in a basket.\n", NUMBER_OF_APPLES);
return 0;
}

// Output: There are 24 apples in a basket.
```

stdio.h is a standard header file required to use input/output functions such as printf()

int indicates that main() returns an integer value

```c
#include<stdio.h>

// define a macro ab with the value 25
#define ab 25

int main() {

/*****************************************************

When the #ifdef condition is encountered:
If ab macro is defined: #ifdef condition is executed
If ab macro is not defined: #else condition is executed

*****************************************************/

   #ifdef ab
   printf("Your lucky number is: %d\n", ab);
   #else
   printf("Error: printing lucky number!");
   #endif

   return 0;
}
```

```c
#include<stdio.h>
int main() {
int x = 16;
int y = (x++, ++x);
printf("%d", y);
return 0;
}

// Output: 18
```

Output:

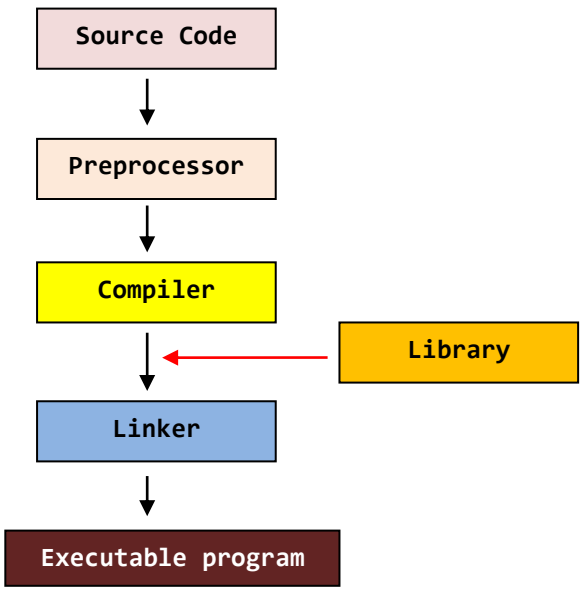
```
Your lucky number is: 25
```

```c
#include<stdio.h>
int main() {
   #ifdef ab
   printf("Your lucky number is: %d\n", ab);
   #else
   printf("Error: printing lucky number!");
   #endif

   return 0;
}
```

Output:

```
Error: printing lucky number!
```

Development of C Program:



ASCII Table

ASCII	Char	ASCII	Char	ASCII	Char	ASCII	Char
0	NUL (null)	32	SPACE	64	@	96	`
1	SOH (start of heading)	33	!	65	A	97	a
2	STX (start of text)	34	"	66	B	98	b
3	ETX (end of text)	35	#	67	C	99	c
4	EOT (end of transmission)	36	$	68	D	100	d
5	ENQ (enquiry)	37	%	69	E	101	e
6	ACK (acknowledge)	38	&	70	F	102	f
7	BEL (bell)	39	'	71	G	103	g
8	BS (backspace)	40	(	72	H	104	h
9	TAB (horizontal tab)	41	)	73	I	105	i
10	LF (NL line feed, new line)	42	*	74	J	106	j
11	VT (vertical tab)	43	+	75	K	107	k
12	FF (NP form feed, new page)	44	,	76	L	108	l
13	CR (carriage return)	45	-	77	M	109	m
14	SO (shift out)	46	.	78	N	110	n
15	SI (shift in)	47	/	79	O	111	o
16	DLE (data link escape)	48	0	80	P	112	p
17	DC1 (device control 1)	49	1	81	Q	113	q
18	DC2 (device control 2)	50	2	82	R	114	r
19	DC3 (device control 3)	51	3	83	S	115	s
20	DC4 (device control 4)	52	4	84	T	116	t
21	NAK (negative acknowledge)	53	5	85	U	117	u
22	SYN (synchronous idle)	54	6	86	V	118	v
23	ETB (end of trans. block)	55	7	87	W	119	w
24	CAN (cancel)	56	8	88	X	120	x
25	EM (end of medium)	57	9	89	Y	121	y
26	SUB (substitute)	58	:	90	Z	122	z
27	ESC (escape)	59	;	91	[	123	{
28	FS (file separator)	60	<	92	\	124	|
29	GS (group separator)	61	=	93	]	125	}
30	RS (record separator)	62	>	94	^	126	~
31	US (unit separator)	63	?	95	_	127	DEL

Data Conversion Functions

Function	Use
atof()	Converts string to float
atoi()	Converts string to int
atol()	Converts string to long
ecvt()	Converts double to string
fcvt()	Converts double to string
gcvt()	Converts double to string
itoa()	Converts int to string
ltoa()	Converts long to string
strtod()	Converts string to double
strtol()	Converts string to long integer
strtoul()	Converts string to an unsigned long integer
ultoa()	Converts unsigned long to string

Character Classification Functions

Function	Use
isalnum()	Tests for alphanumeric character
isalpha()	Tests for alphabetic character
isdigit()	Tests for decimal digit
islower()	Tests for lowercase character
isspace()	Tests for white space character
isupper()	Tests for uppercase character
isxdigit()	Tests for hexadecimal digit
tolower()	Tests character and converts to lowercase if uppercase
toupper()	Tests character and converts touppercase if lowercase

```c
#include <ctype.h>
#include <stdio.h>

int main() {
char x = '8';
if (isxdigit(x) != 0) {
     printf("%c is a hexadecimal character.", x);
   } /* end if */
else {
      printf("%c is not a hexadecimal character.", x);
   } /* end else */
return 0;
}
```

Output:

```
8 is a hexadecimal character.
```

String Manipulation Functions

Function	Use
strchr()	Appends one string to another
strchr()	Finds first occurrence of a given character in a string
strcmp()	Compares two strings
strcmpi()	Compares two strings without regard to case
strcpy()	Copies one string to another
strdup()	Duplicates a string
stricmp()	Compares two strings without regard to case (identical to strcmpi)
strlen()	Finds length of a string
strlwr()	Converts a string to lowercase
strncat()	Appends a portion of one string to another
strncmp()	Compares a portion of one string with portion of another string
strncpy()	Copies a given number of characters of one string to another
strnicmp()	Compares a portion of one string with a portion of another without regard to case
strrchr()	Finds last occurrence of a given character in a string
strrev()	Reverses a string
strset()	Sets all characters in a string to a given character
strstr()	Finds first occurrence of a given string in another string
strupr()	Converts a string to uppercase

```c
#include <stdio.h>
int main() {
double z;
int x = 5;
int y = 8;
z = (double)(x + y)/2 - (x + y)/(double)(x*x + y*y);
printf("%f", z);
return 0;
}

// Output: 6.353933
```

If marks is greater than or equal to 35 then printf("Passed\n") otherwise printf("Failed\n") is executed

```c
#include <stdio.h>
int main() {
int marks = 15;
marks >= 35 ? printf( "Passed\n" ) : printf( "Failed\n" );
return 0;
}
// Output: Failed
```

Searching and Sorting Functions

Function	Use
bsearch()	Performs binary search
lfind()	Performs linear search for a given value
qsort()	Performs quick sort

File Handling Functions

Function	Use
fopen()	opens new or existing file
fprintf()	write data into the file
fscanf()	reads data from the file
fputc()	writes a character into the file
fgetc()	reads a character from file
fclose()	closes the file
fseek()	sets the file pointer to a given position
fputw()	writes an integer to file
fgetw()	reads an integer from file
ftell()	returns current position
rewind()	sets the file pointer to the beginning of the file

```
#include <stdio.h>
int main() {
int x = 3;                    /* loop 10 times */
while(x <= 10) {
x = 3 * x;
printf("%d", x);
} /* end while */
}
```

Output:

927

Memory allocation Functions

Function	Use
malloc()	allocates the specified number of bytes

realloc()	increases or decreases the size of the specified block of memory, moving it if necessary
calloc()	allocates the specified number of bytes andinitializes them to zero
free()	releases the specified block of memory back tothe system

Directory Control Functions

- **chdir()** = Changes current working directory
- **getcwd()** = Gets current working directory
- **fnsplit()** = Splits a full path name into its components
- **findfirst()** = Searches a disk directory
- **findnext()** = Continues findfirst search
- **mkdir()** = Makes a new directory
- **rmdir()**= Removes a directory

Algorithm

A step by step procedure to solve a particular problem

Depending on what we want to do, it might be either easy or difficult

Example: Algorithm to compute the area of a sphere

```
Step 1: Read radius
Step 2: Compute Area [Area = 4 × 3.142 × radius × radius]
Step 3: Print the Area
Step 4: Stop [End of algorithm]
```

Performance:

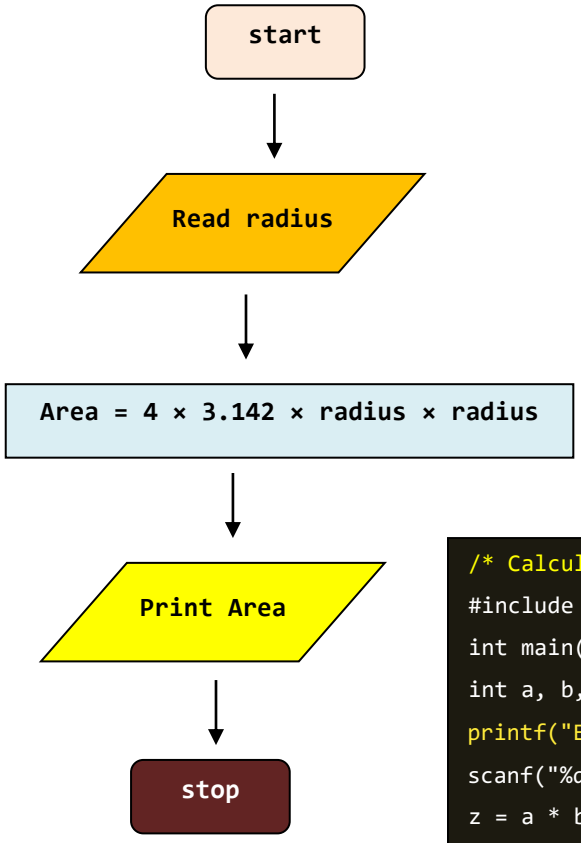
The amount of time, memory, disc space, etc. utilized when a program is executed

Flowchart

A diagrammatic representation of an algorithm

Complexity:

What happens when the size of the problem the software program is solving increases...




```c
#include <stdio.h>
int main() {
int a = 10, b = 2;
a /= b + 1;
printf("%d", a);
return 0;
}

// Output: 3
```

```c
#include <stdio.h>
int main() {
int a = 10, b = 2;
a = a/(b + 1);
printf("%d", a);
return 0;
}
```

```c
/* Calculate the product of 3 numbers */
#include <stdio.h>
int main( void ) {
int a, b, c, z; /* declare variables */
printf("Enter 3 numbers: \n"); /* prompt */
scanf("%d%d%d", &a, &b, &c); /* read 3 numbers */
z = a * b * c; /* multiply values */
printf( "The product of 3 numbers is: %d\n", z); /* display result */
return 0;
} /* end function main */
```

C Program:

```c
#include<stdio.h>
int main() {
float radius; /* radius to be input by user */
float area; /* variable in which area will be stored */
printf("Enter the radius: "); /* prompt */
scanf("%f", &radius); /* read an radius */
area = 4 * 3.14 * radius * radius; /* assign product to area */
printf("Area of a sphere = %f", area); /* print area */
return 0;
}
```

Output:

```
Enter the radius: 2.6  # entered radius

Area of a sphere = 84.905594
```

Logical statements:

```
>           Greater than          7 > 3 is TRUE
<            Less than            3 < 7 is TRUE
>=     greater than or equal      9 >= 9 is TRUE
<=      less than or equal        3 <= 7 is TRUE
==           Equal to             14 == 14 is TRUE
!=         not equal to           7 != 3 is TRUE
```

Value	With an exponent	Can also be written in C as
2.7	0.27×10^{1}	0.27E1
0.00002	0.2×10^{-4}	0.2E-4
5496.788	0.5496788×10^{4}	0.5496788E4
200.0	2.0×10^{2}	2.0E2

BASIC SYNTAX:

```
if (condition) {
Execute all statements inside the body if the condition is true
}
```

If syntax

```
if (condition) {
Execute all statements inside the body if the condition is true
} else {
Execute all statements inside the body if the condition is false
}
```

Else syntax

```
if (condition1) {
Execute all statements inside the body if condition1 is true
} else if (condition2) {
Execute all statements inside the body if the condition1 is false and condition2 is true
} else {
Execute all statements inside the body if the condition1 is false and condition2 is false
}
```

Else if syntax

Break syntax

```
while (condition) {
// statements
if (condition to break) {
break;
}
// statements
}
```

```
do {
// statements
if (condition to break) {
break;
}
// statements
}
while (condition);
```

```
for (variable initialization; condition; variable update) {
// statements
if (condition to break) {
break;
}
// statements
}
```

```
#include <stdio.h>
int main() {
int y =10;
if(y > 6) {
int x = 25;
printf( "%d", x);
} else {
int x = 50;
printf( "%d", x);
}
return 0;
}
```

```
#include <stdio.h>
int main() {
int y =10;
int x = y > 6 ? 25 : 50;
printf( "%d",  x);
return 0;
}
```

Output:

25

Continue syntax

```
while (condition) {
// statements
if (condition to break) {
continue;
}
// statements
}
```

```
do {
// statements
if (condition to break) {
continue;
}
// statements
}
while (condition);
```

```
for (variable initialization; condition; variable update) {
// statements
if (condition to break) {
continue;
}
// statements
}
```

Switch syntax

```
switch (expression) {
case 1:
// statements
break;
case 2:
// statements
break;
default:
// statements
}
```

```
#include<stdio.h>
/* function main begins program execution */
int main() {
int i = 1; /* initialization */
while(i<=10) { /* repetition condition */
printf("%d\n", i); /* display counter */
i++; /* increment */
} /* end while */
return 0; /* indicate program ended successfully */
} /* end function main */
```

Output:

```
1
2
3
4
5
6
7
8
9
10
```

For Loop syntax

```
for (variable initialization; condition; variable update) {
// statements
}
```

While Loop syntax

```
while (condition) {
// statements
}
```

Do While Loop syntax

```
do {
// statements
}
while (condition);
```

```
#include <stdio.h>
#include <string.h>
int main() {
int num = 1;
printf("We have seen %d cat%s.", num, num == 1 ? "" : "s");
return 0;
}

// Output: We have seen 1 cat.

#include <stdio.h>
#include <string.h>
int main() {
int num = 2;
printf("We have seen %d cat%s.", num, num == 1 ? "" : "s");
return 0;
}

// Output: We have seen 2 cats.
```

Goto statement:

```c
#include <stdio.h>
int main() {
int sum=0;
for(int i = 1; i<=9; i++) {
sum = sum+i;
if(i==5) {
goto addition; // Go to the statement labeled addition
}
}
addition:
printf("%d", sum);
return 0;
}

// Output: 15 # 1+2+3+4+5 = 15
```

Nested-if statement:

```c
#include <stdio.h>
int main() {
int a = 10;
if (a == 10) {
if (a < 15)
printf("\n a is smaller than 15");
if (a < 12)
printf("\n a is smaller than 12");
else
printf("\n a is greater than 15");
}
return 0;
}

// Output:
            a is smaller than 15
            a is smaller than 12
```

Nested else if Statement:

```c
#include <stdio.h>
int main() {
int a = 20;
if (a == 10)
printf("a is 10");
else if (a == 15)
printf("a is 15");
else if (a == 20)
printf("a is 20");
else
printf("a is not present");
return 0;
}

// Output: a is 20
```

```c
#include <stdio.h>
int main(void) {
 char ch = 0;
 printf("Enter A or J: ");
 scanf(" %c", &ch); // Get the choice input
 switch(ch) {
 case 'a': case 'A':
 printf("Albert...\n");
 break;
 case 'j': case 'J':
 printf("John...\n");
 break;
 default:
 printf("Mary...\n");
 break;
 }
 return 0;
}
```

The switch provides for the possibility of the user entering an uppercase or a lowercase letter A

Output:

```
Enter A or J: a # entered character
Albert...
```

```c
#include <stdio.h>
int main() {
 for(int x = 1, y = 2 ; x <= 6 ; x++, y = y + 1)
 printf(" %3d", x*y);
 return 0;
}

# Output:
               2   6  12  20  30  42
```

```c
#include <stdio.h>
int main() {
printf("Albert \" 1905 Papers \\\".");
return 0;
}

// Output: Albert " 1905 Papers \".
```

```c
#include<stdio.h> // For input and output functions
#include<ctype.h>  // For mapping characters
int main() {
for(double i = 0.0 ; i <= 3.0 ; i+= 0.3) {
printf("\nx = %.2lf", i);
}
return 0;
}
```

Output:

```
x = 0.00
x = 0.30
x = 0.60
x = 0.90
x = 1.20
x = 1.50
x = 1.80
x = 2.10
x = 2.40
x = 2.70
x = 3.00
```

```c
#include<stdio.h>
int main() {
printf("\n **************");

for(int i = 1; i <= 6; i++)

printf("\n+              +");
printf("\n **************\n");
return 0;
}
```

This expression executes once when the loop starts. It declares "i" and initializes it to 1.

This expression is executed at the end of every loop cycle. It increments "i".

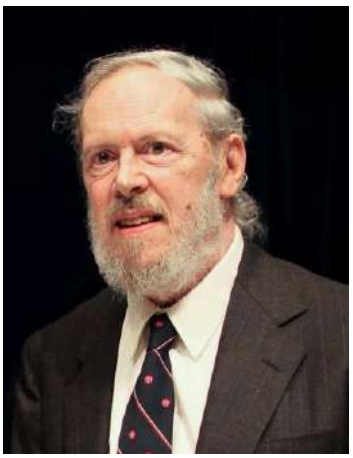
This expression is evaluated at the beginning of each loop cycle. If it is true, the loop continues, and if it is false, the loop ends.

Output:

```
**************
+            +
+            +
+            +
+            +
+            +
+            +
**************
```

C Programming

Paradigm	Imperative (procedural), structured
Designed by	**Dennis Ritchie**
Developer	Dennis Ritchie & Bell Labs (creators); ANSI X3J11 (ANSI C); ISO/IEC JTC1/SC22/WG14 (ISO C)
First appeared	1972; 48 years ago
Stable release	C18 / June 2018; 2 years ago
Typing discipline	Static, weak, manifest, nominal
OS	Cross-platform
Filename extensions	.c, .h



Dennis Ritchie

Creator of C language

Major implementations
K&R C, GCC, Clang, Intel C, C++Builder, Microsoft Visual C++, Watcom C
Dialects
Cyclone, Unified Parallel C, Split-C, Cilk, C*
Influenced by
B (BCPL, CPL), ALGOL 68, Assembly, PL/I, FORTRAN
Influenced
Numerous: AMPL, AWK, csh, C++, C--, C#, Objective-

C, D, Go, Java, JavaScript, Julia, Limbo, LPC, Perl, PHP, Pike, Processing, Python, Ring, Rust , Seed7, Vala, Verilog (HDL), Nim

which uses alphabets, digits, punctuations and some special symbols and cannot be executed directly without being converted into machine level language [the language which uses only 0 and 1]

A general-purpose, procedural, High Level Programming most widely used **computer Language** developed by a man named **Dennis Ritchie** in 1970s at **Bell Telephone laboratories (now named AT & T Bell laboratories), Murray Hill, New Jersey** to develop system application that directly interacts with the hardware devices such as drivers, kernels, etc. **[UNIX operating system]** using the two early programming languages − **Basic Combined Programming Language (BCPL)** and **BASIC (Beginner's All-purpose Symbolic Instruction Code) language.**

Uses: used in the development of

- Operating systems like **LINUX**, UNIX.
- CAD/ CAM Applications and Word processors
- Embedded systems like **ATMs**, printers.
- RDBMS MySQL, Language Compilers and Interpreters, Print Spoolers, Loaders, Linkers, Assemblers, Text Editors, Automation tools, Network Drivers.

Most of the **state-of-the-art software** has been developed using C. C has also greatly influenced many other popular languages [**considered as the base for other programming languages − mother language**], especially C++, which was originally designed as an enhancement to C. Most of the programming languages follow C syntax [**For example: C++, Java, C#, etc.**]

```c
#include <stdio.h>
#include <stdlib.h>
int main() {
    printf("Hello World");
    abort(); // abnormal termination of the program
    return 0;
}
```

Output:

```
Aborted
```

Advantages:

- Relatively simple language
- Reliable (able to be trusted)

- Easy to understand and supports a rich set of data types
- Easy to use, write, modify and debug and quick to learn
- can be compiled on a variety of computer platforms
- C supports a rich set of **operators**:

- The unary minus,!, ++, -- and ~ ⟶ **unary operators**
- The +, - *, ? are arithmetic operators. % operator is another arithmetic operator which results in the remainder after an integer division.
- The <, <=, >=, ==, != are relational operators.

- C supports **bitwise operators** – Bitwise AND, OR, XOR, bitwise complement, left and right shift operators.
- C provides a variety of **conditional control statements**

- *if*-statement
- *if-else* statement
- *nested-if* statement
- *switch* statement

The ALGOL-based language formalized in 1988 by the American National Standard Institute (**ANSI**) and can handle low-level activities

C is called a **structured programming language** because it divides the problem into smaller modules called functions or procedures each of which handles a particular responsibility. Hence it is simple and easy to understand and well suited for small size implementation. However this is not restricted. A large size implementation is possible but complex design and full **object oriented design** cannot be implemented (because complex design concepts like **Polymorphism** and **inheritance** are not available in C). C is widely used in IOT applications.

```c
#include <stdio.h>
#include <stdlib.h>
int main() {
    printf("Hello World");
    exit(EXIT_SUCCESS); // normal termination of the program
    return 0;
}
```

Output:

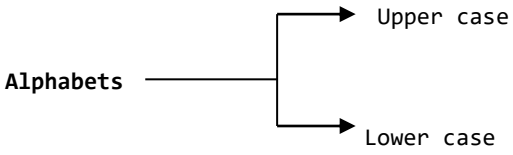
Hello World

```c
#include <stdio.h>
int main() {
char x[] = "Albert Einstein!";
puts(x);
return 0;
}
```

Output:

Albert Einstein!

Characters Set:



Digits [0-9]

```c
#include <stdio.h>
int main() {
  char x[] = {'A', 'l', 'b', 'e', 'r', 't', '\0'};
  char y[] = "Albert";
  printf("%lu\n", sizeof(x)); // Output: 7
  printf("%lu\n", sizeof(y)); // Output: 7
  return 0;
}
```

The size of both arrays is the same

Special characters:

~	Tilde
!	Exclamation mark
#	Number sign
$	Dollar sign
%	Percent sign
^	Caret
&	Ampersand
*	Asterisk
(	Lest parenthesis
)	Right parenthesis
_	Underscore
+	Plus sign
|	Vertical bar
\	Backslash
`	Apostrophe
-	Minus sign
=	Equal to sign
{	Left brace

}	Right brace
[	Left bracket
]	Right bracket
:	Colon
"	Quotation mark
;	Semicolon
<	Opening angle bracket
>	Closing angle bracket
?	Question mark
,	Comma
.	Period
/	Slash

White spaces

- Blank space
- New line
- Carriage return
- Horizontal tab

Limitation: C can't be used for internet programming like Java, .Net, PHP, etc.

```c
#include <stdio.h>
int main() {
int a, b;
a = 2, 4, 6;
b = (2, 4, 6);
printf("a = %d, b = %d\n", a, b);
return 0;
}
```

Output:

```
a = 2, b = 6
```

Features of C:

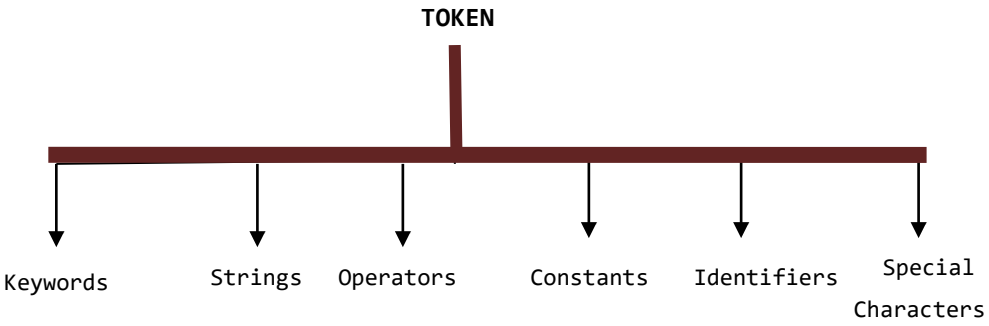
- Simple
- Machine Independent or Portable
- Mid-level programming language
- structured programming language
- Rich Library

```c
#include <stdio.h>
int main() {
int x=30; // A local variable
static int y=30; // A static variable
x=x+1;
y=y+1;
printf("%d, %d", x, y); // Output: 31, 31
return 0;
}
```

- Memory Management
- Fast Speed
- Pointers
- Recursion
- Extensible
- Robust
- Highly portable

```c
#include <stdio.h>

int main() {

int x = 20.397;

double y = 20.397;

printf("%d\n", x); // Output: 20

printf("%f\n", y); // Output: 20.397000

return 0;

}
```

TOKEN is the smallest unit in a '**C**' program.



A Simple C program basically comprises of the following parts:

- Preprocessor Commands
- Functions
- Variables
- Statements and Expressions
- Comments

```c
#include<stdio.h>
int main() {
printf("Hi\n");
main();
return 0;
}
```

The above program give raise to an infinite recursive calls to main(). The console screen will be flooded with Hi's

```c
/* My First C Program */ ──────► Comment

#include<stdio.h> ──────► Preprocessor command
```

```
int main()  ──────►  main function. Execution of C program begins from main()
{  ──────►  Beginning of the main function
printf("Hello,world!");  ──────►  Output statement
return 0;  ──────►  Terminates the execution of the main function
}  ──────►  End of the main function
```

Source Code → C preprocessor → Compiler → Object Program

Stages in C Program compilation

```
#include<stdio.h>
int x = 17; // Global variable declaration
             and initialization
int main() {
printf("%d", x); // Output: 17
return 0;
}
```

```
#include<stdio.h>
int x=17, y=18; // Global variables declaration
                   and initialization
int main() {
printf("x = %d and y = %d", x, y);
// Output: x = 17 and y = 18
return 0;
}
```

```
#include<stdio.h>
int main() {
int x, y;
x = 6;
y = x++ / 2;
printf("%d", y);
// Output: 3
return 0;
}
```

```
#include <stdio.h>
int main() {
int a = 18;
int b = 5;
printf("%ld", sizeof(a+b));
// Output: 4
return 0;
}
```

The above program displays the actual size of the given expression (a+b)

Process of C Program Execution: Every 'C' program follows a basic structure. A C program:

```
#include<stdio.h> //Pre-processor directive
int main() //main function declaration
{
printf("Hello,world!"); // Output the string on a display screen
```

```
return 0; //Terminating function

}
```

is written using Text Editor such as [Notepad (in case of Windows Operating System), vim or vi (in case of Linux Operating System)] and saved with [.c] Extension – for example, hello.c. File Saved with [.c] extension is called Source Program or Source Code. C Source code with [.c] Extension is sent to preprocessor first.

The preprocessor generates an expanded source code. **Expanded source code** is given as input to compiler where the expanded source program is compiled (i.e., the program is entirely read and translated to instructions the computer can understand i.e., machine understandable and readable language i.e., to **machine code sequence** of 0s and 1s). If the C compiler finds any error during compilation, it provides information about the error to the programmer.

The programmer has to review code and re-edit the program. After re-editing program, **Compiler** again check for any error. If program is error-free then it is sent to assembler [where the code is assembled and converted into object code. Now a simple.**obj file** is generated].

```
#include<stdio.h>
int main()
{
printf("Hello,world!");
return 0;
}
```



```
0100000000000000000
0111111111111111111
0101010101010101010
0000000111111111111
0000000000000001010
1111000000000000000
```

C compilation process converts the source code taken as input into the object code or machine code.

The object code is sent to linker (where the object code is linked with appropriate libraries). Then it is converted into a single executable code. A simple.**exe file** is generated. The executable code is sent to loader (where the executable code is loaded into memory and then it is executed). After execution, output:

```
Hello,world!
```

is displayed on the console screen.

```c
#include <stdio.h> // include the input/output related functions in C program
int main() { /* function main begins program execution */
int sum = 0; // Stores the sum of the integers
int i = 0; // The number of integers to be summed
printf("\n Enter a number: "); /* prompt */
scanf(" %d", &i); // Read the number of integers to be summed
// Sum integers from 1 to i
for(int x = 1; x<=i; sum += x++);
printf("\n The sum of the first %d integers is: %d\n", i, sum); /* print sum */
return 0; /* indicate that program ended successfully */
} /* end function main */
```

Output:

```
Enter a number: 10  # entered number

The sum of the first 10 integers is 55
```

```c
#include <stdio.h> // include the input/output related functions in C program
int main() { /* function main begins program execution */
int sum = 0; // Stores the sum of the integers
int i = 0; // The number of integers to be summed
printf("\n Enter a number: "); /* prompt */
scanf(" %d", &i); // Read the number of integers to be summed
// Sum integers from i to 1
for(int x = i; x>=1 ; sum += x--);
printf("\n The sum of the first %d integers is: %d\n", i, sum); /* print sum */
return 0; /* indicate that program ended successfully */
} /* end function main */
```

Output:

```
Enter a number: 10  # entered number

The sum of the first 10 integers is 55
```

`#include<stdio.h>` → preprocessor statement

This statement begins with # symbol and is also called **preprocessor directive**. This statement directs the **C preprocessor** to include header file [`stdio.h`] and also symbolic constants [standard input output functions] into a C program. "**stdio**" means standard input output and stdio.h means standard input output header file.

> printf() and scanf() are not part of the **C language** because there is no input or output defined in C language itself – stdio.h comprises standard input output functions like scanf, printf etc.

`stdio.h` is a **header file library** that add functionality to C programs and allows standard **input /output operations** and it is included into the C program by writing the statement `#include<stdio.h>`

- scanf is an input function and printf is an output function
- Letter "f " denote formatted

If a **C program** is written without the statement:

```
#include<stdio.h>
```

then the **C compiler** can't compile and a **compilation error** is displayedon the screen because **C compiler** fails to recognize the functions such as **printf()** and **scanf()**.

We can also write `#include "stdio.h"` **[user define library header file]** instead of `#include <stdio.h>` **[predefine library header file]** but `#include "stdio.h"` is added by programmer and `#include<stdio.h>` is already exist in compiler. So the statement `#include<stdio.h>` is generally preferred and the statement `#include "stdio.h"` is generally ignored.

`main()`→ main function

As the name itself indicates this is the **main function** of every C Program. **Parentheses** () indicate a function and the word **main** indicate the name of the function.

```
main() implies: main function.
```

Execution of **C program** begins from `main()`. No **C program** is executed without `main()`. It is case sensitive: only lower case letters (or small letters) must be used and should not be enclosed by a **semicolon**. There must be one and only one `main()` function in every **C program**.

We can represent the **main function** in various forms such as:

```
• main()
• int main()
• void main()
• main(void)
• void main(void)
• int main(void)
```

As we know **C is a Platform dependent language**. So the **Operating system** needs to know when the program execution ends. So when there is value return from the main function, the **Operating System** gets to know that the program execution is over.

`int main()` **implies:** `main()` should return integer value.

- If the main function **returns 0 to the operating system, then the program has** completed execution successfully.
- If the main function **returns 1 to the operating system, then the program** has not completed execution successfully.

```
int main() {

}
```

Body of the **main function** within which the sequence of instructions to the computer to perform specific operations in the form of statements [input-output statements, arithmetic statements, control statements and other statements] are **written and executed**

The left curly brace

```
{
```

implies: the beginning of the main function and the right curly brace

```
}
```

implies: the end of the main function. These braces can also be used to indicate the beginning and end of user-defined functions and compound statements.

`return 0;` → implies the **exit status of execution of the program** i.e., at this point, main function returns back the control of the computer to the operating system since the execution is terminated at this point and once a return statement i.e., `return 0;` executed, no further instructions within the **main function** are executed.

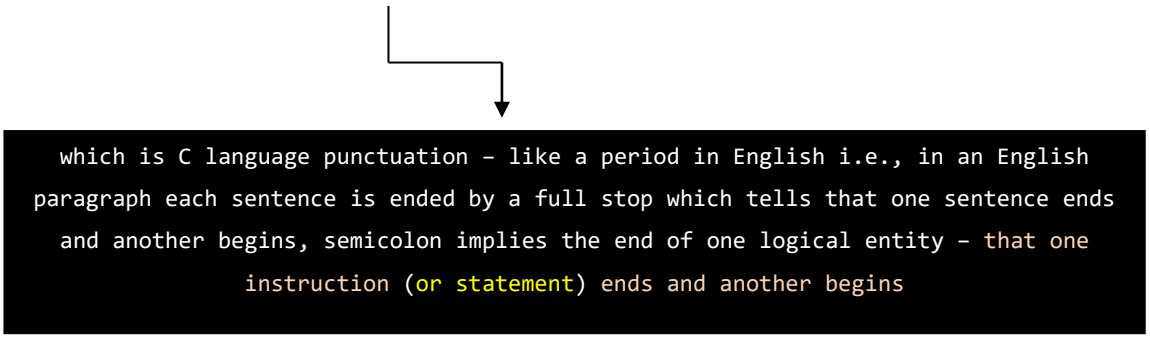
For example:

```c
#include<stdio.h>
int main() {
printf("Hello,world!");
return 0;
printf("Hello,world!");
}
```

Output on the screen:

```
Hello,world!
```

; → implies **semicolon** or **statement terminator** [a delimiter of the declaration] → In **C language**: each statement must end with a **semicolon**. A program is a well-defined set of instructions and each well-defined instruction (in the form of a statement) is ended by a semicolon



`/* My First C Program */` → **Comment**

A **good programmer** who writes codes understood by a human is better than a programmer who generates codes understood only by the machine. So, it is highly recommended to insert comments. **Comment** is explanatory note on some instruction. The statement to be commentedon must be enclosed within `/*` and `*/`. Comment statement is not compiled and executed.

`printf()`→ **the standard way of displaying output on the screen and** output function of the C language which makes provision to print the output:

```
Hello,world!
```

on the screen. **Parentheses ()** indicate a **function** and the word **printf** indicate the **name of the function**. The text: `Hello, world!` should be enclosed by the **double quotation marks** (" ") and should be written within the **printf** function and this printf function should be ended with the semicolon i.e.,

```
printf("Hello,world!");
```

Some of the standard input-output functions in C are given below:

scanf()	gets()
printf()	puts()
getchar()	getch()
putchar()	getche()

```
#include<stdio.h>
int main() {
char ch;
printf("Enter a character: ");
  ch = getchar();
  putchar(ch);

return 0;
}

# Output:
Enter a character: A # entered character
A # output
```

Take a character in the form of input and store it in the **variable** "ch"

Input function

Output function

Display the character stored in the **variable** "ch"

```
#include <stdio.h>
int main() {
char a;
char b[20];
printf( "Enter a string: " );
scanf( "%c%s", &a, b );
printf( "You entered: \n" );
printf( "the character: \"%c\" ", a );
printf( "and the string: \"%s\"\n", b );
return 0;
}
```

Output:

```
Enter a string: Albert # entered string
You entered:
the character: "A" and the string: "lbert"
```

```
#include <stdio.h> /* For puts function */
#include <stdlib.h> /* For atexit function */
void myfunc () {
   printf("Turing... \n");
}

int main() {
   atexit(myfunc);
   puts("Alan...\n");
   puts("Mathison...\n");
   return(0); /* indicates successful termination */
} /* end main */
```

Output:

```
Alan...

Mathison...

Turing...
```

```c
#include <stdio.h> /* For puts function */
#include <stdbool.h> /* For bool function */
int main(void) {
    bool a = true;  /* equivalent to bool a = 1; */
    bool b = false; /* equivalent to bool b = 0; */
    if ((a != 0) || (a != false)) { /* Functionally equivalent to if (a) */
        puts("Albert");
    }
    if ((b == 0) || (b == false)) { /* Functionally equivalent to if (!b) */
        puts("Einstein");
    }
}
```

Output:

```
Albert
Einstein
```

Computer Language	Purpose
FORTRAN	Scientific and Engineering
ALGOL	Scientific and Engineering
COBAL	Business
BASIC	Business, Scientific and Engineering
PI/1, PASCAL	General purpose

	Example	Number of bytes
Single Character	A, a, X, 5	1
Single Digit	1,3,8,0	1
Single Symbol	?, *, #,$	1
Whole Number	22,-102,3610	2
Fractional number	0.1,1.25,0.56729	4
Instruction	Add, Multiply	(1 - 3)
Big Whole / Fractional number	922112580, 12550.126	(4 - 10)

Decimal Number	Binary Number	Hexadecimal Number
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B

12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10
17	10001	11
18	10010	12
19	10011	13
20	10100	14
21	10101	15
22	10110	16
23	10111	17
24	11000	18
25	11001	19
26	11010	1A
27	11011	1B
28	11100	1C
29	11101	1D
30	11110	1E
31	11111	1F
32	100000	20
64	1000000	40
128	10000000	80
256	100000000	100

- **Program 1.1**

  C program to print the word "**Hello Bill Gates**" on screen

```c
#include<stdio.h>
int main()
{
printf("Hello Bill Gates");
return 0;
}
```

The output on the screen:

```c
#include <stdio.h>
#include <assert.h>
int main() {
int i = -5;
assert(i >= 0);
printf("i = %d\n", i);
return 0;
}
// Assertion 'i >= 0' failed
```

```c
#include <stdio.h>
#include <assert.h>
int main() {
int i = 5;
assert(i >= 0);
printf("i = %d\n", i);
return 0;
}
// Output: i = 5
```

```
Hello Bill Gates
```

- The first line tells the compiler to include standard input output header file **#include <stdio.h>** to perform reading and printing of data.
- The second line is the main function of a C program. The body of **C program** contains only two statements i.e.,

```
printf("Hello Bill Gates");
```

When this statement is taken for execution, main() calls the printf() and printf()is included in <stdio.h> . Then printf() prints: Hello Bill Gates on the computer screen.

```
return 0;
```

In C program it is optional to include "**return 0;**" at the end of the **main function** and the compiler includes it automatically.

Macro substitution: Process of replacing an identifier of a C program by a **symbolic constant**.This can be achieved by the directive **#define**.

```
#include<stdio.h>
#define PI 3.14 /* define a macro */
int main()  /* function main begins program execution */
{
float r, area; /* declare variables */
r = 2.5;
area = PI * r * r;  /* assign product to area */
printf("The area of the circle = %f", area); /* display result */
return 0; /* indicate that program ended successfully */
}  /* end function main */
```

Output:

```
The area of the circle = 19.625000
```

```
#include<stdio.h>

#define CONDITION if(x>y)
```

```c
#define PRINT printf("x is greater than y");
int main() {
int x, y;
x=16;
y=2;
CONDITION
PRINT
return 0;
}
```

Output:

x is greater than y

- Blank, tab, and newline are collectively called **whitespaces**.
- C is case sensitive that means a variable named '**age**' and '**AGE**' are different.

```c
#include<stdio.h>
int main(){
double x = 56.3;
int y = (int)x + 50; // data type casting
printf("%f\n", x); // Output: 56.300000
printf("%d\n", y); // Output: 56 + 50 = 106
return 0;
}
```

```c
#include <stdio.h>
int main() {
int y;
int x = 12;
y = (x >= 16) ? 16 : x;
printf("y = %d", y);
return 0;
}
 // Output: y = 12
```

- **Program 1.2**

  C program to print

```
    *
   *****
   *****
   *****
   *****
```

on screen

```c
#include<stdio.h>
int main()
{
```


```c
#include <stdio.h>
int main() {
int y;
int x = 12;
if (x >= 16) y = 16; else y = x;
printf("y = %d", y);
return 0;
}
// Output: y = 12
```

```
printf("\n    *      ");
printf("\n   *****    ");
printf("\n   *****    ");
printf("\n   *****    ");
printf("\n   *****    ");
return 0;
}
```

```
#include <stdio.h>
int main(void) {
enum x{physics= 65, math=95, biology=86};
printf("Physics marks: %d\n", physics);
printf("Biology marks: %d\n", biology);
return 0;
}
```

Output:

```
Physics marks: 65
Biology marks: 86
```

The output on the screen:

```
    *
   *****
   *****
   *****
   *****
```

If new line sequence `(\n)` is not included in the above program then the output on the screen is:

```
********************
```

Write a program to print the following outputs:

(a)

```
    *
   ****
  *******
   ****
    *
```

(b)

```
#include <stdio.h>
int main() {
int i = 10;
char x = 'A';
int z = i + x;
printf("%d\n", z);
return 0;
}

// Output: 75
// 10 + 65
// ASCII value of A = 65
```

```
**********
* *
* Hello World! *
* *
**********
```

```c
#include <stdio.h>
int main() {
typedef int physics;
physics marks = 86;
printf("Physics marks: %d\n", marks);
return 0;
}

// Output:
            Physics marks: 86
```

(c)

```
Braces come in pairs!
Comments come in pairs!
All statements end with a semicolon!
Spaces are optional!
Must have a main function!
C is done mostly in lowercase. It's a case-sensitive language
```

```c
#include<stdio.h>
#include<conio.h>
int main() {
const int x = 8; // x is 8
x = 15 ;  // generates an error
printf("x = %d", x);
return 0;
}
```

Because we are trying to change the value of the constant variable

Answers:

(a)

```c
#include<stdio.h>
int main()
{
printf("\n            *      ");
printf("\n          ****     ");
printf("\n        *******    ");
printf("\n          ****     ");
printf("\n            *     ");
return 0;
}
```

```c
#include <stdbool.h>
#include <stdio.h>
int main() {
int x = 30, y = 24;
bool res = ((x != y) && printf("Albert"));
return 0;
}

// Output: Albert
```

- If x is true, then: !x will be false.
- If x is false, then: !x will be true.

(b)

```
#include<stdio.h>
int main()
{
printf("\n           ***************         ");
printf("\n                * *                ");
printf("\n           * Hello World! *         ");
printf("\n                * *                ");
printf("\n           ***************         ");
return 0;
}
```

```
#include <stdbool.h>
#include <stdio.h>
int main() {
int x = 30, y = 24;
bool res = ((x == y) && printf("Albert"));
return 0;
}
// No Output
```

(c)

```
#include<stdio.h>
int main()
{
printf("\n Braces come in pairs!");
printf("\n Comments come in pairs!");
printf("\n All statements end with a semicolon!");
printf("\n  Spaces are optional!");
printf("\n Must have a main function!");
printf("\n C is done mostly in lowercase. It's a case-sensitive language");
return 0;
}
```

The **variables** represent a particular memory location where the data can be stored.

Their value can be altered and they can be reused many times.

```
int x;
float y;
char z;
Here: x, y, z represent variables. The int, float, char represent the data types.
```

```
int x = 15;
```

↓

Create a variable called x of data type **int** and assign the value 15 to it

Rules for writing variable names:

- Keywords should not be used.

- Special characters should not be used as variables
- Representing the variable names in lowercase is a **virtue programming practice**.
- There is no limit on the number of characters in a variable name.

Valid variables	Invalid variables
ab	8ab
total_mark	total mark
gross_weight_2020	gross-weight-2020
area_of_sphere	area_ _of_ _sphere

There are two methods of assigning values to variables

Method 1:

```
int x = 1;
```

Method 2:

```
int x;
x = 1;
```

- **Program 1.3**

C program to find the total surface area of a cube

```
#include<stdio.h>
/* function main begins program execution */
int main() {
int x, area; // create variables 'x' and 'area' of data type int
x = 2; // assign the value 2 to x
area = 6 * x * x; /* assign product to area */
printf("The total surface area of a cube = %d", area); // print area
return 0; /* indicate that program ended successfully */
} /* end function main */
```

The output on the screen:

```
The total surface area of a cube = 24
```

int

Keyword used to indicate an integer number. Any integer is a sequence of digits without a decimal point.

Valid	Invalid
-248	3,266
14028	-34.0
+1988	+5,456.3
0	999999999999
56780	

An attribute that tells what kind of data that value can own

int means the **data type** is **integer** and the storage size of int data type is usually **2 bytes** (consumes a total of 16 bits in memory). And it can take 232 distinct states from `-2147483648 to 2147483647`. If your computer is an 8-bit device, then the range of int is given by:

$$-2^{8-1} <= \text{integer number} <= +2^{8-1} - 1$$

$$-128 <= \text{integer number} <= +127$$

Note:

- A string, for example, is a data type that is used to categorize text and an integer is a data type used to categorize **whole numbers** or **non fractional** values [0,-5, 10].
- We can't store **decimal values** using int data type.
- If we use int data type to store decimal values, decimal values will be shortened and we will get only whole number. In this case, **float data type** can be used to store decimal values in a variable.

The statement

```
int x, area;
```

imply that we are declaring two **integer variables:**

- `x`
- `area`

and these variables represent some memory location where the data is stored. Each variable is associated with a value. The processof giving values to variables is called **assignment of values**. The assignment operator or **storage operator** (" = ") is used to assign a value to a variable.

The statements

```
x = 2;

area = 6 * x * x;
```

imply that we are assigning the values to the declared variables where:

- `x` → represents the memory location where the **value** '2' is stored
- `area` → represents the memory location where the **value** '24' is stored.

```
6 * x * x

6 * 2 * 2 = 24
```

Comma in the statement:

```
int x, area;
```

imply **variable separator**

```
tells to the compiler that the x and area
are 2 different variables
```

The statement

```
printf("The total surface area of a cube = %d", area);
```

make provision to print the output:

```
The total surface area of a cube = 24
```

on the screen.

In the statement:

```
printf("The total surface area of a cube = %d", area);
```

Format string or format specifier "%d" indicates that the integer value to be displayed after the statement:

```
The total surface area of a cube =
```

enclosed by double quotes needs to be taken from the **variable** area.

If you want to supply the value for x through the **key board**, then the statement

```
int x, area;
x = 2;
```

should be replaced by the statements:

```
int x;
int area;
printf("Enter any number:");
scanf("%d", &x);
```

i.e., the program should be rewritten as:

```
#include<stdio.h>
/* function main begins program execution */
int main() {
int x; /* number to be input by user */
int area; /* variable in which area will be stored */
printf("Enter any number: "); /* prompt */
scanf("%d", &x); /* read an number */
area = 6 * x * x; /* assign product to area */
printf("The total surface area of a cube = %d", area); /* print area */
return 0; /* indicate that program ended successfully */
} /* end function main */
```

The output on the screen:

```
Enter any number: 2 # entered number
The total surface area of a cube = 24
```

We use `print()` to put up an **output message** and `scanf()` to read the **user input** into **variable** "x".

The statement

```
printf("Enter any number: ");
```

make provision to print the text

```
Enter any number:
```

on the screen.

scanf() → formatted input function

- To read the values for the variables in a C program from the keyboard, C provides a function called **scanf()**. It is **equivalent to the** READ statement in FORTRAN or PASCAL.
- **scanf()**→ accepts numeric, character and string type of data. It is included in stdio.h.
- **&** symbol imply the address and **"&x"** imply the address of memory location where the value of input variable "x" should be stored. **"%d"** specification is used to input integer data through the keyboard.

The statement:

```
scanf("%d", &x);
```

make provision to **enter a** number for **"x"** through the **keyboard** and tells the input function **scanf**to read the number entered through the **keyboard** for "x" (which is a integer) and store it in the address of **"x"** in the computer memory (i.e., store the number in "&x").

Note:

As you **enter an integer for** "x" through the **keyboard**, this integer will be stored in the computer memory. If you want to know the **storage size** of the integer in computer memory (i.e., space occupied by the entered integer in the computer memory), you need to appeal to the following program:

```
#include<stdio.h>
/* function main begins program execution */
int main() {
int x; /* number to be input by user */
printf("Enter any number: "); /* prompt */
scanf("%d", &x); /* read an number */
printf("Size of x = %ld", sizeof(x));  /* print the size of a variable x */
return 0; /* indicate that program ended successfully */
} /* end function main */
```

The output on the screen:

```
Enter any number: 2 # entered number
Size of x = 4
```

i.e., integer entered for "x" **i.e.,** 2 has occupied **a space of 4 bytes in the computer memory**.

Field width:

```
scanf("%3d", &x);
```

The digit 3 indicates field width of the input number.

- If input number = 3485 [**4 digits**]: only 3 digits [348] are stored in the memory location of an input number. If the input number has more digits than the specified field width then all extra digits are not stored.

Unary minus:

```
a = 5, b = 6
c = a + (-b) = 5 + (-6) = -1
```

c = -1 because b was initially a positive integer variable when operated by a unary minus gets its value changed to negative

- **Format Specifiers in C**

Data Type	Specifier	Example
int	%d	14, -15, 780, +098
float	%f or %e	12.03, +5.65, 32.56 E-03
char	%c	'A', 'b', '\n', 'E'
double	%lf or %le	3.141569
long int	%ld	-218000

Void data type: A void data type doesn't contain or return any value. It is mostly used for defining functions in 'C'.

Example: void displayData()

- **Decimal constant** contains digits from 0-9.

Example: 111, 1234

- **Octal constant** contains digits from 0-7, and these types of constants are always preceded by 0.

Example: 012, 065

- **Hexadecimal constant** contains a digit from 0-9 as well as characters from A-F.

  Hexadecimal constants are always preceded by 0X.

Example: 0X2, 0Xbcd

```
#include<stdio.h>
/* function main begins program execution */
int main() {
int x; /* number to be input by user */
printf("Enter any number: "); /* prompt */
 scanf("%3d", &x);
printf("You entered:  %d", x);  /* print the entered number */
return 0; /* indicate that program ended successfully */
} /* end function main */
```

Output:

```
Enter any number: 5496 # entered number
You entered:  549
```

```c
#include<stdio.h>
int main() {
for(int i=1;i<=4;i++) {
printf("* ");
}
printf("\n");

for(int i=1;i<=4;i++) {
printf("* ");
}
printf("\n");

for(int i=1;i<=4;i++) {
printf("* ");
}
printf("\n");

for(int i=1;i<=4;i++) {
printf("* ");
}
printf("\n");

return 0;
}
```

```c
#include<stdio.h>
int main() {
for(int j=1;j<=4;j++) {
for(int i=1;i<=4;i++) {
printf("* ");
}
printf("\n");
}
return 0;
}
```

Nested loop

Output:

```
* * * *
* * * *
* * * *
* * * *
```

```c
#include<stdio.h>
int main() {
float r, area;
printf("Enter the radius: ");
scanf("%f", &r);
area = 3.14 * r * r;
printf("The area of the circle = %f", area);
return 0;
}
```

A simple example of **C language** that gets input from the user and prints the area of the circle

Output:

```
Enter the radius: 2.5 # entered radius
The area of the circle = 19.625000
```

float means the **data type** is float. This keyword is used to indicate a **floating-point number.** They are termed **floating-point** because of the shifting of the decimal point either to the left or the right of some digits during manipulation.

Mantissa-exponent notation:

$$0.000000000013 \rightarrow 1.3 \times 10^{-11} \rightarrow 1.3\ E{-}11$$

Valid	Invalid
-263.269	0,0
2.3698 E+03	-3.3.3
0.0268963 E+06	-9999999999.9999999 (out of range)
3689.6 E-04	-+44.44

The statement:

```
float r, area;
```

imply that we are declaring the **floating-point variables** r, area.

Floating point variable means fractional variable or decimal number (for example: 1.5, 2.5, 3.5, 4.7 … etc.) whereas integer means **non-fractional variable** or **whole number** (for example: 1, 2, 3, 4 … etc.)

```
#include<stdio.h>
int main() {
float length, breadth, area;
length = 2.5;
breadth = 3.2;
area = length*breadth;
printf("The area of the rectangle = %f", area);
return 0;
}

// Output: The area of the rectangle = 8.000000
```

C program to print the area of the rectangle (given length = 2.5 and breadth = 3.2)

```c
/* Print one line with three printf statements */

#include <stdio.h>

/* function main begins program execution */

int main() {

printf( "Alan " );

printf( "Mathison " );

printf( "Turing.\n" );

return 0; /* indicate that program ended successfully */

} /* end function main */


// Output: Alan Mathison Turing.
```

```c
/* Print multiple lines with a single printf */

#include <stdio.h>

int main() {

printf( "Alan\nMathison\nTuring\n" );

return 0;

}

// Output:

                    Alan

                 Mathison

                  Turing
```

```c
/* Print the sum of 2 numbers */


#include <stdio.h>

/* function main begins program execution */

int main() {

int x; /* first number to be input by user */

int y; /* second number to be input by user */

int sum; /* variable in which sum will be stored */

printf( "Enter a number: " ); /* prompt */

scanf( "%d", &x ); /* read an first number */

printf( "Enter a number: " ); /* prompt */

scanf( "%d", &y ); /* read an second number */

sum = x + y; /* assign total to sum */

printf( "Sum of %d and %d is: %d\n", x, y, sum ); /* print sum */

return 0; /* indicate that program ended successfully */

} /* end function main */
```

Output:

```
Enter a number: 5 # entered number

Enter a number: 4 # entered number

Sum of 5 and 4 is: 9
```

```c
#include <stdio.h>

int main() {

printf( "%.2f\n", 5.336 ); /* prints 5.34 */

printf( "%.1f\n", 5.336 ); /* prints 5.3 */

return 0;

}
```

Lazy initialization

A Java programming paradigm that states that a program doesn't create an object unless it is actually needed

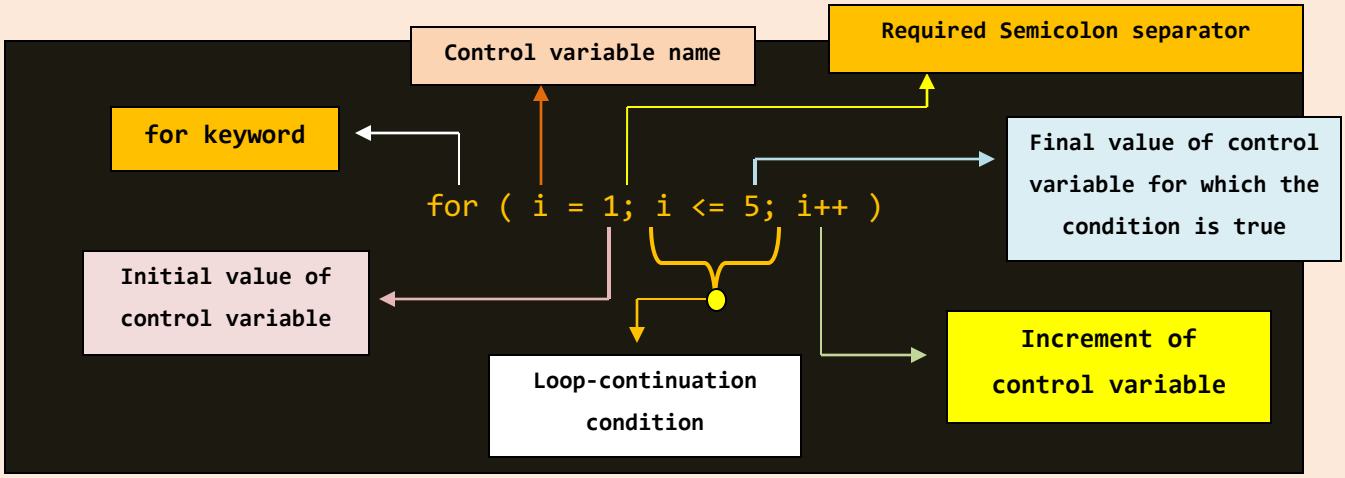
```c
#include <stdio.h>

int main() {

int x = 10; /* define and initialize x */

int i =1; /* define and initialize i */

while ( i <= 10 ) { /* loop 10 times */

// If the remainder after dividing x by 2 != 0 then "+" is printed 10 times

// otherwise "*" is printed 10 times

printf( "%s", x % 2 ? "+": "*" ); /* display result */

i++; /* increment i */

} /* end while */

return 0; /* indicate program ended successfully */

} /* end function main */
```

If int x =13;

Output: ++++++++++

Output:

****



```c
#include <stdio.h>

int main() {

int a = 2, b = 4;

 for(int i = a; i <= 2 * a * b; i += b / a) {

  printf("Hi\n");

 }

return 0;

}
```

```c
#include <stdio.h>

int main() {

int a = 2, b = 4;

 for(int i = 2; i <= 16; i += 2) {

  printf("Hi\n");

 }

return 0;

}
```

Output:

Hi

Hi

Hi

Hi

Hi

Hi

Hi

Hi

If a = 2 and b = 4, the statement: `for(int i = a; i <= 2 * a * b; i += b / a)`

is equivalent to the statement: `for(int i = 2; i <= 16; i += 2)`

```c
#include <stdio.h>

int main() {

 unsigned int x = 500;

 signed int y = -5;

 if (x > y) puts("x > y");

 else puts("x < y");

 return 0;

}

// Output: x < y
```

Since 500 is greater than -1 we would expect the output:

x > y

However this will not be the case. signed int y will get converted to unsigned int y before the comparison.

When -5 is converted to an unsigned int the result obtained is the maximum possible unsigned int value

which is greater than 500: implying that x > y is false.

```c
#include <stdio.h>

int main() {

 _Bool b = 1;

 _Bool c = 0;

 if(b) { /* Functionally equivalent to if (b == 1) */

 puts("Albert");

 }

 if (!c) { /* Functionally equivalent to if (c == 0) */

 puts("Einstein");

 }

}
```

Output:

```
Albert

Einstein
```

```c
#include <stdio.h>

#define Square(b) ((b)*(b))

int main() {

printf("%d\n", Square(3+2));

return 0;

}

// Output: 25 # (3+2)*(3+2) = 5*5=25
```

```c
#include <stdio.h>

#define Square(b) b*b

int main() {

 printf("%d\n", Square(3+2));

 return 0;

}


// Output: 11 # 3+2*3+2 = 3+6+2=11
```

Separation of Concerns

A programatic approach to divide a software application into individual manageable units with minimal overlapping between the functions of the individual units

```c
#include <stdio.h>

#define MIN(a, b) ((a) <= (b) ? (a) : (b))

int main() {

int x = 2;

printf("%d\n", MIN(x+1, 20)); // Output: 3

return 0;

}
```

```c
// ((x+1) <= (20) ? (x+1) : (20))

// ((2+1) <= (20) ? (2+1) : (20))

// ((3) <= (20) ? (3) : (20))

// Since 3 is smaller than 20: Output : 3
```

```c
#include <stdio.h>

int MIN(int a, int b) {

 return a <= b ? a : b;

}

int main() {

 int x = 2;

 printf("%d\n", MIN(x+1, 20)); // Output: 3

 return 0;

}
```

```c
#include <stdio.h>

int main() {

int x = 1905;

char y = 'X';

char* z = "Papers";

printf("%d, %c, %s", x, y, z);

return 0;

}

// Output: 1905, X, Papers
```

```c
#include <stdio.h>

int main() {

 float x;

 x = 0.2;

 if (x > 0.2) printf("%f > 0.2",  x);

 else puts("x = 0.2");

 return 0;

}
```

Prints "0.200000 > 0.2" since "x" is float

C program to find the sum of two numbers

```c
#include<stdio.h>
int main()
{
int a, b, sum;
a=1;
b=2;
sum = a + b;
printf("The sum of a and b = %d", sum);
return 0;
}
```

The output on the screen:

```
The sum of a and b = 3
```

```c
int mult(int a, int b) {
return(a*b);
}
```

mult.h

```c
#include<stdio.h>
#include<mult.h>
int main() {
int x;
x = mult(6,3);
printf("6×3 = %d\n", x);
return 0;
}
// Output: 6×3 = 18
```

If you want to assign the **floating -point numbers** i.e., fractional numbers for a and b (i.e., 1.5 for a and 2.6 for b) through the keyboard, then the statement:

```c
int a, b, sum;
```

should be replaced by the **statement:**

```c
float a, b, sum;
```

and the **statement:**

```c
printf("The sum of a and b = %d", sum);
```

should be replaced by the **statement:**

```c
#include<stdio.h>
int main(int argc, char *argv[]) {
printf("Albert");
return 0;
}
// Output: Albert
```

```
printf("The sum of a and b = %f", sum);
```

i.e.,

```
#include <stdio.h>

int main() {int x=22; printf("%d\n" , x); return 0;}

// Output: 22
```

```
#include<stdio.h>
int main()
{
float a, b, sum;
a=1.5;
b=2.6;
sum = a + b;
printf("The sum of a and b = %f", sum);
return 0;
}
```

The output on the screen:

```
The sum of a and b = 4.1
```

```
#include <stdio.h>
int main() {
/* variable declaration */
int x[3] ; /* 3 element array of type integer */
char *fruit[3] = { "Apple", "Grapes" , "Orange" } ;
for(int x = 0 ; x <= 2 ; x++ )
printf("%s\n" , fruit[x] );
return 0;
}
```

```
# Output:

Apple
Grapes
Orange
```

The statement:

```
printf("The sum of a and b = %f", sum);
```

make provision to print the output:

```
The sum of a and b = 4.1
```

```
#include <stdio.h>
int main() {
int
x
=
22
;
printf("%d\n" , x);
return 0;
}
// Output: 22
```

Even if the statement:

```
printf("The sum of a and b = %f", sum);
```

```
is omitted from the C program, the program will be successfully executed but there
will be no display of the output on the screen.
```

If you want to supply the values for **a** and **b** through the **key board**, then the statements:

```
a=1.5;
b=2.6;
```

should be replaced by the statements:

```
printf("Enter any two numbers:");
scanf("%f %f", &a, &b);
```

i.e., the **program** should be rewritten as:

```
#include<stdio.h>
int main()
{
float a, b, sum;
printf("Enter any two numbers:");
scanf("%f %f", &a, &b);
sum = a+ b;
printf("The sum of a and b = %f", sum);
return 0;
}
```

The output on the screen:

```
Enter any two numbers:

If you enter two numbers 2.9 and 3.6
```

```
#include<stdio.h>
int main() {
/* string initialized at declaration character wise */
char x[4] = {'A', 'l', 'a', 'n'};
printf("%s\n", x); // Output: Alan
return 0;
}
```

```
#include<stdio.h>
int main() {
/* string initialized at declaration: at once */
char x[4] = "Alan";
printf("%s\n", x); // Output: Alan
return 0;
}
```

```
#include<stdio.h>
int main() {
/* string initialized at declaration: at once
but not length is mentioned */
char x[] = "Alan";
printf("%s\n", x); // Output: Alan
return 0;
}
```

```
The sum of a and b = 6.5

will be outputted on the screen.
```

```
#include <stdio.h>
int main() {
int x =2, y=4;
x+=y+1; // x = x + y + 1
printf("%d\n", x);
// Output: 7
return 0;
}
```

As Said Earlier:

ampersand ("&") imply the address and **[&a** and **&b]** imply the addresses of the declared floating-point variables a and b stored in the **computer memory** i.e., when we enter a number for a and b through the keyboard, these numbers are read by scanf() function and they are stored in the computer memory (i.e., the number entered for a is stored in the address of a (i.e., stored in &a) and the number entered for b is stored in the address of b (i.e., stored in &b)).

There are **2 format strings** in the statement:

```
scanf("%f %f", &a, &b);
```

One format string "%f" corresponds to "&a" i.e., **%f** tells the **scanf() function** to read the number entered through the keyboard for "a" and store it in the **address of "a"** in the computer memory.

and the **other format string "%f"** corresponds to "&b" i.e., **%f** tells the **scanf() function** to read the number entered through the keyboard for "b" and store it in the **address of "b"** in the computer memory.

If the 2 format strings are separated by a comma i.e.,

```
scanf("%f, %f", &a, &b);
```

```
~(1010) = 0101 : 10 changes to 5

~(1111) = 0000 : 15 changes to 0

~(10000) =01111 : 16 changes to 15
```

Then the **compilation error** will be displayed on the screen.

- **Note:**

The statement:

```
scanf("%f %f", &a, &b);
```

read the two numbers 2.9 and 3.6 entered through the **keyboard** and store them in the **computer memory** [i.e., the number 2.9 is stored in the address of **"a"** (i.e., stored in &a) and the number 3.6 is stored in the address of **"b"** (i.e., stored in &b)].

The statements:

```
printf("Enter any two numbers:");
scanf("%f %f", &a, &b);
```

can also be replaced by the statements:

```
printf("Enter any number:");
scanf("%f", &a);
printf("Enter any number:");
scanf("%f", &b);
```

i.e.,

```
#include<stdio.h>
int main()
{
float a, b, sum;
printf("Enter any number:");
scanf("%f", &a);
printf("Enter any number:");
```

```
#include <stdio.h>
#include <string.h>
int main() {

//declaring variable with datatype
int salary ;
float bonus;
char name[10];

//assigning proper value
salary = 12000;
bonus = 125.596;
strcpy(name,"John");

//accessing with proper specifier
printf("Name: %s, Salary: %d, Bonus: %f", name , salary, bonus);
return 0;
}

// Output:

          Name: John, Salary: 12000, Bonus: 125.596001
```

```
scanf("%f", &b);
sum = a+ b;
printf("The sum of a and b = %f", sum);
return 0;
}
```

```
#include<stdio.h>
int main() {
char * x = ((13 % 2)==0) ? "TRUE" : "FALSE";
printf("%s\n", x);
// Output: FALSE
return 0;
}
```

Then the **output on the screen**:

```
Enter any number:
If you enter a number 2.9

Enter any number:
If you enter a number 3.6

The sum of a and b = 6.5 will be outputted on the screen.
```

Paul E. Ceruzzi, a computer historian, remarked when Ritchie passed away:

Ritchie remained unnoticed. Although his name was not well-known, if you had a microscope and could look into a computer, you would see all of his work there...

If the **statement:**

```
printf("The sum of a and b = %f", sum);
```

is replaced by the **statement:**

```
printf("The sum of %f and %f = %f", a, b, sum);
```

Then the **output on the screen** is:

```
The sum of 2.9 and 3.6 = 6.5
```

```
#include<stdio.h>
int main() {
for(int x =0 ; x < 5; x++) { // outer loop
for(int y=0; y < 5; y+=2) { // inner loop
printf("*");
}
}
return 0;
} // Output: ***************
```

In the statement

```
printf("The sum of %f and %f = %f", a, b, sum);
```

There are three format strings:

- The format string %f after the statement (The sum of) indicates that the value to be displayed needs to be taken from the variable a.
- The format string %f after the statement (The sum of %f and) indicates that the value to be displayed needs to be taken from the variable b.
- The format string %f after the statement (The sum of %f and %f =) indicates that the value to be displayed needs to be taken from the variable sum.

- **Program 1.4**

  C program to convert the temperature in Celsius to Fahrenheit

```
#include<stdio.h>
int main()
{
float C, F;
C=38.5;
F = 9*C/5 +32;
printf("Temperature in Fahrenheit= %f", F);
// %f is used because the data type for F is float
return 0;
}
```

C++ Program:

```
#include<iostream>
using namespace std;
int main() {
int x = 20;
if(x >= 0)
if(x < 10)
cout << "x < 10\n";
else
cout << "x > 10\n";
return 0;
}

// Output: x > 10
```

The output on the screen:

```
Temperature in Fahrenheit= 101.3
```

```c
#include<stdio.h>
int main() {
int x = 3;
do {
printf("Albert");
x--;
} while (x < 2);
return 0;
}
// Output: Albert
```

This program demonstrates how the do...while structure runs its program at least once. Initially x=3, loop enters, printf() prints, x = 3–1=2, now condition is checked, 2<2 = False, so loop exits

If you want to supply the number **16 digits after decimal point** for C through the key board, then the above program should take the form:

```c
#include<stdio.h>
int main()
{
double C, F;
printf("Enter any number:");
scanf("%lf", &C);
F = 9*C/5 +32;
printf("Temperature in Fahrenheit= %lf", F);
// %lf is used because the data type for F is double
return 0;
}
```

```c
#include<stdio.h>
#include<stdlib.h>

int main() {
for(int x=1; x<=6; x++) {
if(x % 4 == 0)
exit(0);
else
printf ("%d ", x);
}
return 0;
}

// Output: 1 2 3
```

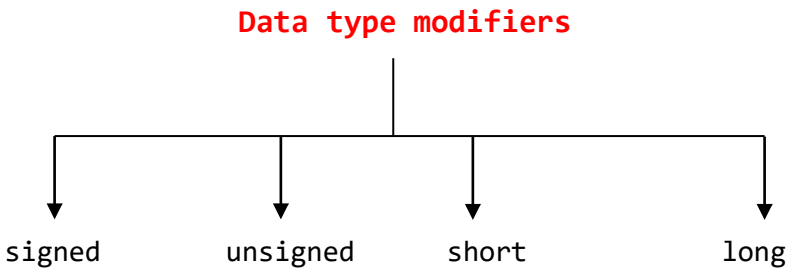
- **Note:**

- double → keyword used to denote a double precision floating-point number.
- float usually stores a maximum of 6 digits after the decimal point. But double stores 16 significant digits after the decimal point. For example:

234.0000000000000000

– 0.0000000099531510

The data type **`double`** and **`long double`** are used to store real numbers with precision up to **`14`** and **`80`** bits respectively.



Type modifier	Size in bytes
int	2
signed int	2
unsigned int	2
short int	2
long int	4
float	4
double	8
char	1
signed char	1
unsigned char	1
unsigned short int	2
unsigned long int	4
long double	10

Multiple assignment statement:

```
int x = y = 10;

float x = y = z = 6.76;
```

- **Write a program to print the sum of three numbers**

Answer:

```
#include<stdio.h>
int main()
{
int a, b, c, sum;
printf("Enter any three numbers:");
scanf("%d %d %d", &a, &b, &c);
sum = a + b + c;
printf("The sum of a, b and c = %d", sum);
return 0;
}
```

```
#include<stdio.h>
int main() {
int x = 64;
if(x > 68 && x < 54)
printf("x > 68 and < 54");
else
printf("%d", x);
return 0;
}

// Output: 64
```

- **Write a program to print the Equivalent hexadecimal value of an integer**

Answer:

```
#include<stdio.h>
int main()
{
int a = 45;
printf("%x", a);
return 0;
}
```

Output on the screen:

```
2d
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
// declaring & initializing variables
int a = 20, b = 4, c;
// doing math operations
c = a / b;
//displaying result
printf("%d / %d = %d \n", a, b, c);
// asking user to press key
printf("Enter any character to exit: \n");
// holding program till key press
getchar();
// exit program with status 0, no error
exit(0);
} // end of main
```

```
20 / 4 = 5
Enter any character to exit:
K # entered character
...Program finished with exit code 0
```

- **Write a program to print the area of a triangle when three sides are given.**

The math.h file is added to the program to use sqrt() function

Answer:

```c
#include<stdio.h>
#include<math.h>
int main()
{
int a, b, c, s, area;
a = 3;
b= 4;
c=5;
s = (a + b + c) / 2;
area = sqrt((s * (s-a) * (s-b) * (s-c)));
printf("The area of the triangle = %d", area);
return 0;
}
```

```c
#include<stdio.h>
#include<math.h>
#define PI 3.14159
int main() {
float x, y, theta, area;
printf("Enter any 2 sides of triangle: \n");
scanf("%f %f", &x, &y);
printf("Enter any one side angle in degrees: \n");
scanf("%f", &theta);
area = x * y * sin(theta * PI / 180)/2;
printf("Area of the triangle = %.4f ", area);
return 0;
}
```

Finding area of triangle when 2 sides and one angle are given

- **Note:** since sqrt() is not part of C language or of standard input output file (i.e., stdio.h file), it is part of math file i.e., (math.h file which defines various mathematical functions) **the statement:**

```c
#include<math.h>
```

should be included in the C program otherwise the **compilation error** will be flagged on the screen stating that sqrt() is not declared or defined.

- **Note:** If the statement

```
area = (s (s-a) (s-b) (s-c))^1/2
```

is written instead of

```
area = sqrt ((s * (s-a) * (s-b) * (s-c));
```

Then the **`compilation error`** will be displayed on the screen because C does not support

```
area = (s (s-a) (s-b) (s-c))^1/2
```

- **Stuff you need to know about:**

- 1 kilobyte = 2^{10} = 1024 bytes
- 1 megabyte = 2^{10} × 2^{10} = 1024 × 1024 bytes
- 1 gigabyte = 2^{10} × 2^{10} × 2^{10} = 1024 × 1024 × 1024 bytes
- 1 terabyte = 2^{10} × 2^{10} × 2^{10} × 2^{10} = 1024 × 1024 × 1024 × 1024 bytes

- **Program 1.5**

C program to find the product of two numbers

```
#include<stdio.h>
int main()
{
int a, b, product;
a=1;
b=2;
product = a * b;
printf("The product of a and b = %d", product);
return 0;
}
```

The output on the screen:

C++ Program:

```
#include<iostream>
#include<string>
using namespace std;

int main () {
string x;
cout << "Enter your name: \n";
getline(cin, x);
cout << "Hi " << x << ".\n";
return 0;
}

// Output:
       Enter your name:
       John # entered name
       Hi John.
```

```
The product of a and b = 2
```

If you want to insert a 10 digit number for a and b i.e.,

```
a=1000000000
b=3000000000
```

, then the statement:

```
int a, b, product;
```

should be replaced by the statement

```
long int a, b, product;
```

and `%ld` **should be used instead of** `%d`

i.e., the program should be rewritten as:

```
#include<stdio.h>
int main()
{
long int a, b, product;
a=1000000000;
b=2000000000;
product = a * b;
printf("The product of a and b = %ld", product);
return 0;
}
```

The output on the screen:

```
#include <stdio.h>
int main() {
printf("1 TO 5 MULTIPLICATION TABLE: \n\n");
for(int x=1; x<= 5; x++) {
for(int y=1; y<= 5; y++)
printf("%6d " , x * y);
putchar('\n');
}
return 0;
}
```

Output:

```
1 TO 5 MULTIPLICATION TABLE:

     1      2      3      4      5
     2      4      6      8     10
     3      6      9     12     15
     4      8     12     16     20
     5     10     15     20     25
```

```
#include<stdio.h>
int main() {
int x = 4, y;
y = (x < 0 ? 0 : x * x);
printf("%d", y);
return 0;
}

// Output: 16
```

```
The product of a and b = 3000000000000000000
```

If you want to supply the values for a and b **through the key board**, then the statements

```
a=1;
b=2;
```

should be replaced by the statements

```
printf("Enter any two numbers:");
scanf("%d %d", &a, &b);
```

i.e.,

```
#include<stdio.h>
int main()
{
int a, b, product;
printf("Enter any two numbers:");
scanf("%d %d", &a, &b);
product = a* b;
printf("The product of a and b = %d", product);
return 0;
}
```

```
#include <stdio.h>
#define begin {
#define end   }
#define program int main()
#define writeln printf

program
 begin
  writeln("Albert ");
  writeln("Einstein");
end

// Output: Albert Einstein
```

Pascal program syntax using **C language**

Parkinson's law

Work expands to fill the time available to complete it

The output on the screen:

```
Enter any two numbers:
```

```
If you enter two numbers 1 and 3

The product of a and b = 3 will be outputted on the screen.
```

If you replace the statements:

```
printf("Enter any two numbers:");
scanf("%d %d", &a, &b);
```

by the statements:

```
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
```

C++ Program:

```
#include <iostream>
using namespace std;
int main() {
int x;
cout << "Enter any number: ";
cin >> x;
(x % 2 == 0) ? cout << x << " is even." : cout << x << " is odd.";
return 0;
}
```

Output:

```
Enter any number: 4 # entered number
4 is even.
```

Then the output on the screen will be:

```
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 3
The product of a and b = 9
will be outputted on the screen.
```

If the statement

```
printf("The product of a and b = %d"; product);
```

is written instead of the statement

```
#include<stdio.h>
int main() {
printf ("%c", "albert"[3]);
// Output: e
return 0;
}
```

```
printf("The product of a and b = %d", product);
```

i.e., instead of **variable separator** (i.e., comma) **semicolon** is used – Then the **compilation error** will be displayed on the screen.

C++ Program:

```
#include<iostream>
using namespace std;
void func1(const string &i) {
cout << i << endl;
}
void func2(string i) {
cout << i << endl;
}
int main() {
func1("Albert");
func2("Albert");
return 0;
}
```

Output:

```
Albert
Albert
```

- **Note:**

```
#include <stdio.h>
int main()
{
printf("Hello, World!");
printf("Hello, World!\b");
printf("Hello, World!\b");
printf("Hello, World!\b");
return 0;
}
```

i.e., if **back space** "\b" is used then

```
Hello, World!Hello, World!Hello, World!Hello, World!
```

will be outputted on the screen.

If carriage return "\r" is used instead of "\b" i.e.,

```
#include <stdio.h>
int main()
{
printf("Hello, World!");
printf("Hello, World!\r");
```

```
printf("Hello, World!\r");
printf("Hello, World!\r");
return 0;
}
```

The output on the screen is:

```
Hello, World!Hello, World!
Hello, World!
Hello, World!
```

```
#include<stdio.h>
int main() {
int x;
printf("Enter any number: \n");
scanf("%d", &x);
(x == 2 ? printf("Albert") : printf("Einstein"));
return 0;
}

// Output:
              Enter any number:
              5 # entered number
              Einstein
```

If Horizontal tab "\t" is used instead of "\r" i.e.,

```
#include <stdio.h>
int main()
{
printf("Hello, World!\t");
printf("Hello, World!\t");
printf("Hello, World!\t");
printf("Hello, World!\t");
return 0;
}
```

C++ Program:

```
#include<iostream>
using namespace std;
int main() {
int x=2, y=3, z=0;
if(x!=0) {
z=x*y;
cout<<"z:"<<z;
}
return 0;
}
// Output: z:6
```

The output on the screen is:

```
Hello, World!       Hello, World!      Hello, World!      Hello, World!
```

If vertical tab "\v" is used instead of "\t" i.e.,

```c
#include <stdio.h>

int main()

{

printf("Hello, World!\v");

printf("Hello, World!\v");

printf("Hello, World!\v");

printf("Hello, World!\v");

return 0;

}
```

C++ Program:

```cpp
#include <iostream>
using namespace std;

namespace X {
int a = 25;
int b = 50;
}

namespace Y {
double a = 4.5496;
double b = 3.1988;
}

int main () {
using X::a;
using Y::b;
cout << a << endl; // 25
cout << b << endl; // 3.1988
cout << X::b << endl; // 50
cout << Y::a << endl; // 4.5496
return 0;
}
```

The output on the screen is:

```
Hello, World!
Hello, World!
Hello, World!
Hello, World!
```

- **Program 1.5**

  C program to find the square of a number

```c
#include<stdio.h>
int main()
{
int a, b;
a=2;
b = a * a;
printf("The square of a = %d", b);
}
```

```c
#include<stdio.h>

int main() {

printf("%f",16.0+17);

// Output: 33.000000

return 0;

}
```

The output on the screen:

```
The square of a = 4
```

If the **statement:**

```
b = a * a;
```

is replaced by

```
b = pow((a), 2);
```

C++ Program:

```
#include<iostream>
using namespace std;
int main() {
int x, y, z, g;
cout<<"Enter any 3 numbers:  \n";
cin>>x>>y>>z;
(x > y ? (x > z ? g = x : g = z) : (y > z ? g = y : g = z));
cout<< g << " is the greatest of 3 numbers.";
return 0;
}
```

i.e., if the **above program** is rewritten as:

```
#include<stdio.h>
#include<math.h>
int main()
{
int a, b;
a=2;
b = pow((a), 2);
printf("The square of a = %d", b);
return 0;
}
```

Output:

```
Enter any 3 numbers:
25 # entered number
36 # entered number
102 # entered number
102 is the greatest of 3 numbers.
```

Then there will be no display of **compilation error** on the screen or there will be no change in the output on the screen i.e.,

```
The square of a = 4
```

will be outputted on the screen.

which means:

```
b = pow((a), 2); is the same as b = a*a;
```

Since `b = pow((a), 2);` is used instead of `b = a*a;`

`#include<math.h>` should be included in the C program as `b = pow((a),2);` is supported by `#include<math.h>`

Otherwise **`compilation Error`** will be displayed on the screen.

If you want to supply the **`integer value`** for `"a"` through the **`key board`**, then the **statement:**

```
a=2;
```

is replaced by the statements:

```
printf("Enter any number:");
scanf("%d", &a);
```

i.e.,

```
#include<stdio.h>
int main()
{
int a, b;
printf("Enter any number:");
scanf("%d", &a);
b = a * a;
printf("The square of a = %d", b);
return 0;
}
```

C++ Program:

```
#include<iostream>
using namespace std;
int main() {
int x = 15;
int &y = x; // y is a reference variable
cout<<x<<endl; // will print 15
cout<<y<<endl; // will print 15
x=x+10;
cout<<y<<endl; // will print 25
return 0;
}
```

The output on the screen:

```
Enter any number:
If you enter a number 4
the square of a = 16 will be outputted on the screen.
```

- **Note:**
  - If `scanf(%d, &a);` is written instead of **`scanf("%d", &a);`**
  - If `printf(The square of a = %d, b);` is written instead of

    `printf("Thesquare of a = %d", b);`
  - If the **`main function`** is followed by a **`semicolon`**

    `i.e.,` `int main();` **`is written instead of`** `int main()`

    Then the **`compilation error`** will be displayed on the screen.

But if the **`body of the main function`** is followed by a semicolon `i.e.,`

```
int main()
{
};
```

is written instead of

```
int main()
{
}
```

```
#include <stdio.h>
int main() {
int a = 3, b, c;
b = a = 10;
c = a < 10;
printf("\na: %d b: %d c: %d", a, b, c);
// Output: a: 10 b: 10 c: 0
return 0;
}
```

There will be no display of the **`compilation error`** on the screen.

```
int main(); ⟶ ERROR

int main()
{
```

```
}; ⟶ NO ERROR
```

- **Write a program to print the cube of a number**

Answer:

```
#include<stdio.h>
#include<math.h>
int main()
{
int a, b;
a=2;
b = pow((a), 3);
printf("The cube of a = %d", b);
return 0;
}
```

```
#include <stdio.h>
int main() {
int x = 66;
char y = 'B';
if(x == y)
printf("Albert");
else
printf("Einstein");
return 0;
}

// Output: Albert
// ASCII value of 'B' is 66
```

- **Write a program to print the energy of the substance using energy = mc^2**

Answer:

```
#include<stdio.h>
#include<math.h>
int main()
{
int m;
long int c, energy;
m=2;
c = 300000000;
energy = m * pow((c), 2);
printf("The energy of the substance = %ld joules", energy);
return 0;
}
```

```
#include<stdio.h>
int main() {
int a = 25 ;
printf("%d %d %d", a != 25, a = 20, a < 35);
// Output: 1 20 1
return 0;
}
```

The syntax of `if statement` is:

```
if (this condition is true)
{
print this statement;
}
```

where **condition** → a logical expression that results in true or false.

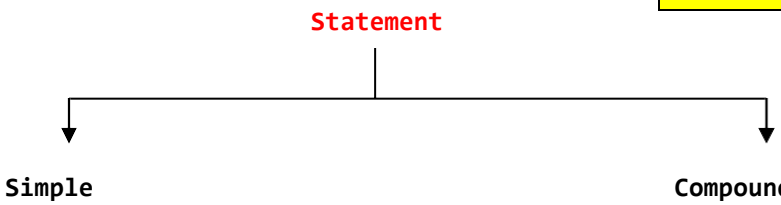
```
#include<stdio.h>
int main()
{
int a;
printf("Enter any number: \n");
scanf("%d", &a);
if((a%2)!=0){
printf("%d is an odd number\n", a);
}
```

C++ Program:

```
#include<iostream>
using namespace std;
int main() {
char string[20];
cout<<"Enter a string: "<<endl;
cin.get(string, 20);
cout<<string<<endl;
return 0;
}
```

Output:

```
Enter a string:
Alan # entered string
Alan
```



- `Simple statement` → A single statement.
- `Compound statement` → A collection of two or more statements placed between the braces.

- **Program 1.6**

  C program to find the greatest of two numbers using **`if - else`** statement

- **The syntax of `if - else` statement** (Conditional Statement):

```
if (this condition is true)
{
print this statement;
}
else
{
print this statement;
}
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
printf("Press Enter to clear the screen: ");
getchar();
system("cls"); /* Windows */
system("clear"); /* Mac - Unix */
puts("Albert Einstein");
return(0);
}
```

Output ?

If-else statement ⟶ termed as branching as a program decides which statement to execute based on the result of the evaluated condition.

```
#include<stdio.h>
int main()
{
int a, b;
a = 2;
b = 3;
if(a>b)
{
printf("a is greater than b");
}
else
{
printf("b is greater than a");
}
return 0;
}
```

```
#include <stdio.h>
int main() {
int x = 10, y = 15;
if(x % 2 == y % 3)
printf("Albert");
else
printf("Einstein");
return 0;
}

// Output: Albert
```

The output on the screen:

```
b is greater than a
```

Since the **condition**

```
a>b
```

within the parentheses is not true, the statement **a is greater than b** is not executed; instead the **execution skips** and pass to print the statement **b** is greater than **a**.

> In simpler words, (a>b) is the condition **(i.e., logical expression that results in true or false)** and if the condition (a>b) is true, then the statement:

```
printf("a is greater than b");
```

is executed to print the output:

a is greater than b

else the **statement:**

```
printf("b is greater than a");
```

is executed to print the output:

b is greater than a

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int x = 0;
x+1;
if(x<=5) {
printf("Albert Einstein");
exit(0);
main();
}
}

// Output: Albert Einstein
```

If you want to supply the **integer values for a** and **b** through the **key board**, then the statements

```
a=2;
b=3; should be replaced by the statements:
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
```

i.e., the **program** should be rewritten as:

```
#include<stdio.h>
int main()
{
int a, b;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
if(a>b)
{
printf("a is greater than b");
}
else
{
printf("b is greater than a");
}
return 0;
}
```

The output on the screen:

```
Enter any number:
If you enter the number 6
Enter any number:
If you enter the number 3
a is greater than b
```

```
#include<stdio.h>
int main() {
int x, i = 75;
x = (i > 18 ? (i <= 85 ? 500 : 1000) : 190);
printf ("%d", x) ;
return 0;
}

// Output: 500
```

```
#include<stdio.h>
int main() {
int x = 0, y = 1;
if(x == 0)
  (y > 1 ? printf("Albert") : printf("Einstein"));
else
printf("Albert Einstein");
return 0;
}

// Output: Einstein
```

```
will be outputted on the screen.
```

Nested If-else Statement:

When a series of decision is required, **nested if-else** is used. **Nesting** means using one **if-else construct** within another one.

```
#include<stdio.h>
int main()
{
        int a =1;
        if(a<10)
        {
                if(a==1)
                {
                        printf("The value is:%d\n", a);
                }
                else
                {
                        printf("The value is greater than 1");
                }
        }
        else
        {
                printf("The value is greater than 10");
        }
        return 0;
}
```

```
#include<stdio.h>
int main() {
char x[10];
printf("Enter your name: \n");
fgets(x, 4, stdin);
printf("I'm delighted to have you, %s.\n", x);
return(0);
}
```

```
Enter your name:
Albert # entered name
I'm delighted to have you, Alb.
```

The output on the screen:

```
The value is: 1
```

- **Program 1.7**

  C program to find the greatest of three numbers using `if - else if - else` statement

  - **The syntax of `if - else if - else` statement:**

```
if (this condition is true)
{
print this statement;
}
else if (this condition is true)
{
print this statement;
}
else
{
print this statement;
}
```

```
#include<stdio.h>
int main() {
static int x = 4;
printf("%d\n", x--);
if(x != 0)
main();
return 0;
}
// Output:
                    4
                    3
                    2
                    1
```

```
#include<stdio.h>
int main() {
int x=12, y;
x >= 15 ? (y = 10) : (y = 15);
printf("%d %d", x, y);
return 0;
}

// Output: 12 15
```

```
#include<stdio.h>
int main()
{
int a, b, c;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
printf("Enter any number:");
scanf("%d", &c);
if(a>b&&a>c)
{
printf("%d is greater than %d and %d", a, b, c);
}
else if (b>a&&b>c)
{
printf("%d is greater than %d and %d", b, a, c);
}
else
{
printf("%d is greater than %d and %d", c, b, a);
```

```
}
return 0;
}
```

```
#include<stdio.h>
int main() {
int x = 0;
while(x++ < 5)
printf("%d ", x);
return 0;
}

// Output: 1 2 3 4 5
```

The output on the screen:

```
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 4
4 is greater than 3 and 2
will be outputted on the screen.
```

```
#include<stdio.h>

int main() {

char x[2] = "B";

printf ("%c\n", x[0]);

// Output: B

printf ("%s", x);

// Output: B

return 0;

}
```

double `ampersand` "&&" imply and.

`(a>b&&a>c)`

`(b>a&&b>c)`

denote `conditions`.

i.e., the condition

`(a>b&&a>c)` imply:

a is greater than b and a is greater than c

and if this `condition` is true, then the **statement:**

```
printf("a is greater than b and c");
```

is executed to print the output:

```
a is greater than b and c
```

```
#include<stdio.h>
int main() {
static int x[5];
for(int i = 0; i <= 4; i++)
printf("%d ", x[i]);
// Output: 0 0 0 0 0
return 0;
}
```

and if the condition (a>b&&a>c) is not true

the **statement**

```
printf("a is greater than b and c");
```

is not executed; instead the execution skips and passes to the condition (b>a&&b>c)

and if this **condition** is true, then the statement:

```
printf("b is greater than a and c");
```

```
#include<stdio.h>
int main() {
char name[] = "Albert";
int x = 0;
while(name[x] != '\0') {
printf("%c", name[x]);
x++;
}
return 0;
}

// Output: Albert
```

is executed to print the **output**:

```
b is greater than a and c
```

and if the condition **(b>a&&b>c)** is not true, then the statement

```
printf("b is greater than a and c");
```

is not executed; instead the execution skips and the statement

```
printf("c is greater than b and a");
```

is executed to print the output:

```
c is greater than b and a
```

If the statements:

```
if(a>b&&a>c)
{
printf("%d is greater than %d and %d", a, b, c);
}
else if (b>a&&b>c)
{
printf("%d is greater than %d and %d", b, a, c);
}
else
{
printf("%d is greater than %d and %d", c, b, a);
}
```

```
#include<stdio.h>
#include<stdlib.h>

int main() {
int x, y = 3;
switch(x = y + 1) {
 case 4:
 printf ("Albert");
 break;
 default:
 printf ("Elsa");
}
return 0;
}

// Output: Albert
```

are replaced by the **statements**:

```
if(a>b&&a>c)

printf("%d is greater than %d and %d", a, b, c);

else if (b>a&&b>c)

printf("%d is greater than %d and %d", b, a, c);

else

printf("%d is greater than %d and %d", c, b, a);
```

i.e., if the program is rewritten as:

```
#include<stdio.h>
```

```
int main()
{
int a, b, c;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
printf("Enter any number:");
scanf("%d", &c);
if(a>b&&a>c)
printf("%d is greater than %d and %d", a, b, c);
else if (b>a&&b>c)
printf("%d is greater than %d and %d", b, a, c);
else
printf("%d is greater than %d and %d", c, b, a);
return 0;
}
```

```
#include<stdio.h>
int main() {
int x = 5;
while(x == 5) {
x = x - 5;
printf("%d", x);
}
return 0;
}

// Output: 0
```

There will be no display of error on the screen

c is greater than b and a

will be successfully outputted on the screen

- **What will be the output of the following program?**

```
#include <stdio.h>
int main()
{
int a, b;
a=2;
b=2;
if(a>b || a==b)
printf("a is greater than or equal to b");
else
printf("b is greater than a");
return 0;
```

```
#include<stdio.h>
int main() {
do {
printf("Albert");
} while(14 < 11);
return 0;
}

// Output: Albert
```

The loop's body is executed first and only after that is the condition tested

```
}
```

```
#include<stdio.h>
int main() {
for(int i=0;i<3;i=i+1,printf("%d ", i))
;
return(0);
}  // Output: 1 2 3
```

Answer:

```
a is greater than or equal to b
```

Note:

symbol || denote OR i.e., a>b || a==b denote **a is greater than** or **a is equal to b.**

```
#include<stdio.h>
#include <string.h>
int main() {
char x[] = "Einstein";
char y[15] = "Albert ";
strcat(y, x);
printf("%s", y);
return 0;
}

// Output: Albert Einstein
```

- **Program 1.8**

  C program to find the average of 10 numbers

```
#include<stdio.h>
int main()
{
int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, X;
printf("Enter any 10 numbers:");

scanf("%d%d%d%d%d%d%d%d%d%d", &N1, &N2, &N3, &N4, &N5, &N6, &N7, &N8, &N9, &N10);
X = (N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9 + N10) /10;
printf("The average of 10 numbers = %d", X);
return 0;
}
```

The output on the screen:

```
Enter any 10 numbers:
If you enter ten numbers 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10
The average of 10 numbers = 5
will be outputted on the screen.
```

- **Note:** The average of 10 numbers is 5.5, the output on the screen is 5 because the **data type int** is used instead of float.

- **Any mathematical expression should be written in C equivalent expression to prevent the display of compilation error on the screen because C language does not accept the general mathematical expressions.**

Mathematical expression	**C equivalent expression**
$x \times \frac{y}{z}$	x * y / z
(ax + 1) (by + 2)	(a * x + 1) * (b * y + 2)
$\frac{(a+b)^2}{(a-b)^2}$	(a+b) * (a+b) / (a-b) * (a-b) or pow((a+b), 2) / pow((a - b), 2)
$\log_{10}(\frac{x}{y} + c)$	log 10 (x/y + c)
ax^2+bx+c	a*x*x+b*x+c
lnx	log(x)
$e^x + b$	exp (x) + b
sinθ + cosθ	sin (theta) + cos (theta)
α = β + γ	alpha = beta + gamma
$\sqrt{x}$	sqrt(x)
$\sqrt[3]{x}$	cbrt(x)
$\sqrt{p^2 + q^2}$	sqrt (p*p + q*q)

$2a^2 + 3b$	2a *a + 3b + 2
$a = e^{\frac{x}{\sqrt{1+\sin\theta}}}$	a = exp (x / sqrt (1 + sin (theta)))

- **What will be the output of the following programs?**

(a)

```c
#include<stdio.h>
#include<math.h>
int main()
{
int a, b, x;
x=2;
b=2;
a = exp(x) + b;
printf("The value of a = %d", a);
return 0;
}
```

```c
#include<stdio.h>
int main() {
int i;
for(i=5; i<6; i=i+1) {
printf("%d\n", i);
}
return(0);
}

// Output: 5
```

Answer:

```
The value of a = 9
```

(b)

```c
#include<stdio.h>
int main() {
int i;
for(;i>=-5;i--)
printf("%d ",i);
return(0);
}

// Output: 0 -1 -2 -3 -4 -5
```

```c
#include<stdio.h>
#include<math.h>
int main()
{
int alpha, beta, gamma;
alpha =2;
beta=2;
gamma= 2 * alpha + beta;
printf("The value of gamma = %d", gamma);
return 0;
}
```

```c
#include<stdio.h>
#define x 5
int main() {
for(int i=1;i<=10;i++)
printf("%d %% %d = %d\n", i, x, i%x);
return(0);
}
```

Output:

```
1 % 5 = 1
2 % 5 = 2
3 % 5 = 3
4 % 5 = 4
5 % 5 = 0
6 % 5 = 1
7 % 5 = 2
8 % 5 = 3
9 % 5 = 4
10 % 5 = 0
```

Answer:

```
The value of gamma = 6
```

(c)

```c
#include <stdio.h>
#include<math.h>
int main()
{
double theta, result;
theta = 90;
result = sin(theta);
printf ("The sine 90 degrees is = %lf ", result);
return 0;
}
```

```c
#include<stdio.h>
int main() {
char *x = "Albert\n";
while(putchar(*x++))
;
return(0);
}

// Output: Albert
```

Answer:

```
The sine 90 degrees is = 0.893997
```

What is C equivalent expression of $\left[\frac{x}{y}\right]^{n-1}$?

Answer:

```
pow((x/y), n-1)
```

```
#include<stdio.h>
int main() {
char *names[] = {
 "Albert",
 "Alan",
 "John",
 "James",
 "Mary"
 };
for(int i=0;i<5;i++) {
puts(names[i]);
}
return(0);
}
```

```
# Output:

Albert
Alan
John
James
Mary
```

- **Program 1.9**

  C program to find the square root of a number

```
#include<stdio.h>
#include<math.h>
int main()
{
int a, b;
printf("Enter any number:");
scanf("%d", & a);
b = sqrt(a);
printf("The square root of a number = %d", b);
return 0;
}
```

```
#include<stdio.h>
int main() {
printf("%%14s = %14s\n","Albert");
return(0);
}

// Output: %14s =         Albert
```

The output on the screen:

```
Enter any number:

If you enter the number 4
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
 int i;
 int *pi;
 pi=&i;
 for(i=1;i<=5;i++)
 printf("%d ",*pi);
 return 0;
}

// Output: 1 2 3 4 5
```

```
The square root of a number = 2

is outputted on the screen.
```

$$\sqrt{2} = 1.41421356237$$

Suppose if you enter the number "2", the **square root of a number = 1** is outputted on the screen because **int** is used instead of **float**.

```c
#include<stdio.h>
#include<ctype.h>
int main() {
char x;
printf("Do you want to be like Albert Einstein?\n");
scanf("%c", &x);
x = toupper(x);
if(x=='Y')
puts("Albert");
else
puts("Elsa");
return(0);
}
```

Output:

```
Do you want to be like Albert Einstein?
y # entered character
Albert
```

```c
#include<stdio.h>
int main() {
char *names[] = {
 "Albert",
 "Alan",
 "John",
 "James",
 "Mary"
 };
for(int i=0;i<5;i++) {
putchar(**(names+i));
putchar('\n');
}
return(0);
}
```

Output:

```
A
A
J
J
M
```

C++ Program:

```cpp
#include<iostream>
using namespace std;
int main() {
int x = 15;
int y = x;
y += 50;
cout << y << endl;
// Output: 65
return 0;
}
```

```c
#include<stdio.h>
int main() {
char x = 'B';
switch(x) {
 case 'A':
 case 'E':
 case 'I':
 case 'O':
 case 'U':
 printf("Vowel");
 break;
 default:
 printf("Not a vowel");
}
return 0;
}

// Output: Not a vowel
```

C++ Program:

```cpp
#include<iostream>
using namespace std;

int main() {
const char x = '2';
const char y = x + 1;
const char z = '\n';
cout << y << z;
return 0;
}

// Output: 3
```

- **Note:**

```
The square root of a number = 2.000000
```

```c
#include<stdio.h>
#include<math.h>
int main()
{
printf("The square root of a number = %f", sqrt(4));

return 0;
}
```

```
|| imply or
> imply greater than
< imply less than
== imply equal to
! imply not
!= imply not equal to
&& imply and
```

```
& imply address
```

C++ Program:

```
#include<iostream>
using namespace std;
int main() {
string x = "Albert Einstein";
for(auto i : x)
cout << i;
return 0;
}

// Output: Albert Einstein
```

- **Program 2.0**

  C program to find the simple interest

```
#include<stdio.h>
int main()
{
int P,T, R, SI;
P = 1000;
T = 2;
R = 3;
SI = P*T*R/100;
printf("The simple interest = %d", SI);
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
string x = "Albert ";
string y = "Einstein";
x += y;
cout<< x <<endl;
return 0;
}
// Output: Albert Einstein
```

The output on the screen:

```
The simple interest = 60
```

- **Note:**

  If you write `SI = PTR/100;` instead of `SI=P*T*R/100;`

  Then **compilation error** is displayed on the screen because **C language** does not accept the general expressions.

If you want to supply the values for `P`, `T` and `R` through the **key board**, then the statements:

```
P = 1000;
T = 2;
```

```
R = 3;
```

should be replaced by the statements:

```
printf("Enter principal amount:");
scanf("%d", &P);
printf("Enter time:");
scanf("%d", &T);
printf("Enter rate of interest:");
scanf("%d", &R);
```

i.e., the above program should take the form:

```
#include<stdio.h>
int main()
{
int P,T, R, SI;
printf("Enter principal amount:");
scanf("%d", &P);
printf("Enter time:");
scanf("%d", &T);
printf("Enter rate of interest:");
scanf("%d", &R);
SI = P*T*R/100;
printf("The simple interest = %d", SI);
return 0;
}
```

The output on the screen:

```
Enter principal amount:
```

C++ Program:

```
#include<iostream>
using namespace std;
int main() {
string x = "Alan";
string y = "Albert";
const char *z = ", ";
string w = x + z + y + " ...";
cout<< w <<endl;
return 0;
}

// Output: Alan, Albert ...
```

```
#include<iostream>
using namespace std;
int main() {
string x = "Albert and Elsa";
x.replace(6, 4, " nor");
cout<< x <<endl;
return 0;
}

// Output: Albert nor Elsa
```

```
#include<iostream>
using namespace std;
int main() {
string x = "Albert and Elsa";
x.replace(x.begin() + 10, x.end(), " Alan");
cout<< x <<endl;
return 0;
}

// Output: Albert and Alan
```

```
If you enter the principal amount 1000
Enter time:
If you enter the time 2
Enter rate of interest:
If you enter the rate of interest 3

The simple interest = 60

will be outputted on the screen.
```

Weekly Active Users:

The number of unique users who have interacted with a platform or web application over the course of a week

goto statement: Like other languages, C supports an unconditional control statement `[goto]` to transfer the control from one point to another in a C program. The **goto** is a **branching statement** and requires a label. The syntax of **goto** statement is as follows:

```
goto label;
```

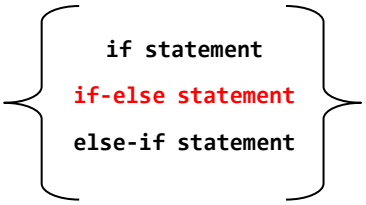
The label can be written anywhere in the **C program** either before or after the **goto** statement. For example:

```
goto END;
.....
.....
.....
END:
.....
```

The **label END:** is written after the **goto END;** statement.

```
START:
.....
.....
.....
goto START;
```

The **label START:** is written before the **goto START;** statement.



Control statements because it controls the flow of execution of a program.

- **Program 2.1**

  C program to find whether the person is senior citizen or not

```c
#include<stdio.h>
int main()
{
int age;
age=20;
if(age> = 60)
{
printf("senior citizen");
}
if(age<60)
{
printf("not a senior citizen");
}
return 0;
}
```

```cpp
#include<iostream>
#include<vector>
using namespace std;
int main() {
vector<int> primes = {2, 3, 5, 7, 11, 13};
for(auto x : primes) {
cout << x << endl;
}
return 0;
}
```

Output:

2
3
5
7
11
13

The output on the screen:

```
not a senior citizen
```

Zero-based numbering:

A method of numbering the elements in an array so that the index of the first element is zero

- `(age>= 60)` means age **greater than** or **equal to** 60

If you want to supply the value for **age** through the **key board**, then the **statement:**

```
age=20;
```

should be replaced by the statements:

```
printf("Enter age:");
scanf("%d", &age);
```

i.e., the above program should take the form:

```
#include<stdio.h>
int main()
{
int age;
printf("Enter age:");
scanf("%d", &age);
if(age>60)
{
printf("senior citizen");
}
if(age<60)
{
printf("not a senior citizen");
}
return 0;
}
```

The output on the screen:

```
Enter age:
```

```
#include<iostream>
using namespace std;
int main() {
int x = 25;
auto mult = [x](int i) { return i * x; };
cout << mult(3);
return 0;
// Output: 75
}
```

```
#include<iostream>
using namespace std;
int main() {
string x = "Alan Mathison Turing";
x.replace(x.begin() + 6, x.begin() + 9, 3, '*');
cout<<x<<endl;
return 0;
}
// Output: Alan M***ison Turing
```

```
#include<iostream>
using namespace std;
int main() {
enum names {Alan, John, James};
enum class subjects {Biology, Chemistry, Maths};
cout << is_enum<names>::value << "\n";
// Prints 1 (True)
cout << is_enum<subjects>::value << "\n";
// Prints 1 (True)
cout <<is_enum<int>::value << "\n";
// Prints 0 (False)
return 0;
}
```

```
If you enter the value 60
senior citizen will be outputted on the screen.

Suppose if you enter the value 27
not a senior citizen will be outputted on the screen.
```

```
#include<iostream>
using namespace std;
int main() {
constexpr int x = 22 * 4;
cout << x << endl;
// Output: 88
return 0;
}
```

- **Program 2.2**

  C program to get marks for 3 subjects and declare the result.

  If the marks >= 35 in all the subjects the student passes else fails.

```
#include<stdio.h>
int main()
{
int M1, M2,M3;
M1 = 38;
M2= 45;
M3 = 67;
if(M1>= 35 && M2>= 35 && M3>= 35)
{
printf("candidate is passed");
}
else
{
printf("candidate is failed");
}
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
int x = ((cout << "Albert"), 0);
return 0;
}

// Output: Albert
```

```
#include<iostream>
using namespace std;
int main() {
struct x {int a, b;};
class y {
public:
 int a, b;
};
enum class names {Albert, Alan, John};
cout << is_class<x>::value << "\n";
// Prints 1 (True)
cout << is_class<y>::value << "\n";
// Prints 1 (True)
cout << is_class<names>::value << "\n";
// Prints 0 (False)
return 0;
}
```

The output on the screen:

```
candidate is passed
```

`>=` imply **`greater than or equal to`** and `double ampersand` imply **`and`**

`(M1>= 35 && M2>= 35 && M3>= 35)` denote the condition and this condition imply `M1` **is greater than** or **equal to** `35` and `M2` **is greater than** or **equal to** `35` and `M3` **is greater than** or **equal to** `35.` And if this condition is **TRUE**, then the statement:

```
{
printf("candidate is passed");
}
is executed to print the output:
candidate is passed
else the statement
{
printf("candidate is failed");
}
```

is executed to print the output:

```
candidate is failed
```

```
#include<iostream>
using namespace std;
int main() {
for(int x = 1; x <= 2; ++x)
for(int y = 1; y <= 2; ++y)
cout << '[' << x << ',' << y << "]\n";
return 0;
}

// Output:
                    [1,1]
                    [1,2]
                    [2,1]
                    [2,2]
```

If you want to supply the integer values for marks `M1`, `M2` and `M3` through the **`key board`**, then the statements:

```
M1 = 38;
M2= 45;
M3 = 67;
```

should be replaced by the statements:

```
printf("Enter any three numbers:");
scanf("%d%d%d", &M1, &M2, &M3);
```

```
#include<iostream>
using namespace std;
int main() {
static int i = 0;
cout << ++i << " medal won on the final day.\n";
return 0;
}

// Output: 1 medal won on the final day
```

i.e.,

```c
#include<stdio.h>
int main()
{
int M1, M2, M3;
printf("Enter any three numbers:");
scanf("%d%d%d", &M1, &M2, &M3);
if(M1>= 35 && M2>= 35 && M3>= 35)
{
printf("candidate is passed");
}
else
{
printf("candidate is failed");
}
return 0;
}
```

```c
#include<stdio.h>
int main() {
int x=14; x++ & printf("%d\n", x);
// Output: 15
return 0;
}

#include<stdio.h>
int main() {
int x=14; x++ && printf("%d\n", x);
// Output: 15
return 0;
}
```

```c
#include<iostream>
struct C {
int x;
char y;
float z;
};
int main() {
std::cout << sizeof(C) << std::endl;
return 0;
}
// Output: 12
```

The output on the screen:

```
Enter any three numbers:

If you enter three numbers 26, 28, 39


candidate is failed will be outputted on the screen.
```

- **Write a program to check whether a character is an alphabet or not using the function `isalpha()`**

Object oriented

↓

A method of programming that wraps data in functions

```c
#include<stdio.h>
#include<ctype.h>
int main()
{
```

```
int a=2;
if(isalpha(a))
{
printf("The character a  is an alphabet");
}
else
{
printf("The character a  is not an alphabet");
}
return 0;
}
```

The output on the screen:

```
The character a  is not an alphabet
```

```
#include<stdio.h>
#include<ctype.h>
int main()
{
char a = 'b';
if( isalpha(a) )
{
printf("The character a  is an alphabet");
}
else
{
printf("The character a  is not an alphabet");
}
return 0;
}
```

The output on the screen:

```
The character a is an alphabet
```

```
#include<iostream>
using namespace std;
int main() {
int x{50};
cout << x << endl;
// Output: 50
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
int x = 13;
int y = -24;
int z = 30;
cout << -x << " " << -y << " " << -z << endl;
// Output: -13 24 -30
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
cout << -(14 - 15) << endl;
// Output: 1
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
cout << 10/3 << " " << 3/10 << endl;
// Output: 3  0
cout << 10.0/3.0 << " " << 3.0/10.0 << endl;
// Output: 3.33333   0.3
return 0;
}
```

If the statement `char a = b;` is written instead of `char a = 'b';` Then the **compilation error** will be flagged on the display screen.

- **Program 2.3**

  C program to find profit or loss

```c
#include<stdio.h>
int main()
{
int CP, SP, loss, profit;
printf("Enter cost price:");
scanf("%d",  &CP);
printf("Enter selling price:");
scanf("%d", &SP);
if(SP>CP)
{
printf("profit=%d", SP-CP);
}
else
{
printf("loss =%d", CP-SP);
}
return 0;
}
```

```cpp
#include<iostream>
using namespace std;
int main() {
cout << -13 + 22 << endl;
// Output: 9
cout << -(13 + 22) << endl;
// Output: -35
return 0;
}
```

```cpp
#include<iostream>
using namespace std;
int main() {
int a = 987654321; // Largest possible int value
cout << a << " + 1 = " << a + 1 << endl;
cout << a << " + 2 = " << a + 2 << endl;
cout << a << " + 3 = " << a + 3 << endl;
return 0;
}
```

Output:

```
987654321 + 1 = 987654322
987654321 + 2 = 987654323
987654321 + 3 = 987654324
```

The output on the screen:

```
Enter cost price:
If you enter the cost price 25
Enter selling price:
If you enter the selling price 26
profit = 1 will be outputted on the screen.
```

Implicit

A value whose data type has been modified automatically

If the **condition** `(SP>CP)` is true, then the statement:

```
{
printf("profit=%d", SP-CP);
}
```

is executed to print the **output:**

```
profit = SP-CP (in this case profit = 26-25 =1)
```

else the statement

```
{
printf("loss=%d", CP-SP);
}
```

is executed to print the output:

```
loss = CP-SP
```

- **Program 2.4**

  C program to convert inches into centimeter

```
#include<stdio.h>
int main()
{
float I, C;
I=3.5;
C = 2.54*I;
printf("Length in centimeters= %f", C);
return 0;
```

```
#include<iostream>
using namespace std;
int main() {
int x;
cout<<"Enter any number: \n";
cin.operator>>(x);
cout.operator<<(x);
return 0;
}

// Output:
Enter any number:
25 # entered number
25
```

```
#include<iostream>
using namespace std;
int main() {
int x;
cout<<"Enter any number: \n";
cin >> x;
cout<< x;
return 0;
}

// Output:
Enter any number:
25 # entered number
25
```

```
}
```

Array function: A user-defined task function that is specifically intended to process an array

The output on the screen:

```
Length in centimeters = 8.89
```

```
#include<iostream>
using namespace std;
int main() {
string x = "Albert";
cout << "\"" << x << "\" contains " << x.length() << " letters." << endl;
return 0;
} // Output:  "Albert" contains 6 letters.
```

If you want to supply the **floating-point variable** for "I" through the **key board**, then the above program should take the form:

```
#include<stdio.h>
int main()
{
float I, C;
printf("Enter the length in inches:");
scanf("%f", &I);
C = 2.54*I;
printf("Length in centimeters= %f", C);
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
enum Names { Alan, John, Albert, Mary};
cout << John << " " << Mary << endl;
// Output: 1 3
return 0;
}
```

Class member

An operator that is used to call functions associated with a class

The output on the screen:

```
Enter the length in inches:
25.5 # entered number
Length in centimeters = 64.9
```

Increment operator: This operator is used to increment the value of an integer number by `1`. This is represented by `'++'` [**double plus**] symbol. This symbol can be placed after the integer variable. **For example**, if `int a = 4;` then, `a++` indicate `a = a+1`. Thus, the value of `"a"` is `5`.

Decrement operator: This operator is used to reduce the value of an integer number by `1`. This is represented by `'--'` [**double minus**] symbol. This symbol can be placed after the integer variable. **For example,** if `int a = 4;` then, `a--` indicate `a = a-1`. Thus, the value of `"a"` is `3`.

- **Program 2.5**

  C program to find the **incremented** and **decremented** values of two numbers.

```c
#include<stdio.h>
int main()
{
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
printf("The incremented value of a =%d", c);
printf("The incremented value of b =%d", d);
printf("The decremented value of a =%d", e);
printf("The decremented value of b =%d", f);
return 0;
}
```

Machine dependent:

A programming language feature that alters based on the CPU of the computer

Variable:

A piece of data whose value is subject to change while the programme is running

Array

A grouping in order of similar data type elements that share a single name for identification

The output on the screen:

```
The incremented value of a = 11 The incremented value of b = 13 The decremented value
of a = 9 The decremented value of b = 11
```

If the **statements:**

```
printf("The incremented value of a =%d", c);
printf("The incremented value of b =%d", d);
printf("The decremented value of a =%d", e);
printf("The decremented value of b =%d", f);
```

Parameter passing:

How data is passed into and out of a function

are replaced by the **statements:**

```
printf("The incremented value of a =%d\n", c);
printf("The incremented value of b =%d\n", d);
printf("The decremented value of a =%d\n", e);
printf("The decremented value of b =%d\n", f);
```

Truncation:

The fractional component of a **floating-point data type** that is eliminated during integer conversion

OR

```
printf("\n The incremented value of a =%d", c);
printf("\n The incremented value of b =%d", d);
printf("\n The decremented value of a =%d", e);
printf("\n The decremented value of b =%d", f);
```

sizeof

An **operator** that displays the number of bytes a particular data type consumes up in memory

Then the output on the screen is:

```
The incremented value of a = 11
The incremented value of b = 13
The decremented value of a = 9
The decremented value of b = 11
```

Versatile:

Simple code modification to handle a different data type

```
#include<iostream>
#include<string>
using namespace std;
int main() {
string x;
cout <<"Please type one line of content here: \n";
cin >> x;
cout <<"You entered: \"" << x << "\"" << endl;
return 0;
}
```

```
Please type one line of content here:
Albert Einstein  # entered text
You entered: "Albert"
```

```
#include<iostream>
#include<string>
using namespace std;
int main() {
string x;
cout <<"Please type one line of content here: \n";
getline(cin, x);
cout <<"You entered: \"" << x << "\"" << endl;
return 0;
}
```

```
Please type one line of content here:
Albert Einstein # entered text
You entered: "Albert Einstein"
```

If you want to supply the values for `a` and `b` through **the `key board`**, then the above program should take the form:

```
#include<stdio.h>
int main()
{
int a, b, c, d, e, f;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
c=a+1;
```

Branching:

A **control structure** that permits the flow of execution to jump to another section of the program

Branching control structures

↓

Permit the flow of execution to change to another section of the program

```
d=b+1;
e=a-1;
f=b-1;
printf("The incremented value of a =%d\n", c);
printf("The incremented value of b =%d\n", d);
printf("The decremented value of a =%d\n", e);
printf("The decremented value of b =%d\n", f);
return 0;
}
```

The output on the screen:

```
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3

The incremented value of a = 3
The incremented value of b = 4
The decremented value of a = 1
The decremented value of b = 2

will be outputted on the screen.
```

```
#include<iostream>
#include<vector>
using namespace std;
int main() {
vector<int> num { 1, 2, 3, 4 };
int x;
cout << "Enter an index: \n";
cin >> x;
if(0 <= x && x< num.size()) {
cout << num.at(x) << endl;
}
else {
cout << "Vector out of bound exception" << endl;
}
}
```

Output:

```
Enter an index:
2 # entered index
3
```

Bitwise complement:

During bitwise complement operation each zero gets changed to 1 and each one gets changed to 0.

a = 10 and its equivalent binary value is 1010

b = ~a = ~ (1010) = 0101 [**which is 1's complement of a**]

Consider the binary number 10000100. It is equivalent to 132 in decimal.

- **Binary number:** 10000100
- Multiplying each digit of the **binary number** [10000100] by the corresponding power of 2:

$$1\times2^7 + 0\times2^6 + 0\times2^5 + 0\times2^4 + 0\times2^3 + 1\times2^2 + 0\times2^1 + 0\times2^0$$

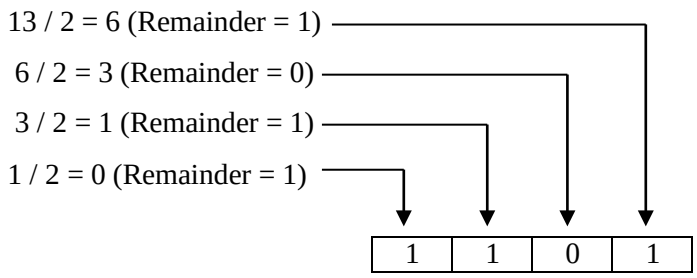
$$1\times128 + 0\times64 + 0\times32 + 0\times16 + 0\times8 + 1\times4 + 0\times2 + 0\times1$$

$$\mathbf{128 + 0 + 0 + 0 + 0 + 4 + 0 + 0 = 132}$$

132 is the decimal equivalent of the binary number **10000100**

Consider the decimal number 13. It is equivalent to 1101 in binary.

- **Decimal number:** 13
- Dividing the **Decimal number** [13] repeatedly by 2 until we get the quotient as 0:



1101 is the binary equivalent of the decimal number **13**

```
#include<stdio.h>
int main()
{
int a, b;
a=128;
b=32;
a=a>>1;
printf("After right-shifting by 1, a =%d\n", a);
b=b<<2;
printf("After left-shifting by 2, b =%d\n", b);
return 0;
```

```
}
```

The output on the screen:

```
After right-shifting by 1, a = 64
After left-shifting by 2, b = 128
```

Little's Law

The average number of items in a system is determined by multiplying the average rate at which items enter and leave the system by the average period of time each item spends in the system

- **Program 2.6**

The percentage marks are entered and the grades are allotted as follows:

percentage>= 60 First Class

percentage>=50 and per <= 60 Second Class

percentage>= 40 and per <= 50 Pass Class

percentage< 40 Fail

Write a C program for the above:

```
#include<stdio.h>
int main()
{
int P;
printf("Enter the percentage:");
scanf("%d", &P);
if(P >= 60)
{
printf("first class");
}
if(P>=50&&P <60)
{
printf("second class");
}
else if(P>=40&&P<=50 )
{
printf("pass class");
```

```
#include<iostream>
using namespace std;

int main() {
int x = 2;
cout << x << endl;
// Output: 2
cout.operator<<(x).operator<<(endl);
// Output: 2
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
string x = "Albert";
cout << x.operator[](3) << endl;
// Output: e
cout << x[3] << endl;
// Output: e
return 0;
}
```

The letter at index 3

```
}
else
{
printf("fail");
}
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
int x = 0;
while (x < 5) {
cout << "*";
x++;
}
cout << endl;
return 0;
}

// Output: *****
```

The output on the screen:

```
Enter the percentage:
If you enter the percentage 65
first class will be outputted on the screen.
```

- **Program 2.7**

  C program to calculate the discounted price and the total price after discount

  Given:

  - If purchase value is greater than 1000, 10% discount
  - If purchase value is greater than 5000, 20% discount
  - If purchase value is greater than 10000, 30% discount

- **Discounted price**

```
#include<stdio.h>
int main()
{
double PV, dis;
printf("Enter purchased value:");
scanf("%lf", &PV);
if(PV>1000)
{
printf("dis=%lf", PV* 0.1);
```

```
#include<iostream>
using namespace std;
int main() {
int x = 5;
while (x < 6) {
if (x % 5 == 0)
cout << "*";
x++;
}
cout << endl;
return 0;
}

// Output: *
```

```
}
else if(PV>5000)
{
printf("dis =%lf", PV* 0.2);
}
else
{
printf("dis=%lf", PV* 0.3);
}
return 0;
}
```

Java Program:

```
public class MyClass {
public static void main(String[] args) {
Boolean x = Boolean.TRUE;
if (x) {
System.out.println("Albert");
}
}
}

// Output: Albert
```

The output on the screen:

```
Enter purchased value:
If you enter the purchased value 6500
dis = 1300.000000 will be outputted on the screen.
```

- **Total price**

```
#include<stdio.h>
int main()
{
double PV, total;
printf("Enter purchased value:");
scanf("%lf", &PV);
if(PV<1000)
{
printf("total=%lf", PV - PV* 0.1);
}
else if(PV<5000)
{
printf("total =%lf", PV- PV* 0.2);
```

```
import java.util.Scanner;
public class MyClass {
public static void main(String[] args) {
String x;
Scanner scan = new Scanner(System.in);
do {
x = scan.nextLine();
} while (!x.equals("Mary"));
System.out.println(x);
}
}
```

Output:

```
John # entered string
James # entered string
Albert # entered string
Mary # entered string
Mary
```

```
}
else
{
printf("total=%lf", PV- PV* 0.3);
}
return 0;
}
```

```
public class MyClass {

public static void main(String[] args) {

String x = String.format("%.2f", 1.5496);

System.out.println(x);

// Output: 1.55

}

}
```

The output on the screen:

```
Enter purchased value:
If you enter the purchased value 650
total = 585.000000  will be outputted on the screen.
```

- **Now, Combing both the programs (above), we can write:**

```
#include<stdio.h>
int main()
{
double PV, dis, total;
printf("Enter purchased value:");
scanf("%lf", &PV);
if(PV>1000)
{
printf("dis=%lf", PV* 0.1);
printf("total=%lf", PV - PV* 0.1);
}
else if(PV>5000)
{
printf("dis =%lf", PV* 0.2);
printf("total=%lf", PV - PV* 0.2);
}
else
{
```

```
public class MyClass {

public static void main(String[] args) {

String x = String.format("%.3f", 1.5496);

System.out.println(x);

// Output: 1.550

}

}
```

```
printf("dis=%lf", PV* 0.3);
printf("total=%lf", PV - PV* 0.3);
}
return 0;
}
```

The output on the screen:

```
Enter purchased value:
If you enter the purchased value 850
dis = 85.000000
total = 765.000000
will be outputted on the screen.
```

```
import java.text.DecimalFormat;
public class MyClass {
public static void main(String[] args) {
String x = new
DecimalFormat("0.#").format(5.6400);
System.out.println(x);
// Output: 5.6
}
}
```

```
import java.text.DecimalFormat;
public class MyClass {
public static void main(String[] args) {
String x = new DecimalFormat("0.##").format(1.6534000);
System.out.println(x);
// Output: 1.65
}
}
```

```
public class MyClass {
public static void main(String[] args) {
String[] fruits = {"Apple", "Mango", "Grapes", "Orange"};
for (String x : fruits) {
// Grapes is not printed out
if (x.equals("Grapes"))
continue;
System.out.println(x);
}
}
}
```

Output:

Apple
Mango
Orange

Jakob's law

Users expect that your website function similarly to all the other websites they are already familiar with

What is a Loop?

A Loop executes the sequence of statements many times until the stated condition becomes false.

Types of Loops: Depending upon the position of a control statement in a program, a loop is classified into two types:

- Entry controlled loop → a condition is checked before executing the body of a loop. It is also called as a pre-checking loop.
- Exit controlled loop → a condition is checked after executing the body of a loop. It is also called as a post-checking loop.

- **Program 2.8**

  C program to print the first ten natural numbers using **`for loop`** statement

```c
#include<stdio.h>
int main()
{
int i;
for(i=1; i<=10; i++)
printf("value of i =%d", i);
return 0;
}
```

```java
import java.text.DecimalFormat;

public class MyClass {
public static void main(String[] args) {
String x = new DecimalFormat("0").format(987653421);
System.out.println(x);
// Output: 987653421
}
}
```

The output on the screen is:

```
value of i = 1 value of i = 2  value of i= 3  value of i= 4  value of i= 5  value of i= 6 value of i = 7 value of i= 8  value of i = 9  value of i = 10
```

`for(i=1; i<=10; i++)` denote the **`for loop statement`** and the **syntax** of the `for loop statement is:`

```
for(initialization; condition; increment)
```

Here:

```
i=1 denote initialization (i.e., from where to start)
i<=10 denote the condition (i.e., stop when 10 is reached)
i++ implies increment (which tells the value of i to increase by 1 each time the loop
is executed) and i++ is the same as i+1.
```

The **number of iterations** required to execute the body of **`for loop`** is computed using the formula:

$$\text{Number of iteration} = \frac{(\text{Final value} - \text{initial value} + \text{step increment})}{\text{step increment}}$$

- **When a `for loop` executes, the following occurs:**

```
i = 1
Is the condition (i<=10) is true?
Yes because i=1
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 1
Now, the value of i is:
i =  1+1 = 2
Is the condition (i<=10) is true?
Yes because i=2
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 2
```

```
Now, the value of i is:
i =  2+1 = 3
Is the condition (i<=10) is true?
Yes because i=3
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 3
Now, the value of i is:
i =  3+1 = 4
Is the condition (i<=10) is true?
Yes because i=4
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 4
Now, the value of i is:
i =  4+1 = 5
Is the condition (i<=10) is true?
Yes because i=5
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 5
Now, the value of i is:
i =  5+1 = 6
Is the condition (i<=10) is true?
Yes because i=6
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 6
Now, the value of i is:
i =  6+1 = 7
Is the condition (i<=10) is true?
Yes because i=7
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 7
Now, the value of i is:
i =  7+1 = 8
Is the condition (i<=10) is true?
Yes because i=8
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 8
Now, the value of i is:
i =  8+1 = 9
Is the condition (i<=10) is true?
Yes because i=9
The statement printf("value of i =%d", i); is executed to print the output:
```

```
value of i = 9
Now, the value of i is:
i =  9+1 = 10
Is the condition (i<=10) is true?
Yes because i=10
The statement printf("value of i =%d", i); is executed to print the output:
value of i = 10
and stop because the condition i<=10 is achieved.
```

If the **statement:**

```
printf("value of i =%d", i);
```

is replaced by the **statement**:

```
printf("value of i =%d\n", i);
```

OR

```
printf("\n value of i =%d", i);
```

Then the **output on the screen** is:

```
value of i = 1
value of i = 2
value of i = 3
value of i = 4
value of i = 5
```

```
public class MyClass {
public static void main(String[] args) {
int x = 654321;
int y = 654_321;
System.out.println(x == y);
// Output: true
}
}
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println(null + " is NULL");
// Output: null is NULL
}
}
```

```
public class MyClass {
public static void main(String[] args) {
int x = 3;
System.out.println(7 * (x > 0 ? 4 : 8));
// Output: 28
}
}
```

```
value of i = 6
value of i = 7
value of i = 8
value of i = 9
value of i = 10
```

```
public class MyClass {
public static void main(String[] args) {
int x = 2147483647;
System.out.println(x); // prints 2147483647
x = x + 1;
System.out.println(x); // prints -2147483648
}
}
```

If the `for loop` statement:

```
for(i=2; i<=10; i++)
```

is written instead of the **statement**:

`for(i=1; i<=10; i++)`, then the **output on the screen** is:

```
value of i = 2  value of i = 3  value of i= 4  value of i= 5  value of i= 6 value of i = 7 value of i= 8  value of i = 9  value of i= 10
```

If the `for loop` statement:

```
for(i=1; i<10; i++)
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println(Integer.MAX_VALUE);
// Output: 2147483647 (maximum value of int)
System.out.println(Integer.MIN_VALUE);
// Output: -2147483648 (minimum value of int)
}
}
```

is written instead of the **statement:**

```
for(i=1; i<=10; i++)
```

```
public class MyClass {
public static void main(String[] args) {
char x = '\u2764';
System.out.println(Character.toString(x));
// Output: ♥
}
}
```

then the **output on the screen** is:

```
value of i = 1 value of i = 2  value of i= 3  value of i= 4  value of i= 5  value of i= 6 value of i = 7 value of i= 8  value of i = 9
```

> Note: the condition i<=10 tells to print till value of i =10 but the condition i<10 tells to print till value of i=9.

If the statement:

```
for(i=1; i=10; i++)
```

is written instead of the statement:

```
for(i=1; i<=10; i++)
```

then the output on the screen is:

```
value of i = 10  value of i = 10  value of i = 10  value of  i = 10  value of i= 10
value of i= 10 value of i = 10 value of i= 10  value of i = 10  value of i = 10
value of i = 10 value of i = 10  value of i = 10 value of i = 10 value of i = 10
continues ....
```

```
#include <stdio.h>
int main() {
for(int i = 1; i != 5; i++) {
printf( "%f\n", (float) i / 10 );
}
return 0;
}
```

Output:

```
0.100000
0.200000
0.300000
0.400000
```

Note:

If the statement:

```
printf("value of i =%d", i);
```

is replaced by the statement

```
#include <stdio.h>
void myfunc(int* x, double* y) {
 *x = 64;
 *y = 296.0;
}

int main() {
int w = 0;
double z = 0.0;
myfunc(&w , &z);
printf("w: %d, z: %f\n", w , z);
return 0;
}
```

Output:

```
w: 64, z: 296.000000
```

```
printf("%d\n", i);
```

Then the **output on the screen** is:

```
1
2
3
4
5
6
7
8
9
10
```

```
#include <stdio.h>
int main() {
for(int i = 2; i <= 10; i += 2) {
printf( "%d\n", i );
}
return 0;
}

# Output:

                    2
                    4
                    6
                    8
                    10
```

- **What will be the output of the following program?**

```
#include<stdio.h>
int main()
{
int i; /* define i */
for(i=1; i<=5; i++) // i starts at 1 and finishes with 5
printf("Linux is not portable\n", i);
return 0;
}
```

Answer:

```
Linux is not portable
```

```
#include<stdio.h>
#include<math.h>
int main() {
float x = 13.0, y = 3.0, z = 4.0;
printf("%.2f", sqrt( x + y * z));
return 0;
}

# Output:
                    5.00
```

```
Linux is not portable
Linux is not portable
Linux is not portable
Linux is not portable
```

- **C program to print the first ten natural numbers using while loop statement:**

The syntax of while loop statement is:

```
while (this is the condition)
{
Execute this statement;
}
```

```
#include<stdio.h>
int main()
{
int i = 1;
while (i<=10)
{
printf("%d\n", i++);
}
return 0;
}
```

The output on the screen is:

```
1
2
3
```

```
#include <stdio.h>
int main() {
int a = 25;
int b = 92;
printf("%d, %d\n", 1 ? a : b, 0 ? a : b);
return 0;
}

// Output: 25, 92
```

```
#include <stdio.h>
int main() {
int a = 102, b = 102;
printf("%d\n", (a *= 3, b));
return 0;
}

// Output: 102
```

```
4
5
6
7
8
9
10
```

```c
#include <stdio.h>
int main() {
int x[] = { 11, 12, 13, 14, 15 };
printf("x[3] = %d\n", x[3]);
printf("4[x] = %d\n", 4[x]);
return 0;
}
```

Output:

```
x[3] = 14
4[x] = 15
```

`(i<=10)` is the condition and

The statement:

```c
printf("%d\n", i++);
```

is repeatedly executed as long as a given condition (`i<=10`) is true.

Ken Thompson (left) with Dennis Ritchie

Creator of 'B' programming language

If the statement:

```c
int i=1;
```

is replaced by the statement:

```c
int i;
```

Then the **compilation error** will be displayed on the **console screen** because initialization is notdefined i.e., from where to start is not declared.

If the statement:

```c
int i = 1;
```

is replaced by

```c
int i = 0;
```

```c
#include <stdio.h>
#include <string.h>
int main() {
 char x[] = "albert";
 char y[8];
 strcpy(y, x); // copying x to y
 printf("%s\n", y);
/* "albert" will be printed */
 return 0;
}
```

Then the **output on the screen** is:

```
0
1
2
3
4
5
6
7
8
9
10
```

```
#include <stdio.h>
int main() {
char *x = NULL;
printf("%s", x);
return 0;
}

// Output: (null)
```

Similarly if the statement `int i = 0;` is replaced by the `int i = 7;`

Then the output on the screen is:

```
7
8
9
10
```

- **C program to print first 10 numbers using do while loop statement**

The syntax of do while loop statement is:

```
do
{
Execute this statement;
}
while(this is the condition);
```

```
#include<stdio.h>
int main()
```

```
#include <stdio.h>
int main() {
 int x = 63;
 int y = -25;
 /* 'y == -25' is not evaluated,
 since 'x != 63' is false. */
 if (x != 63 && y == -25) {
 printf("Albert\n");
 }
 else {
 printf("Einstein\n");
 }
 return 0;
}

// Output: Einstein
```

```
{
int i =1;
do
{
printf("i= %d\n", i++);
} while (i<=10);
return 0;
}
```

The output on the screen is:

```
i=1
i=2
i=3
i=4
i=5
i=6
i=7
i=8
i=9
i=10
```

The statement:

```
printf("i= %d\n", i++);
```

is executed and then condition (i<=10) is checked. If condition (i<=10) is true then

The statement:

```
#include <stdio.h>

#define max(a, b) ((a) > (b) ? (a) : (b))

int main() {

int x = max(22, 53); /* Output: 53 */

printf("%d\n", x);

int y = max(22 + 57, 75 - 16); /* Output: 79 */

printf("%d\n", y);

return 0;

}
```

```
#include <stdio.h>
int main() {
int x = 5;
if ((x > 1) && (x > 2)) {
puts("x is larger than 1 and 2");
}
else {
 puts("x is not larger than 1 and 2");
}
return 0;
}

// Output: x is larger than 1 and 2
```

```
printf("i= %d\n", i++);
```

is executed again. This process repeats until the given **condition** `(i<=10)` becomes false.

- **Why LOOP is USED?**

If loop is not used then the C program to print first 10 numbers should be written as follows:

```
#include<stdio.h>
int main()
{
printf("\n i = 1");
printf("\n i = 2");
printf("\n i = 3");
printf("\n i = 4");
printf("\n i = 5");
printf("\n i = 6");
printf("\n i = 7");
printf("\n i = 8");
printf("\n i = 9");
printf("\n i = 10");
return 0;
}
```

```
#include <stdio.h>
#include <string.h>
int main() {
const char *x = "Alan Turing"; /* initialize char pointer */
char i = 'a'; /* initialize 'i' */
/* if 'i' is present in 'x' */
if (strchr(x, i) != NULL) {
printf( "\'%c\' is present in \"%s\".\n", i, x);
} /* end if */
else { /* if 'i' is not present in 'x' */
printf( "\'%c\' is not present in \"%s\".\n", i, x);
} /* end else */
return 0;
}
```

Output:

```
'a' is present in "Alan Turing".
```

It takes pretty long time to write the code and the execution time is pretty long i.e., because to reduce the time taken to write the code and to reduce the execution time – `loop statement` is used.

- **Write a program to print:**

```
Never test for an error condition you don't know how to handle
```

5 times using for loop statement.

Answer:

```
#include<stdio.h>
int main()
{
int i;
for (i =1; i<=5; i ++)
printf("Never test for an error condition you don't know how to handle \n");
return 0;
}
```

- **Note:**

For the program:

```
#include<stdio.h>
int main()
{
int i;
for (i=1; i=5; i++)
printf("Linux is not portable");
return 0;
}
```

```
#include <stdio.h>
#include <string.h>
int main() {
const char *x = "Alan Turing"; /* initialize char pointer */
char i = 'z'; /* initialize 'i' */
/* if 'i' is present in 'x' */
if (strchr(x, i) != NULL) {
printf( "\'%c\' is present in \"%s\".\n", i, x);
} /* end if */
else { /* if 'i' is not present in 'x' */
printf( "\'%c\' is not present in \"%s\".\n", i, x);
} /* end else */
return 0;
}
```

Output:

```
'z' is not present in "Alan Turing".
```

The output on the screen is:

```
Linux is not portable Linux is not portable Linux is not portable Linux is not portable
Linux is not portable Linux is not portable Linux is not portable Linux is not portable
Linux is not portable Linux is not portable Linux is not portable Linux is not portable
Linux is not portable …. continues
```

- **Program 2.9**

  C program to print the characters from A to Z using for loop, do while loop and while loop statement.

  - **C program to print the characters from A to Z using `for loop` statement:**

```c
#include<stdio.h>
int main()
{
char a;
for(a='A'; a<='Z'; a++)
printf("%c\n", a);
return 0;
}
```

The output on the screen:

```
A
B
C
D
E
F
G
H
I
J
K
L
M
N
```

```c
#include <stdio.h>
#include <string.h>
int main () {
   const char x[] = "Users/Manju/1.txt";
   const char i = '/';
   char *y;

   y = strrchr(x, i);

   printf("String after '%c' is: '%s'\n", i, y);
   return(0);
}
```

Searches for the last occurrence of the character i ('/') in the string x ('Users/Manju/1.txt')

Output:

```
String after '/' is: '/1.txt'
```

```c
#include <stdio.h>
int main() {
const char *x = "Albert"; /* char pointer */
printf("%s\n", x);
return 0; /* indicates successful termination */
} /* end main */

// Output: Albert
```

```
O
P
Q
R
S
T
W
X
Y
Z
```

```c
#include <stdio.h>
int main() {
printf("%10s%10d%10c%10f\n\n", "Albert", 19, 'E', 1.05 );
return 0;
}
```

Output:

```
    Albert        19         E  1.050000
```

char means the **data type** is character.

- char → Keyword used to denote the character type data and takes 8-bits for storage.
- A **character constant** can be termed as any single character enclosed within a pair of apostrophes ['a','c','2','$','?', etc.].
- A **string constant** is termed as a sequence of printable **ASCII characters** placed between double quotes. Examples: "computer","keyboard","Hello","Ram", etc.

The statement

`char a;` imply that we are creating the character `'a'`.

Since `char a` is used. Therefore: the **format specifier** `%c` should be used instead of `%d` or `%f` otherwise `error` will be flagged on the screen.

If the statement

```c
for(a=A; a<=Z; a++)
```

is written instead of the statement

```c
#include <stdio.h>
int main() {
printf( "%d\n%d\n", -549, 549 );
printf( "%+d\n%+d\n", 549, -549 );
return 0;
}
```

Output

```
-549
549
+549
-549
```

```
for(a='A'; a<='Z'; a++)
```

Then the compilation error will be displayed on the console screen.

- **Which loop to Select?**

  Selection of a loop is always a tough task for a programmer, to select a loop do the following steps:

- Analyze the problem and check whether it requires a pre-test or a post-test loop.
- If pre-test is required, use while or for loop.
- If post-test is required, use do-while loop.

- **C program to print the characters from A to Z using while loop statement:**

```
#include<stdio.h>
int main()
{
char a = 'A';
while (a<='Z')
{
printf("%c\n", a++);
}
return 0;
}
```

```
#include <stdio.h>
int main() {
printf( "%+09d\n", 5496 );
// Output: +00005496
printf( "%09d\n", 5496 );
// Output: 000005496
return 0;
}
```

- **C program to print the characters from A to Z using do while loop statement:**

```
#include<stdio.h>
int main()
{
char a = 'A';
do
```

```
{
printf(" %c\n", a++);
} while (a<='Z');
return 0;
}
```

```
for(i=3;i<=100;i=i+3)
```

The loop starts at 3 because of the i=3 part of the for loop and increments up to 100 by using the i=i+3 formula. The for loop reads like this: "Start with i equal to 3 – and while the value of i is less than 100 – repeat the following: adding 3 to variable i each time we loop. It stops at 100 because of the i<=100 part of the loop."

- **Program 3.0**

  C program to print the given number is even or odd.

```
#include<stdio.h>
int main()
{
int a;
printf("Enter any number:");
scanf ("%d", &a);
if(a%2 == 0)
{
printf("The number is even");
}
else
{
printf("The number is odd");
}
return 0;
}
```

```
#include <stdio.h>
int main() {
int a, b;
printf( "Enter a number: " );
scanf( "%2d%d", &a, &b );
printf( "%d and %d\n", a, b);
return 0;
}

// Output:

Enter a number: 54968 # entered number

54 and 968
```

The output on the screen:

```
#include <stdio.h>
int main() {
int i = 2;
if(i > 1) {
int i;
i = 15;
printf("i: %d\n", i);
}
}

// Output: i: 15
```

```
Enter any number:
If you enter the number 4
The number is even will be outputted on the screen.
```

- **Mathematical symbol** `%` denote modulus and `(a%2 == 0)` is the **condition** and this condition imply: `a` **divided by** `2` **yields** `reminder = 0`.

For example: if you enter the number 4

Then `a = 4`

Then `4` divided by `2` yields the `remainder = 0`

Then the statement:

```
printf("The number is even");
```

is executed to print the output:

```
The number is even
```

```
#include <stdio.h>
int main() {
for(int i=1; i<10; i++)
putchar('.');
return 0;
}

// Output: .........
```

(**Note:** in C language `==` implies **equal to**)

Suppose if you enter the number 3

Then `a = 3`

Then `3` divided by `2` yields the `remainder = 1`

Then the statement

```
printf("The number is odd");
```

is executed to print the output:

```
The number is odd
```

```
#include <stdio.h>
int main() {
int i = 2;
if(i > 1) {
int i;
i = 15;
printf("i: %d\n", i); // i: 15
}
printf("i: %d\n", i); // i: 2
return 0;
}
```

Extended Backus-Naur Form:

A convention used to formally describe a programming language's syntax (i.e., indicating which programs are grammatically sound)

```
#include <stdio.h>
int main() {
extern int x, y;
printf("%d : %d", x, y); // 91 : 92
return 0;
}
int x = 91, y = 92;
```

- **Program 3.1**

  C program to print the remainder of two numbers

```c
#include<stdio.h>
int main()
{
int a, b, c;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
c = a%b;
printf("The remainder of a and b = %d", c);
return 0;
}
```

```c
#include <stdio.h>

int main() {

int i = 52;

printf("%d", i>25); // Output: 1 (True)

return 0;

}
```

```c
#include <stdio.h>

int main() {

int i = 52;

printf("%d", i<25); // Output: 0 (False)

return 0;

}
```

The output on the screen:

```
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 2
The remainder of a and b = 1 will be outputted on the screen.
```

Since (a=3 and b=2). Therefore:

3 divided by 2 (i.e., a divided by b) yields the remainder equal to 1

If the statement:

printf("The remainder of a and b = %d", c);

is replaced by the statement:

```
printf("The remainder of %d and %d = %d", a, b, c);
```

Then the output on the screen is:

Abstract syntax tree:

An intermediate depiction of the program's structure during compilation by the compiler

```
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 2
The remainder of 3 and 2 = 1 will be outputted on the screen.
```

- **Program 3.2**

  C program to check the equivalence of two numbers.

```
#include<stdio.h>
int main()
{
int x, y;
printf("Enter any number:");
scanf ("%d", &x);
printf("Enter any number:");
scanf ("%d", &y);
if(x-y==0)
{
printf("The two numbers are equivalent");
}
else
{
printf("The two numbers are not equivalent");
}
return 0;
}
```

```
#include <stdio.h>
int main() {
int a = 50;
int b = a>19 ? 450 : 150;
printf("%d", b ); // Output: 450
return 0;
}

#include <stdio.h>
int main() {
int b, a = 50;
if(a>9) b = 450;
else b = 150;
printf("%d", b ); // Output: 450
return 0;
}
```

The output on the screen:

```
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 2
The two numbers are equivalent will be outputted on the screen.
```

Since 2-2 is equal to 0 (i.e., x-y == 0).

Therefore: **the statement:**

```
{
printf("The two numbers are equivalent");
}
```

is executed to print the output:

```
The two numbers are equivalent
```

Access Modifier:

An expression in Java that specifies if or how a class member can be accessed. Public, private, protected, static, and final are examples of common access modifiers

If you enter the integers 3 and 2:

The output on the screen:

```
The two numbers are not equivalent
```

Since 3-2 is not equal to 0 (i.e., x-y != 0). Therefore: **the statement:**

```
{
printf("The two numbers are not equivalent");
}
```

is executed to print the output:

Dispatch

The method for determining which specific version of code is really executed when polymorphism is present

```
The two numbers are not equivalent
```

(As said earlier: in **C language** the symbol != implies not equal to)

- **What is the mistake in the following program?**

```
#include<stdio.h>
int main()
{
int year;
year =1996;
if(year%4==0)
printf("Leap year");
else
printf("Not a leap year");
return 0;
}
```

```
#include <stdio.h>

int main() {

float x, y;

float *i;

y = 10.56;

i = &y;

x = *i;

printf("%f", x); // Output: 10.560000

return 0;

}
```

Answer:

There is no mistake in the above program. **The output on the screen is:**

Leap year

Since year=1996. **Therefore:**

1996 divided by 4 (i.e., year divided by 4) yields the remainder equal to 0.

The **statement:**

```
printf("Leap year");
```

is executed to print the output:

Leap year

If the year is = 1995. Then

1995 divided by 4 (i.e., year divided by 4) yields the remainder not equal to 0.

The statement:

```
printf("Not a leap year");
```

is executed to print the output:

Not a leap year

Note: for a year to be leap year, year divided by 4 should yield remainder = zero.

> "An algorithm must be seen to be believed."
>
> **– Donald Knuth**

Local variables:

Variable whose existence is known only to the main program or functions are called local variables. Local variables are declared within the main program or a function.

Global variables:

Variables whose existence is known to the both **main()** as well as other functions are called global variables. Global variables are declared outside the **main()** and other functions.

```
#include<stdio.h>
int x=1; → global variable
int main()
{
int y=3; → local variable
}
```

What will be the output of the following programs?

```
#include <stdio.h>
int main () {
```

```
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
printf("Greeting message: %s\n", greeting );
return 0;
}
```

```
#include <stdio.h>
char x[50];
int main() {
x[2] = 'B';
printf("%c", x[2]); // Output: B
return 0;
}
```

```
#include <stdio.h>
#define LENGTH 10
#define WIDTH 5
#define NEWLINE '\n'

int main() {
int area;
area = LENGTH * WIDTH;
printf("Value of area : %d", area);
printf("%c", NEWLINE);
return 0;
}
```

- **Program 3.3**

  C program to print whether the given number is positive or negative

```
#include<stdio.h>
int main()
{
int  a;
a = -35;
if(a>0)
{
printf("Number is positive");
}
else
{
printf("Number is negative");
}
return 0;
}
```

```
#include <stdio.h>
int main() {
puts("Enter a character: ");
puts("To stop, press # and then Enter.");
for(;;) {
char x=getchar();
if(x=='#') {
break;
}
}
printf("......!\n");
return(0);
}
```

Output:

```
Enter a character:
To stop, press # and then Enter.
A # entered character
B # entered character
# # entered character
......!
```

The output on the screen:

```
Number is negative
```

Since `a = -35`. Therefore: **`a is less than 0`** i.e., `a < 0` because any **negative number** is always less than zero.

The statement:

```
printf("Number is negative");
```

is executed to print the output:

Number is negative

> **Language standard**
>
> ↓
>
> Rules governing what constitutes valid code in a specific language and what a compiler or interpreter must ensure for how this is carried out

- **Program 3.4**

  C program to print the sum of the first 10 digits using **`for loop statement`**

```
#include<stdio.h>
int main()
{
int i, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
printf("sum of the first 10 digits =%d", sum);
return 0;
}
```

The output on the screen:

```
#include <stdio.h>
int main() {
int x, y;
printf("Enter a number: ");
scanf("%d", &x);
printf("Enter a number: ");
scanf("%d", &y);
if(y != 0) printf("%f\n", (float)x/y);
else printf("Cannot divide by zero.\n");
return 0;
}
```

Output:

```
Enter a number: 2 # entered number
Enter a number: 0 # entered number
Cannot divide by zero.
```

```
sum of the first 10 digits = 55
```

- **How the sum of the first 10 digits = 55 is outputted on the screen through the for Loop statement?**

```
i=1 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<=10 true?
Yes, do this
sum = sum + i = 0 +1 =1
Now,
i=2 (sum = 1)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +2 =3
Now,
i=3 (sum = 3)
Is i<=10 true?
Yes, do this
sum = sum + i = 3 +3 = 6
Now,
i=4 (sum = 6)
Is i<=10 true?
Yes, do this
sum = sum + i = 6 + 4= 10
Now,
i=5 (sum = 10)
Is i<=10 true?
Yes, do this
sum = sum + i = 10 + 5= 15
Now,
i=6 (sum = 15)
Is i<=10 true?
Yes, do this
sum = sum + i = 15 + 6 = 21
Now,
i=7 (sum = 21)
Is i<=10 true?
```

```c
#include <stdio.h>
int main() {
for(int x=1; x<=5; x++)
printf("%d / 2 is: %f\n", x, (double) x /2);
return 0;
}

// Output:
                1 / 2 is: 0.500000
                2 / 2 is: 1.000000
                3 / 2 is: 1.500000
                4 / 2 is: 2.000000
                5 / 2 is: 2.500000
```

```c
#include <stdio.h>
int main() {
int x, i;
printf("Enter a number: ");
scanf("%d", &i);
x = i>0 ? i*i : -(i*i);
printf("%d × %d: %d", i, i, x);
return 0;
}
```

Output:

```
Enter a number: -5 # entered number
-5 × -5: -25
```

```
Yes, do this
sum = sum + i = 21 + 7 = 28
Now,
i=8 (sum = 28)
Is i<=10 true?
Yes, do this
sum = sum + i = 28 + 8 = 36
Now,
i=9 (sum = 36)
Is i<=10 true?
Yes, do this
sum = sum + i = 36 + 9 = 45
Now,
i=10 (sum = 45)
Is i<=10 true?
Yes, do this
sum = sum + i = 45 + 10 = 55
stops because the condition i<=10 is achieved
```

```
#include <stdio.h>
int main() {
for(int i=90; i != 55; i -= 5) {
printf("\n%d × %d, %d", i, i, i*i);
}
return 0;
}

// Output:
            90 × 90, 8100
            85 × 85, 7225
            80 × 80, 6400
            75 × 75, 5625
            70 × 70, 4900
            65 × 65, 4225
            60 × 60, 3600
```

"i" is initialized to 90 and 5 is subtracted from it each time the loop repeats. The squaring of "i" and the call to printf() are executed until "i" equals 55

The statement:

```
printf("sum of the first 10 digits =%d", sum);
```

is executed to print the output:

```
sum of the first 10 digits = 55
```

If the statement:

```
int i, sum = 0;
```

is replaced by:

```
int i, sum = 1;
```

Anti-pattern:

Typical approaches to a problem that uses rigid, rigidly structured, or inaccurate representations of coding

Then the **output on the screen** is:

```
sum of the first 10 digits = 56
```

- **What will be the output if the for loop statement `for(i =1; i<=10; i++)` is replaced bythe statement `for(i =2; i<10; i++)`?**

Answer: sum of 10 digits = 44

If the statement:

```
int i, sum, sum = 0;
```

is written instead of:

```
int i, sum = 0;
```

```
#include <stdio.h>
int main() {
int a = 15;
for(int b=10; b != a; b++) {
printf("%d\n", b);
}
return 0;
}
```

Output:

```
10
11
12
13
14
```

Then the **compilation error message** will be displayed on the screen (`stating that sum is twice declared`).

If the **`for loop`** is ended with a semicolon i.e.,

```
for( i=1; i<=10; i++);
```

Then the **compilation error** will be displayed on the console screen.

- **Program 3.5**

  C program to print the average of the first 10 numbers using for loop statement

```
#include<stdio.h>
int main()
{
```

```
int i, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
printf("sum of the first 10 numbers =%d", sum);
printf("average of the first 10 numbers =%d", avg);
return 0;
}
```

The output on the screen:

```
sum of the first 10 numbers = 55
average of the first 10 numbers = 5
```

The average of the first 10 numbers = 55/10 = 5.5 not 5. But the **output on the screen** is: average of the first 10 numbers = 5 because int is used instead of float.

If the **data type** float is used i.e.,

```
#include<stdio.h>
int main()
{
float i, avg, sum = 0;
for( i=1; i<=10; i++) sum
= sum + i;
avg = sum/10;
printf("sum of the first 10 numbers =%f", sum);
printf("average of the first 10 numbers = %f", avg);
return 0;
}
```

```
#include <stdio.h>
int main() {
int x=1;
while(x<3) {
printf("Hi..\n");
x++;
}
return(0);
}

// Output:
          Hi..
          Hi..
```

The output on the screen:

```
sum of the first 10 numbers = 55
average of the first 10 numbers = 5.5
```

- **Program 3.6**

  C program to print the product of the first 10 digits using **for loop statement**

```
#include<stdio.h>
int main()
{
int i, product = 1;
for( i=1; i<=10; i++)
product = product * i;
printf("The product of the first 10 digits =%d", product);
return 0;
}
```

The output on the screen:

```
The product of the first 10 digits = 3628800
```

- **How the product of the first 10 digits = 3628800 is outputted on the screen through the for Loop statement?**

```
i=1 (product = 1 because the product is initialized to 1 in the statement int i,
product = 1;)
Is i<=10 true?
Yes, do this
product = product *  i = 1 * 1 =1
Now,
i=2 (product = 1)
Is i<=10 true?
Yes, do this
product = product *  i = 1 * 2 = 2
Now,
i=3 (product = 2)
Is i<=10 true?
Yes, do this
product = product *  i = 2 * 3 = 6
```

```
#include <stdio.h>
int main() {
for(int x = 1; x <= 3; x++)
printf("%2d %2d *\n", x, x * x);
return 0;
}

// Output:
                    1  1 *
                    2  4 *
                    3  9 *
```

```
Now,
i=4 (product = 6)
Is i<=10 true?
Yes, do this
product = product *  i = 6 * 4 = 24
Now,
i=5 (product =24)
Is i<=10 true?
Yes, do this
product = product *  i = 24 * 5 =120
Now,
i=6 (product =120)
Is i<=10 true?
Yes, do this
product = product *  i = 120 * 6 = 720
Now,
i=7 (product =720)
Is i<=10 true?
Yes, do this
product = product *  i = 720 * 7 = 5040
Now,
i=8 (product =5040)
Is i<=10 true?
Yes, do this
product = product *  i = 5040 * 8 = 40320
Now,
i=9 (product = 40320)
Is i<=10 true?
Yes, do this
product = product *  i = 40320 * 9 = 362880
Now,
i=10 (product = 362880)
Is i<=10 true?
Yes, do this
product = product *  i = 362880 * 10 = 3628800

stops because the condition i<=10 is achieved.
```

```c
#include <stdio.h>
int main() {
char x = '\0';
printf("Enter a character: \n");
for( ; ; ) {
x = getchar(); /* get a character */
if(x == 'b')
break; /* exit the loop */
}
printf("You have entered 'b'");
return 0;
}

// Output:
Enter a character:
a # entered character
c # entered character
b # entered character
You have entered 'b'
```

```c
#include <stdio.h>
int main() {
for(double i = -2; i <= 2; i++)
printf("%e %+e %e\n", i, i, i);
return 0;
}
```

```
-2.000000e+00  -2.000000e+00 -2.000000e+00
-1.000000e+00  -1.000000e+00 -1.000000e+00
0.000000e+00  +0.000000e+00  0.000000e+00
1.000000e+00  +1.000000e+00  1.000000e+00
2.000000e+00  +2.000000e+00  2.000000e+00
```

The statement:

`printf("The product of the first 10 digits =%d", product);` is executed to display the output:

```
The product of the first 10 digits = 3628800
```

If the statement `int i, product = 1;` is replaced by:

```
int i, product = 0;
```

Then the output on the screen is:

```
The product of the first 10 digits = 0
```

If the statement:

```
for(i=1; i<=10; i++)
```

is replaced by

```
for(i=5; i<=8; i++)
```

```
#include <stdio.h>
int main() {
int a=15, b=110;
printf("Maximum of %d and %d is: %d\n", a, b, (a>b ? a : b));
return 0;
}

// Output: Maximum of 15 and 110 is: 110
```

Then the output on the screen is:

```
The product of the first 10 digits = 1680
```

- **Program 3.7**

  C Program to print the table of a number using the **`for loop`** statement

```
#include<stdio.h>
int main()
{
int n, i;
printf("Enter any number:");
scanf("%d", &n);
for( i=1; i<=5; i++)
printf("%d * %d = %d\n", n, i, n*i);
return 0;
}
```

Forwards compatibility:

Hardware or software system that is compatible with the systems' later iterations

The output on the screen:

```
Enter any number:
If you enter the number 2 (i.e., n=2)
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
will be outputted on the screen.
```

Backwards compatibility:

A hardware or software system that can successfully utilize the interfaces and data of previous system iterations

- **How the execution takes its Way through the for Loop statement**

```
Since you entered the number 2, therefore: n=2.
i=1
Is i<=5 true?
Yes, print this
2 * 1 = 2
using the statement printf("%d * %d = %d\n", n, i, n*i);

Now,
i=2
Is i<=5 true?
Yes, print this
2 * 2 = 4
using the statement printf("%d * %d = %d\n", n, i, n*i);

Now,
i=3
Is i<=5 true?
Yes, print this
2 * 3 = 6
using the statement printf("%d * %d = %d\n", n, i, n*i);

Now,
```

```
i=4
Is i<=5 true?
Yes, print this
2 * 4 = 8
using the statement printf("%d * %d = %d\n", n, i, n*i);

Now,
i=5
Is i<=5 true?
Yes, print this
2 * 5 = 10
using the statement printf("%d * %d = %d\n", n, i, n*i);

stop Now because the condition i <=5 is achieved.
```

```c
#include <stdio.h>
int main() {
 printf("%.4f\n", 549.9876545321);
 // Output: 549.9877
 printf("%3.8d\n", 5000);
 // Output: 00005000
 printf("%10.15s\n", "Alan Mathison Turing");
 // Output: Alan Mathison T
 return 0;
}
```

If the symbol * is replaced by +

i.e.,

```c
#include<stdio.h>
int main()
{
int n, a;
printf("Enter any number:");
scanf("%d", &n);
for( i=1; i<=5; i++)
printf("%d + %d = %d\n", n, i, n+ i);
return 0;
}
```

Then the output on the screen is:

```
Enter any number:
If you enter the number 2 (i.e., n=2)

2 + 1 = 3
2 + 2 = 4
2 + 3 = 5
```

Profiling:

Examining the time the code spends to find performance problems and potential improvement areas

```
2 + 4 = 6
2 + 5 = 7

will be outputted on the screen.
```

Sideways compatibility:

An assurance that code will remain to be compatible with other code

- **Program 3.8**

  C program:

  If you enter a character M

  Output must be: ch = M

```
#include<stdio.h>
int main()
{
char M;
printf("Enter any character:");
scanf("%c", &M);
printf("ch=%c", M);
return 0;
}
```

```
#include <stdio.h>
#include <string.h>
int main() {
if(strchr("albert", 'l'))
printf("l is in 'albert'\n");
// Output: l is in 'albert'
if(strstr("alan turing", "ring"))
printf("Found ring");
// Output: Found ring
return 0;
}
```

The output on the screen:

```
Enter any character:
If you enter the character M
ch = M will be outputted on the screen.
```

- **Note:**

```
getchar() function is simplified version of the scanf function
```

If we replace the statement:

```
scanf("%c", &M);
```

by the statement:

```
M = getchar();
```

```
#include <stdio.h>
int main() {
printf("%*.*f\n", 10, 2, 5496.85577);
printf("%*.*f\n", 20, 3, 5496.85577);
return 0;
}

// Output:

   5496.86
            5496.856
```

i.e.,

```
#include<stdio.h>
int main()
{
char M;
printf("Enter any character:");
M = getchar();
printf("ch=%c", M);
return 0;
}
```

There will be no change in the output on the screen i.e., **the output on the screen is:**

```
Enter any character:
If you enter the character K
ch = K will be outputted on the screen.
```

```
putchar() function is simplified version of the printf function
```

If we replace the **statement**:

```
printf("ch=%c", M);
```

```
#include <stdio.h>
int main() {
 int x = -5;
 for(int x=0; x < 5; x++)
 printf("%d ", x);
 printf("\n");
 printf("x : %d", x);
 return 0;
}

// Output:
            0 1 2 3 4
              x : -5
```

by the statement:

```
putchar (M);
```

i.e.,

```
#include<stdio.h>
```

```
int main()
{
char M;
printf("Enter any character:");
scanf("%c", &M);
putchar (M);
return 0;
}
```

Then there will be no change in the output on the screen i.e., **the output on the screen is:**

```
Enter any character:
If you enter the character M
M will be outputted on the screen.
```

If you replace the **statement:**

```
scanf("%c", &M);
```

by the statement:

```
M = getchar();
```

and the statement

```
printf("ch=%c", M);
```

by the statement:

```
putchar (M);
```

i.e.,

```
#include<stdio.h>
int main()
{
char M;
printf("Enter any character:");
M = getchar();
```

```
#include <stdio.h>
int main() {
int i = 100;
int *x, *y;
x = &i;
y = x;
/* print the value of 'i' twice */
printf("%d %d\n", *x, *y);
return 0;
}


// Output: 100 100
```

```
putchar (M);
return 0;
}
```

The output on the screen:

```
Enter any character:
If you enter the character S
S will be outputted on the screen.
```

- **Program 3.9**

  C program to print the first 5 numbers starting from one together with their squares.

```
#include<stdio.h>
int main()
{
int i;
for( i=1; i<=5; i++)
printf("number=%d its square=%d\n", i , i*i);
return 0;
}
```

The output on the screen:

```
number=1 its square=1
number=2 its square=4
number=3 its square=9
number=4 its square=16
number=5 its square=25
```

```
#include <ctype.h>
#include <stdio.h>
int main() {
printf("Enter a character: \n");
for(;;) {
char x = getchar();
if(x == '#') break;
if(isalnum(x)) printf("%c is alphanumeric\n", x);
}
return 0;
}
```

Output:

```
Enter a character:
b # entered character
b is alphanumeric
1 # entered number
1 is alphanumeric
# # entered character
```

- **How the execution takes its way through the for loop statement**

```
i=1
```

```
Is i<=5 true?
Yes, print this
number=1 its square=1
using the statement printf("number=%d its square=%d\n", i , i*i);

Now,
i=2
Is i<=5 true?
Yes, print this
number=2 its square=4
using the statement printf("number=%d its square=%d\n", i , i*i);

Now,
i=3
Is i<=5 true?
Yes, print this
number=3 its square=9
using the statement printf("number=%d its square=%d\n", i , i*i);

Now,
i=4
Is i<=5 true?
Yes, print this
number=4 its square=16
using the statement printf("number=%d its square=%d\n", i , i*i);

Now,
i=5
Is i<=5 true?
Yes, print this
number=5 its square=25
using the statement printf("number=%d its square=%d\n", i , i*i);

stop Now because the condition (i<=5) is achieved.
```

- **Note:**

If the statement

```
printf("number=%d its square=%d\n", i , i*i);
```

is replaced by the statement:

```
printf("\n number=%d/t its square=%d", i , i*i);
```

Then the output on the screen is:

```
number=1     its square=1
number=2     its  square=4
number=3     its  square=9
number=4     its square=16
number=5     its square=25
```

```
#include <stdio.h>
#include <string.h>
int main() {
char x[80];
printf("Enter a string: \n");
fgets(x, 10, stdin);
printf("%s", x);
return 0;
}
```

Output:

```
Enter a string:
alan turing # entered string
alan turi # output
```

`tab "/t"` is included because to leave space between

number=1 and its square=1

Suppose

```
printf("number=%d its square=%d", a , a*a);
```

is replaced by the statement:

```
printf("number=%d\n its square=%d\n", a , a*a);
```

The output on the screen is:

```
number=1
its square=1
number=2
its square=4
number=3
its square=9
number=4
its square=16
number=5
its square=25
```

```c
#include <stdio.h>
int main() {
int i, *x, **y;
i = 55;
x = &i;
y = &x;
/* print the value of i */
printf("%d", **y);
// Output: 10
return 0;
}
```

If you replace the **printf** statement:

```c
printf("number=%d its square=%d", a , a*a);
```

by the statement:

```c
printf("number=%d\n, its square=%d\n", a , a*a);
```

i.e., if you place variable separator (i.e., comma) between **number=%d\n** and **its square=%d\n**
Then the compilation error will be displayed on the screen.

- **Write a program to print the first 10 numbers starting from one together with their squares and cubes?**

Answer:

```c
#include<stdio.h>
int main()
```

```
{
int i;
for( i=1; i<=10; i++)
printf("number=%d its square=%d its cube=%d\n", i , i*i, i*i*i);
return 0;
}
```

Rules for switch statement:

- A switch is a decision making construct in 'C.'
- A switch is used in a program where multiple decisions are involved.
- A switch must contain an executable test-expression.
- Each case must include a break keyword.
- Case label must be constants and unique.
- The default is optional.

no-op:

Code that does nothing

Pointer

A variable that can hold the address of other variables, arrays, structures, unions and functions that are used in the C program. It contains only the memory location of the variable rather than its content. The purpose of pointer is to save memory space and achieve faster execution time.

Advantages:

- Dynamic memory allocation.
- More compact and efficient coding.
- To return multiple values via functions.
- To point to different data structures.

Breakpoint

(Break execution)

A directive instructing the debugger to halt and wait for user input

Operators used with pointers

There are 2 basic operators used with pointers:

- The address operator: & (ampersand)
- The indirection operator: * (asterisk)

The **address operator** gives the address of a variable while the **indirection operator** gives the value of the variable that the pointer is pointing to.

Suppose x is a variable that holds data of type int. We can access the address of x through an address operator [&]. This address can be stored in some variable [say y]. Thus we can relate x and y as:

"y = &x"

y is called a pointer variable because it holds the address of another variable [i.e., x] but not the actual value of x.

There are two integer variables 100 and 640 stored respectively at memory locations 1000 and 1003. Let us assign:

x = 100; y = 640;

In order to access the value of a variable 'x' via pointer, we must need a variable.

Assume that variable be px. The following statement assigns the address of 'x' to px:

px= &x = 1000

In order to access the value of a variable 'y' via pointer, we must need a variable.

Assume that variable be py. The following statement assigns the address of 'y' to py:

py= &y = 1003

x, y, px and py are declared in the program as:

int x, y, *px, *py;

x and y are integer variables. The asterisk that appears before px and py indicates that the variables are pointer variables.

```c
#include <stdio.h>
int main() {
 struct {
 int x;
 int y;
 } a, b;

 a.x = 56;
 b = a; /* assign one structure to another */
 printf("%d", b.x);
 return 0;
}

// Output: 56
```

```c
#include <stdio.h>
int main() {
int x = 5;

char names[][12]={
 "Albert",
 "Alan",
 "John",
 "James",
 "Mary",
 "David"
};

printf("%s", names[x]);
// Output: David
return 0;
}
```

- **Program 4.0**

  C program to print the sum of two numbers using pointers

If we create an **integer variable** x by declaring the statement:

```
int x;
```

within the body of the **main** function `int main()` – this variable is stored in the computer memory i.e., this variable occupies a specific location in the space of **computer memory**. And this integer variable x is assigned an address (`i.e., &x`) to locate its position in the computer memory (`like a house in the street is assigned an address to locate its position in the street`). **`Pointers`** are the variables that represent the address of x in the computer memory i.e., `p = &x`, where `&x` imply the **`address of x`** in the computer memory and **`p`** is the pointer variable (`which is the variable that represent the address of x in the computer memory`). And further if you assign a value to the **`variable x`** by declaring the statement:

```
x=1;
```

within the body of the main function — this value is stored in the address of x in the computer memory. "*" denote pointer operator and *p denote the pointer (which represent the value stored in the **address of x** in the computer memory).

- **C program to print the address of x and the value assigned to x**

```c
#include <stdio.h>
int main()
{
int x, *p;
x = 1;
p = &x;
printf("The address of the variable x =%d", p);
printf("The value of the variable x =%d", *p);
return 0;
}
```

C++ Program:

```cpp
#include<iostream>
using namespace std;
int main() {
int x, y, z, sum;
for ( ; ; ) {
cout << "Enter x, y and z: "<< endl;
cin >> x >> y >> z;
if(!cin)
break;
sum = x + y + z;
cout << "x + y + z = " << sum << endl;
}
cout <<".......\n";
}

// Output:

Enter x, y and z:
2 # entered number
4 # entered number
5 # entered number
x + y + z = 11
Enter x, y and z:
! # entered character
.......
```

The output on the screen:

```
The address of the variable x = 0x7fffc60478a4

The value of the variable x = 1
```

Since p = &x:

```
*p= *&x
```

The value of the **variable** x = 1 because you have assigned a value to the variable x by declaringthe statement:

```
x=1;
```

within the body of the main function.

Minimum working example:

A code snippet that has no redundant information and can be copied and pasted into an empty Lean file while retaining all of its functionality (minimal)

If the statements:

```
printf("The address of the variable x =%d", p);
printf("The value of the variable x =%d", *p);
```

are replaced by the statement:

```
printf("The address of the variable x =%d and its value =%d", p,*p);
```

i.e.,

Code path:

The path used by the program's source code when it runs

```
#include <stdio.h>
int main()
{
int x, *p;
x=1;
p = &x;
printf("The address of the variable x =%d and its value =%d", p,*p);
return 0;
}
```

Then the **output on the screen** is:

```
The address of the variable x = 0x7fffc60478a4 and its value = 1
```

```
#include <stdio.h>
int main()
{
int x, y, *p, *q, sum;
printf("Enter any number:");
```

```
scanf("%d", &x);
printf("Enter any number:");
scanf("%d", &y);
p = &x;
q = &y;
sum = *p + *q;
printf("Sum of entered numbers = %d\n", sum);
return 0;
}
```

```
#include <stdio.h>
#define MAX 55
int main() {
#if MAX>85
 printf("55<85\n");
#else
 printf("85>55\n");
#endif
 return 0;
}

// Output:

            85>55
```

The output on the screen:

```
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3
Sum of entered numbers = 5 will be outputted on the screen.
```

Since `pointer *p` imply the value assigned to the variable `x (i.e., 2)` through the keyboard and the `pointer *q` imply the value assigned to the variable `y (i.e., 3)` **through the keyboard**. Therefore: `sum = *p + *q = 2 + 3 = 5` (**which will be outputted on the screen**).

- **C program to print the product, subtraction and division of two numbers using pointers**

```
#include <stdio.h>
int main()
{
int x, y, *p, *q, product, subtract, div;
printf("Enter any number:");
scanf("%d", &x);
printf("Enter any number:");
scanf("%d", &y);
```

MoSCoW method:

A technique for prioritizing where criteria are categorized into elements that frequently occur in a certain iteration of a piece of code and are thus regarded as such include elements that Must, Should, Could, and Won't be done

```
p = &x;
q = &y;
product = *p * *q;
subtract = *p - *q;
div= *p / *q;
printf("product of entered numbers = %d\n", product);
printf("subtract of entered numbers = %d\n", subtract);
printf("division of entered numbers = %d\n", div);
return 0;
}
```

```
#include <stdio.h>
int main() {
int i, x = 1;
for (i = 1; i <= 5; i++)
printf("%d", i & x);
// Output: 10101
return 0;
}
```

The output on the screen:

```
Enter any number:

If you enter the number 4

Enter any number:

If you enter the number 2

product of entered numbers = 8

subtract of entered numbers = 2

division of entered numbers = 2

will be outputted on the screen.
```

```
#include <stdio.h>
int main() {
int i, x = 1;
for (i = 1; i <= 5; i++)
putchar(((i & x) == 0) ? '0': '1');
// Output: 10101
return 0;
}
```

- **C program to find the greatest of two numbers using pointers**

```
#include<stdio.h>
int main()
{
int x, y, *p, *q;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
p = &x;
q = &y;
```

```
#include <stdio.h>
#define mkstr(x) # x
int main() {
printf(mkstr(Albert));
// Output: Albert
return 0;
}
```

```
if(*p>*q)
{
printf("x is greater than y");
}
if(*q>*p)
{
printf("y is greater than x");
}
return 0;
}
```

```
#include <ctype.h>
#include <stdio.h>
int main() {
printf("Enter a character: \n");
for(;;) {
char x = getchar();
if(x == '#') break;
if(isdigit(x)) printf("%c is a digit\n", x);
}
return 0;
}
```

Output:

```
Enter a character:
J # entered character
2 # entered number
2 is a digit
# # entered character
```

The output on the screen:

```
Enter any integer:
If you enter the integer 10
Enter any integer:
If you enter the integer 16
y is greater than x will be outputted on the screen.
```

- **What is the output of the following programs?**

i)

```
#include <stdio.h>
int main() {
printf("Alan %c %s", 'M', "Turing");
return 0;
}
```

```
#include <ctype.h>
#include <stdio.h>
int main() {
printf("Enter a character: \n");
for(;;) {
char x = getchar();
if(x == '#') break;
if(isupper(x)) printf("%c is uppercase\n", x);
}
return 0;
}
```

Output:

```
Enter a character:
j # entered character
A # entered character
A is uppercase
# # entered character
```

Answer:

Alan M Turing

ii)

```c
#include <stdio.h>
int main()
{
int x, t, c;
x =12;
t = 2;
c = x/t;
printf("velocity = %d m/s", c);
return 0;
}
```

```c
#include <stdio.h>
#include <string.h>
int main() {
char *x;
x = strchr("This is my book.", ' ');
printf("%s", x);
return 0;
}

// Output: is my book.
```

Answer:

velocity = 6 m/s

- **Program 4.1**

C program to print the sum of two numbers using functions

```c
#include<stdio.h>
int addition();
int main()
{
int answer;
answer = addition();
printf("The sum of two numbers is: %d\n", answer);
return 0;
}

int addition()
{
int x, y;
printf("Enter any integer:");
```

Input sanitation

↓

Eliminate invalid characters from input

```
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
return x+y;
}
```

The output on the screen:

```
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
The sum of two numbers = 8 will be displayed on the screen.
```

C Program:

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
   int a=6, b=2;
   printf("%d", a+(~b)+1);
   return 0;
}
```

Output:

4

Java Program:

```
public class HelloWorld {
public static void main(String[] args) {
int x = 0;
while (true) {
System.out.println("\n " + x++);
if (x == 4)
break;
}
}
}
```

Output:

0

1

2

3

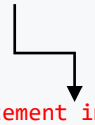
```
public class HelloWorld {
public static void main(String[] args) {
while(true) {
 System.out.println("infinite while loop");
}
}
}
```

Output:

infinite while loop
infinite while loop
infinite while loop
infinite while loop
infinite while loop .....

Cyber Analytics: An approach for identifying potential vulnerabilities and minimizing harm from existing cyberthreats that combines concepts of cybersecurity and data science

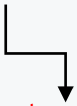
```
int addition();
```



The statement implies function declaration

int means integer and int addition() implies: addition() should return integer value.

```
int addition()
```

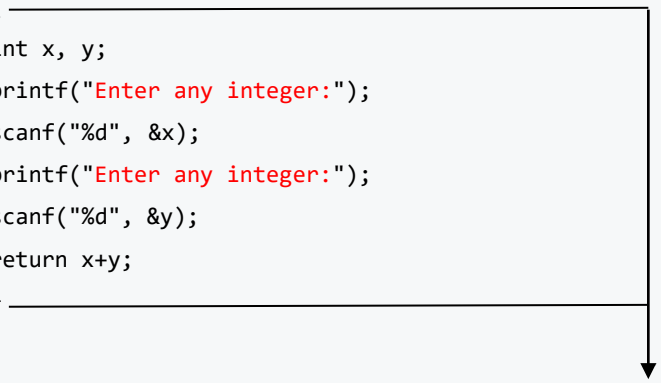


The function to add the entered values (i.e., 3 and 5) and return the result (i.e., 3 + 5 i.e., 8) to the statement:

printf("The sum of two numbers = %d", answer); to make provision to display the output:

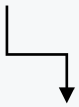
The sum of two numbers = 8

```
{
int x, y;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
return x+y;
}
```



The body of the function int addition()

```
answer = addition();
```



The function call i.e., this statement calls the function:

addition()

to add the entered values (i.e., 3 and 5) and return the result (i.e., 3 + 5 i.e., 8) to the statement:

```
printf("The sum of two numbers = %d", answer);
```

to make provision to display the output:

```
The sum of two numbers = 8
```

on the screen.

In the statement:

```
printf("The sum of two numbers=%d", answer);
```

The format string "%d" indicates that the value to be displayed at that point in thestring i.e., after the statement:

```
The sum of two numbers =
```

needs to be taken from the result returned by the function int addition().

- **C program to print the product of two numbers using functions**

```
#include<stdio.h>
int multiplication();
int main()
{
int answer;
answer = multiplication();
printf("The product of two numbers is: %d\n", answer);
return 0;
}

int multiplication()
{
int x, y;
```

```
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
return x*y;
}
```

```
#include <math.h>
#include <stdio.h>
int main() {
printf("%1.1f %1.1f", fabs(2.0), fabs(-2.0));
return 0;
}

// Output: 2.0 2.0
```

The output on the screen:

```
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
The product of two numbers = 15 will be outputted on the screen.
```

- **C program to print the greatest of two numbers using functions**

```
#include<stdio.h>
int largest();
int main()
{
int answer;
answer = largest();
printf("The largest of two numbers is: %d\n", answer);
return 0;
}

int largest()
{
int x, y;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
```

```
#include <stdio.h>
#define PUT(x) printf("Albert\n");
int main() {
PUT(Hi from main());
return 0;
}

// Output: Albert
```

```
if(x>y)
return x;
if(y>x)
return y;
}
```

```
#include <stdio.h>
int main() {
char x[] = "Ilbert";
char *z = x;
*z = 'A';
printf("%s\n", x);
}

// Output: Albert
```

The output on the screen:

```
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
The largest of two numbers= 5 will be outputted on the screen.
```

- **C program to print the greatest of three numbers using functions**

```
#include <stdio.h>
#define NAMES(x, y, z) printf(#x ", " #y ", "  #z " and Mary!\n")
int main() {
NAMES(Albert, John, James);
return 0;
} // Output: Albert, John, James and Mary!
```

```
#include<stdio.h>
int largest();
int main()
{

int answer;
answer = largest();
printf("Largest of three numbers=%d", answer);
return 0;
}
int largest()
{
int x, y, z;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
printf("Enter any integer:");
scanf("%d", &z);
```

```
#include <stdio.h>
int main() {
printf("Alan " "Mathison "  "Turing");
// Output: Alan Mathison Turing
return 0;
}
```

```
if(x>y&& x>z)
return x;
if(y>x&& y > z)
return y;
if(z>x && z>y)
return z;
}
```

```
#include <math.h>
#include <stdio.h>
int main() {
int e;
double x;
x = frexp(10.0, &e);
printf("%f %d", x, e);
}

// Output: 0.625000 4
```

$x = Mantissa \times 2^{Exponent}$

$x = 0.625000 \times 2^{4}$

$x = 10.0$

Exponent

Mantissa

The output on the screen:

```
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
Enter any integer:
If you enter the integer 10
Largest of three numbers = 10 will be outputted on the screen.
```

- **C program to print the square of the number using functions**

```
#include<stdio.h>
int  square();
int main()
{

int  answer;
answer = square();

printf("Square of the number=%d", answer);
}
int square()
{
int x;
```

```
#include <stdio.h>
int main() {
printf("Albert \"Einstein\"!\n");
// Output: Albert "Einstein"!
return 0;
}
```

```
printf("Enter any integer:");
scanf("%d", &x);
return x*x;
}
```

```
#include <stdio.h>
int main() {
char *x = "alan", *y = "alan";
if (x == y)
printf("The 2 strings have the same address\n");
else
printf("The 2 strings have the different address\n");
return 0;
}
// Output: The 2 strings have the same address
```

The output on the screen is:

```
Enter any integer:
If you enter an integer 5
Square of the number = 25 will be outputted on the screen.
```

- **What is the output of the following program?**

```
#include<stdio.h>
int main()
{
int x=6;
printf("The address of x = %d", &x);
return 0;
}
```

```
#include <stdio.h>
int main() {
double i;
for(i=2.0; i<1.0e+10; i=i*10)
printf("%g ", i);
return 0;
}
```

Answer:

```
The address of x = 343441332
```

- **Program 4.2**

```
# Output:
      2 20 200 2000 20000 200000 2e+06 2e+07 2e+08 2e+09
```

If-else statement provides a way for selecting any one of the 2 possible alternatives. And, **nested- if** allow us to select one of the many alternatives but it is time consuming. To overcome this, the switch case statement is used. **Switch (case)** allows the user to make decision from the number of choices i.e., **from the number of cases.**

For example:

```c
#include<stdio.h>
int main()
{
char ch;
printf("Enter any character:");
scanf("%c", &ch);
switch(ch)
{
case 'R':
printf("Red");
break;
case 'W':
printf("White");
break;
case 'Y':
printf("Yellow");
break;
case 'G':
printf("Green");
break;
default:
printf("Error");
break;
} ——————————→ End of switch
return 0;
} ——————————→ End of main()
```

Python code:

```python
import numpy as np
a = [15,16,17]
b = np.array(a)
print("1 Dimensional array: ", b)
```

Output:

1 Dimensional array: [15 16 17]

```python
import numpy as np
a = [[15,16,17], [18,19,20]]
b = np.array(a)
print("2 Dimensional array: ",b)
```

Output:

2 Dimensional array: [[15 16 17]
[18 19 20]]

```python
import numpy as np
a = [[[15,16,17],[18,19,20],[21,22,23]]]
b = np.array(a)
print("3 Dimensional array: ",b)
```

Output:

3 Dimensional array: [[[15 16 17]
[18 19 20]
[21 22 23]]]

The output on the screen:

```
Enter any character:
If you enter a character R
Red will be outputted on the screen.
```

`switch(ch)` allow us to make decision from the number of choices i.e., from the number of cases

```
case 'R':
case 'W':
case 'Y':
case 'G':
```

Since we have entered the **`character`** `R` (`which corresponds to case 'R':`)

The statement

```
printf("Red");
```

is executed to display the output:

`Red`

on the screen.

Suppose you enter a **`character`** `K`

Then the output on the screen is:

```
Error
```

(`Entered character K does not correspond to any of the cases:`

```
case 'R':
case 'W':
case 'Y':
case 'G':
```

Therefore the statement:

```
printf("Error");
```

```
import numpy as np
a = np.array([11, 12, 13, 14], ndmin=5)
print(a)
print('Dimensions of array:', a.ndim)
```

Output:

[[[[[11 12 13 14]]]]]
Dimensions of array: 5

is executed to display the output:

```
Error
```

on the screen).

The `break` statement denotes the end of a particular case and thereby the switch statement is terminated. The case `default` is executed, when the value of an expression is not matched with any of the cases.

If the statements:

```
case 'R':
printf("Red");
break;
case 'W':
printf("White");
break;
case 'Y':
printf("Yellow");
break;
case 'G':
printf("Green");
break;
default:
printf("Error");
break;
```

are replaced by the statements:

```
case 'R':
printf("Red");
case 'W':
printf("White");
case 'Y':
printf("Yellow");
break;
case 'G':
printf("Green");
break;
```

Coffman's conditions that cause a deadlock:

- **Mutual Exclusion:** A crucial resource can only be used by one process at a time
- **Hold and Wait:** Some resources may be allocated to a process while others are being waited for
- **No Pre-emption:** No resource can be forcibly taken from a process that is holding it
- **Circular Wait:** Each process in a closed chain of processes retains at least one resource required by another process in the chain

```
#include <stdio.h>
int main() {
    float x=0.2d;
    if(x>0.9) {
    printf("Albert");
    }
    else {
    printf("Einstein");
    }
    return 0;
}
```

Output:

Einstein

```
#include <stdio.h>
int main() {
    float x=0.2d;
    if(x<0.9) {
    printf("Albert");
    }
    else {
    printf("Einstein");
    }
    return 0;
}
```

Output:

Albert

```
default:
printf("Error");
break;
```

Terraform

An open source infrastructure orchestration tool that automates the deployment and management of infrastructure

Then the **output on the screen** is:

```
Red
White
Yellow
```

i.e., the output will be printed till **`yellow`** even though you have entered the **`character R.`**

Arrays

Ordered list of homogeneous data elements

```
from copy import copy
a = [1, 2, [3, 5], 4]
b = copy(a)
b[3] = 7
b[2].append(6)
print(a)
print(b)
```

Output:

[1, 2, [3, 5, 6], 4]
[1, 2, [3, 5, 6], 7]

- Elements may be of data type int, float, char or double.
- All the elements are stored in consecutive memory locations [on RAM].

In an array named odd, the individual items are as shown below:

odd[1]	odd[2]	odd[3]	odd[4]	odd[5]

where:

1, 2, 3, 4 and 5 are subscripts

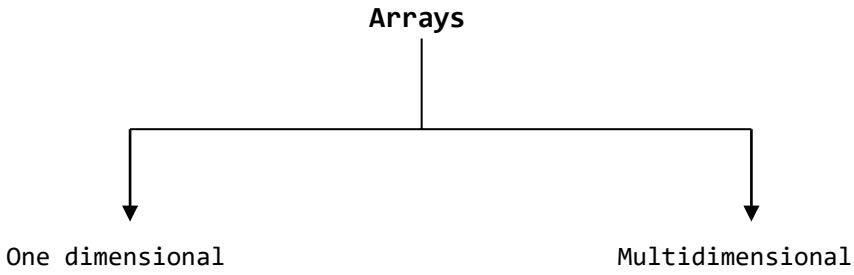
odd[1] : the first element in an array odd.
odd[2] : the second element in an array odd.
odd[3] : the third element in an array odd.
odd[4] : the fourth element in an array odd.
odd[5] : the fifth element in an array odd.

In general, odd[i] denote the i^{th} element of an odd array.

```
from copy import deepcopy
a = [1, 2, [3, 6], 5]
b = deepcopy(a)
b[3] = 9
b[2].append(7)
print(a)
print(b)
```

Output:

[1, 2, [3, 6], 5]
[1, 2, [3, 6, 7], 9]



- **One dimensional array:** Number of subscripts is 1
- **Multidimensional array:** Number of subscripts is more than 1

```
array1[] → one-dimensional array

array2[] []  → two-dimensional array

array3[] [] [] → three-dimensional array, and so on.
```

- **Program 4.3**

  C program to print the output:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
```

```
#include <math.h>
#include <stdio.h>
int main() {
printf("%f", ldexp(3,4));
return 0;
}

// Output: 48.000000
```

ldexp(3,4) = 3 × 2^4 = 3 × 16

ldexp(3,4) = 48.000000

using arrays:

```
#include<stdio.h>
int main()
{
```

```
int i;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
printf("\n Element [%d] = %d", i, num[i]);
return 0;
}
```

```
#include <stdlib.h>
#include <stdio.h>
int main() {
for(int x=1; x<=5; x++)
// Display 5 random numbers
printf("%d\n", rand());
return 0;
}
```

The output on the screen:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
```

The statement:

```
int num [5] = {16, 18, 20, 25, 36};
```

imply that we are creating an integer array (`and the name of array is num`) consisting of 5 values(`i.e., 16, 18, 20, 25, 36`) of the same **`data type int`**.

The number of values between the braces `{ }` cannot be larger than the number of values that wedeclare for the array between square brackets `[ ]`.

There are `5` integers i.e., `16, 18, 20, 25, 36` within the braces { }, so `5` is written within the squarebrackets [ ].

If there were 6 integers i.e., `16, 18, 20, 25, 36, 42` within the braces `{ }`, then `6` must be written within the square brackets [ ].

- **Note:** With the declaration `int num [5]`, **computer** creates `5` memory cells with name `num[0], num[1], num[2], num[3], num[4]`.

And since:

```
int num [5] = {16, 18, 20, 25, 36};
```

the values `16, 18, 20, 25, 36` are stored in `num[0], num[1], num[2], num[3], num[4]` respectively.

- **How the execution takes its way through the `for loop` statement?**

```
i=0
Is i<5 true?
Yes, print this
Element [0] = 16
using the statement:
                        printf("\n Element [%d] = %d", i, num[i])
format string %d in the square brackets indicates  that the value to be displayed at that point in the string i.e., within the square brackets [ ] needs to be taken from a variable (which is i i.e., i=0) and the format  string %d after the statement (\n Element [%d] = ) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] = ) needs to be taken from a variable (which is stored in num[i] i.e., num[0] i.e., 16).

Now,
i=1
Is i<5 true?
Yes, print this
Element [1] = 18
using the statement:
                        printf("\n Element [%d] = %d", i, num[i])
format string %d in the square brackets indicates that the value to be displayed at that point in the string i.e., within the square brackets [ ] needs to be taken from a variable (which is i i.e., i=1) and the format  string %d after the statement (\n Element [%d] = ) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] = ) needs to be taken from a variable (which is stored in num[i] i.e., num[1] i.e., 18).

Now,
i=2
```

```
Is i<5 true?
Yes, print this
Element [2] = 20
using the statement:
                        printf("\n Element [%d] = %d", i, num[i])
format string %d in the square brackets indicates  that the value to be displayed at that point in the string i.e., within the square brackets [ ] needs to be taken from a variable (which is i i.e., i=2) and the format  string %d after the statement (\n Element [%d] = ) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] = ) needs to be taken from a variable (which is stored in num[i] i.e., num[2] i.e., 20).

Now,
i=3
Is i<5 true?
Yes, print this
Element [3] = 25
using the statement:
                        printf("\n Element [%d] = %d", i, num[i])
format string %d in the square brackets indicates  that the value to be displayed at that point in the string i.e., within the square brackets [ ] needs to be taken from a variable (which is i i.e., i=3) and the format  string %d after the statement (\n Element [%d] = ) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] = ) needs to be taken from a variable (which is stored in num[i] i.e., num[3] i.e., 25).

Now,
i=4
Is i<5 true?
Yes, print this
Element [4] = 36
using the statement:
                        printf("\n Element [%d] = %d", i, num[i])
Stop because the condition i<5 is achieved.
```

Format string %d in the square brackets indicates that the value to be displayed at that point in the string i.e., with the square brackets [ ] needs to be taken from a variable (which is i i.e., i=4)

and the format string %d after the statement `(\n Element [%d] = )` indicates that the value to be displayed at that point in the string i.e., after the statement `(\n Element [%d] = )` needs to be taken from a variable (`which is stored in num[i] i.e., num[4] i.e., 36`).

Suppose the statement:

```
printf("\n Element [%d] = %d", i, num[i]);
```

is replaced by the statement:

```
printf("\n Element [%d] = %d", i, num[0]);
```

Then the **output on the screen**:

```
Element [0] = 16
Element [1] = 16
Element [2] = 16
Element [3] = 16
Element [4] = 16
```

```c
#include <stdio.h>
/* myfunc() function */
void myfunc() {
printf("Bill\n");
}
/* Program starts here */
int main() {
printf("Albert\n");
printf("John\n");
myfunc();
printf("Mary\n");
return(0);
}

// Output:
            Albert
             John
             Bill
             Mary
```

Suppose the statement:

```
printf("\n Element [%d] = %d", i, num[i]);
```

is replaced by the statement:

```
printf("\n Element [%d] = %d", i, num[1]);
```

The output on the screen:

```
Element [0] = 18
```

```c
#include<stdio.h>
int main() {
int x = 0, y = 0, z;
z = 0 && (x = y = 555);
printf("%d %d %d\n", x, y, z);
// Output: 0 0 0
z = 555 || (x = ++y);
printf("%d %d %d\n", x, y, z);
// Output: 0 0 1
return 0;
}
```

```
Element [1] = 18
Element [2] = 18
Element [3] = 18
Element [4] = 18
```

```c
#include <stdio.h>
int main() {
printf("A%cl%ca%cn\n", '\0',  '\0', '\0') ;
return 0;
}

// Output: Alan
```

Suppose the statement:

```
printf("\n Element [%d] = %d", i, num[i]);
```

is replaced by the statement:

```
printf("\n Element [%d] = %d", i, num[2]);
```

The output on the screen:

```
Element [0] = 20
Element [1] = 20
Element [2] = 20
Element [3] = 20
Element [4] = 20
```

Enter 0 after typing it. The computer asks once more. Input 101 and hit Enter. The computer asks once more. Any value outside of the range of 1 to 100 prompts the program to repeatedly prompt the user to enter the correct number.

```c
#include <stdio.h>
int main() {
int x;
do {
printf("Enter a number: \n");
scanf("%d", &x);
}
while(x<1 || x>100);
return 0;
}
```

```
Enter a number:
0 # entered number
Enter a number:
101 # entered number
Enter a number:
2 # entered number
```

Suppose the statement:

```
printf("\n Element [%d] = %d", i, num[i]);
```

is replaced by the statement:

```
printf("\n Element [%d] = %d", i, num[3]);
```

The output on the screen:

```
Element [0] = 25
```

```
Element [1] = 25
Element [2] = 25
Element [3] = 25
Element [4] = 25
```

Suppose the statement:

```
printf("\n Element [%d] = %d", i, num[i]);
```

is replaced by the statement:

```
printf("\n Element [%d] = %d", i, num[4]);
```

The output on the screen:

```
Element [0] = 36
Element [1] = 36
Element [2] = 36
Element [3] = 36
Element [4] = 36
```

```
#include <stdio.h>
int main() {
int x;
for (x = 'a'; x <= 'z'; x++)
/* a b c d ...z is printed */
printf("%c ", x);
return 0;
}
```

If the condition:

i<5

is replaced by the condition:

i<=5

Then the **output on the screen** is:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
Element [5] = 3656
```

```
#include <stdio.h>
int main() {
char x;
for (x = 65; x <= 90; x++)
/* A B C D ...Z is printed */
printf("%c ", x);
return 0;
}
```

- **3656** is the number stored in the memory i.e., any number stored in the memory will be displayed.

If the statement:

`int num [5] = {16, 18, 20, 25, 36};` is replaced by the statement:

```
int num [i] = {16, 18, 20, 25, 36};
```

Then the **`compilation error`** will be displayed on the screen because there are `5 elements` within the braces `{}` not i elements.

- **Note:**
  - **C program to print the sum of the elements in array.**

```
#include<stdio.h>

int main()

{

int i, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
printf("Sum of the Elements in the array = %d", sum);
return 0;
}
```

```
#include <stdio.h>
int main() {
int x=2, y=1, z=3;
x += y += z;
printf("x = %d, y = %d, z = %d\n", x, y, z);
// Output: x = 6, y = 4, z = 3
return 0;
}
```

The output on the screen:

```
Sum of the Elements in the array = 115
```

i.e., 16 + 18 + 20 + 25 + 36 = 115

- **How the Execution takes its way through the for loop statement?**

```
i=0 (sum = 0)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[0] = 0 +16 =16
```

Now,
i=1 (sum = 16)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[1] = 16 +18 =34

Now,
i=2 (sum = 34)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[2] = 34 +20 =54

Now,
i=3 (sum = 54)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[3] = 54 +25 =79

Now,
i=5 (sum = 79)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[5] = 79 + 36 =115
stop because the condition i<5 is achieved

```
#include <stdio.h>
int main() {
int a, b, c;
int x=0;
if(x==0) {
a = 12;
b = 13;
c = 15;
}
else {
a = -11;
b = -12;
c = -13;
}
printf("%d", a + b + c);
return 0;
}

// Output: 40
```

The statement:
printf("Sum of the Elements in the array = %d", sum); is executed to display the output:
Sum of the Elements in the array = 115
on the screen.

If the statement:

```
int i, sum = 0;
```

is replaced by int i, sum = 1;
Then The output on the screen:

```
Sum of the Elements in the array = 116
```

- **C program to print the average of the elements in array**

```
#include<stdio.h>
int main()
{
int i, avg, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num [i];
avg = sum/5;
printf("Sum of the Elements in the array = %d", sum);
printf("average of the elements in the array= %d", avg);
return 0;
}
```

The output on the screen:

```
Sum of the Elements in the array = 115
average of the elements in the array = 23
```

Write a program to print:

```
Element [0] = E
Element [1] = I
Element [2] = N
Element [3] = S
Element [4] = T
Element [5] = E
Element [6] = I
Element [7] = N
```

```
#include <stdio.h>
int main() {
int x = 20+/* add the 2 numbers */50;
printf("%d", x);
// Output: 70
return 0;
}
```

using arrays

Answer:

```
#include<stdio.h>
int main()
{
int i;
char name [8] = {' E' , ' I', ' N', ' S', ' T ', ' E', ' I', ' N'};
for(i=0; i<8; i++)
printf("\n Element [%d] = %c", i, name[i]);
return 0;
}
```

- **Note:**

If the format string %d is used instead of %c i.e., if the statement:

```
printf("\n Element [%d] = %c", name[i], name[i]);
```

is written instead of the statement:

```
printf("\n Element [%c] = %c", name[i], name[i]);
```

Then the **output on the screen** is:

```
Element [69] = E
Element [73] = I
Element [78] = N
Element [83] = S
Element [84] = T
Element [69] = E
Element [73] = I
Element [78] = N
```

```
#include <stdio.h>
int main() {
int i = 0, x = 16;
while ((x /= 2) > 0)
printf("%-6d", x);
return 0;
}

// Output:
          8     4     2     1
```

- **What will be the output of the following programs?**

i)

```c
#include <stdio.h>
#include <math.h>
int main()
{
printf("%f", cbrt(27));
return 0;
}
```

Answer:

```
3.000
```

ii)

```c
#include <stdio.h>
int main()
{
char i;
char body [4] = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
printf("\n body[%c] = %c", body[i] , body[i]);
return 0;
}
```

Answer:

```
body [b] = b
body [o] = o
body [d] = d
body [y] = y
```

iii)

```c
#include <stdio.h>
int main() {
printf("\"Albert\"");
return 0;
}

// Output: "Albert"


#include <stdio.h>
int main() {
printf("%cAlbert%c", '\'', '\'');
return 0;
}

// Output: 'Albert'


#include <stdio.h>
int main() {
printf("%cAlbert%c", '\"', '\"');
return 0;
}

// Output: "Albert"

#include <stdio.h>
int main() {
printf("\'Albert'");
return 0;
}

// Output: 'Albert'
```

```c
#include <malloc.h>
int main()
{
int x=2;
printf("%d", malloc(200*sizeof(x)));
return 0;
}
```

C++ program:

```cpp
#include <iostream>
using namespace std;
int main(int argc, char **argv) {
    cout << 16u - 2;
    return 0;
}
```

Output:

14

Answer: 8183824

Law of Increasing Complexity:

An evolving software program will become more complex as it is continuously altered.

- **What is the mistake in the following program:**

```cpp
#include <iostream>
using namespace std;
int main(int argc, const char * argv[]) {
    int x[] = {11, 12, 13, 14, 15, 16};
    cout << (1 + 2)[x] - x[0] + (x + 2)[3];
    return 0;
}
```

Output:

19

```c
#include<stdio.h>
int main()
{
int i;
int num [] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
printf("\n Element [%d] = %d", i, num[i]);
return 0;
}
```

Answer: There is no mistake in the above program. The **output on the screen** is:

Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36

Java Program:

```java
public class MyClass {
public static void main(String[] args) {
String x = "Albert";
for(int i = x.length()-1; i >= 0; i--) {
System.out.print(x.charAt(i));
}
}
} // Output: treblA
```

Law of Continuing Change

↓

Software programs are never truly finished. Programs must be updated as requirements change

- **Program 4.3**

  C program to print the output:

```
#include <iostream>
using namespace std;
int main(){
    extern int x;
    cout<< x <<endl;
    return 0;
} int x = 55;
```

Output:

55

Name of the book = B

Price of the book = 135.00

Number of pages = 300

Edition of the book = 8

using structures

```
#include<stdio.h>
int main()
{
struct book {
char name;
float price;
int pages;
int edition;
};

struct book b1;
b1.name = 'B';
b1.price = 135.00;
b1.pages = 300;
b1.edition = 8;
printf("\n Name of the book = %c", b1.name);
printf("\n Price of the book = %f", b1.price);
printf("\n Number of pages = %d", b1.pages);
printf("\n Edition of the book = %d", b1.edition);
return 0;
}
```

```
#include <iostream>
using namespace std;
void myfunc(int& x, int& y, int& z) {
    x *=6;
    y *=4;
    z *=3;
}
int main() {
    int p = 11, q = 12, r = 15;
    myfunc(p, q, r);
    cout<<" "<<p<<"\n "<<q<<"\n "<<r;
    return 0;
    }
```

Output:

66

48

45

The output on the screen:

```
#include <iostream>
using namespace std;
int main() {
    int x=11, y=13, z;
    z=(x>y) ? x : y;
    cout<<z;
    return 0;
}
```

Output:

13

```
Name of the book = B
Price of the book = 135.00
Number of pages = 300
Edition of the book = 8
```

```
#include <iostream>
using namespace std;
int myfunc(int x, int y=2) {
    int z;
    z = x * y;
    return z;
}

int main() {
    cout<<myfunc(16);
    cout<<"\n"; cout<<myfunc(8,9);
    return 0;
}
```

Output:

32

72

The statement:

```
struct book {
char name;
float price;
int pages;
int edition;
};
```

imply the structure definition i.e., we are defining a structure (and the data type name of the structure is book) and it consists of elements:

- **name** (which is of data type char), **price** (which is of data type float), **pages** (which is of data type int), **edition** (which is of data type int) – which are placed within the body of the structure.

The statement:

Denote the structure variable

```
struct book b1;
```

imply the **structure variable declaration**

Law of Invariant Work Rate: The overall activity rate in a programming project is statistically invariant for the duration of the program's active life

Java program:

```
import java.util.Arrays;
import java.util.LinkedHashSet;
import java.util.Set;

public class MyClass {
public static void main(String[] args) {
Set x = new LinkedHashSet();
for (int a = 1; a <= 5; a++) {
x.add(a + a);
System.out.println(Arrays.toString(x.toArray()));
}
}
}
```

```
// Output:    [2]
              [2, 4]
              [2, 4, 6]
              [2, 4, 6, 8]
              [2, 4, 6, 8, 10]
```

- **Why structure variable b1 is declared or defined?**

In order to assign the values to the elements within the body of the structure, each element must be linked with structure variable with **dot operator** or **period operator** or **member accessibility operator**.

For example: name is the element which must be linked with structure variable b1 with dot operator to assign a value B to the element "**name**".

Format string %c (corresponding to the **data type** char) in the statement:

```
printf("\n Name of the book = %c", b1.name);
```

indicates that the value to be displayed at that point in the string i.e., after the statement "\n Name of the book = " needs to be taken from b1.name.

The statement:

```
printf("\n Name of the book = %c", b1.name);
```

make provision to print the output: Name of the book = B on the screen

Format string %f (corresponding to the **data type** float) in the statement:

```
printf("\n Price of the book = %f", b1.price);
```

indicates that the value to be displayed at that point in the string i.e., after the statement "\n Price of the book = " needs to be taken from b1.price.

The statement:

```
printf("\n Price of the book = %f", b1.price);
```

make provision to print the output: Price of the book = 135.00 on the screen

```
#include<bits/stdc++.h>
using namespace std;
int main() {
float x=5.22222;
cout << floor(x) << endl;
cout << ceil(x) << endl;
cout << trunc(x) << endl;
cout << round(x) << endl;
cout << setprecision(5) << x;
return 0;
}
```

```
# Output:
5
6
5
5
5.2222
```

Java Program:

```
public class MyClass {
public static void main(String[] args) {
String x = "Albert Einstein";
for(int i=0; i<= x.length(); i++)
 System.out.println(x.substring(6));
}
}
```

```
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
Einstein
```

Law of Perceived Complexity: The additions and deletions of the subsequent releases of an evolving program are statistically invariant during its active life

Format string %d (corresponding to the **data type** int) in the statement:

```
printf("\n Number of pages = %d", b1.pages);
```

indicates that the value to be displayed at that point in the string i.e., after the statement "\n Number of pages = " needs to be taken from b1.pages.

The statement:

```
printf("\n Number of pages = %d", b1.pages);
```

make provision to print the output: Number of pages = 300 on the screen.

Format string %d (corresponding to the **data type** int) in the statement:

```
printf("\n Edition of the book = %d", b1.edition);
```

indicates that the value to be displayed at that point in the string i.e., after the statement "\n Edition of the book = " needs to be taken from b1.edition.

The statement:

```
printf("\n Edition of the book = %d", b1.edition);
```

make provision to print the output: Edition of the book = 8 on the screen.

```
# Java program:
public class MyClass {
public static void main(String[] args) {
String[] names = {"Albert", "Mary", "John"};
for(String i:names) {
 System.out.println(i);
}
}
}
```

```
# Output:

Albert
Mary
John
```

- **What will be output of the following programs?**

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main() {
cout <<abs(55) <<endl;
cout << labs(4567899L)<<endl;
cout<< llabs(45678991LL) <<endl;
return 0;
}
```

Output:

```
55
4567899
45678991
```

A)

```
#include<stdio.h>
```

```
struct book {
char name;
float price;
int pages;
int edition;
};
int main()
{
struct book b1;
b1.name = 'B';
b1.price = 135.00;
b1.pages = 300;
b1.edition = 8;
printf("\n Name of the book = %c", b1.name);
printf("\n Price of the book = %f", b1.price);
printf("\n Number of pages = %d", b1.pages);
printf("\n Edition of the book = %d", b1.edition);
}
```

Answer:

Name of the book = B

Price of the book = 135.000000

Number of pages = 300

Edition of the book = 8

```
#include <iostream>
#include <random>
using namespace std;
int main() {
int a=10, b=2,i;
int x = a - b + 1;
for (i=b; i<a; i++)
{
int c = rand() % x + b;
cout<<c<<endl;
}
return 0;
}
```

Output:

3
9
2
9
7
9
3
5

```
#include<iostream>
#include<string>
using namespace std;

int main(){
int b = 15;
string x = to_string(b);
cout << x <<endl;
return 0;
}
```

Output:

15

B)

```
#include <stdio.h>
int main(){

for( ; ; ) {
printf("This loop will run forever.\n");
}
return 0;
```

Conway's law

An aphorism that claims companies build systems that reflect their own communication framework

```
}
```

Law of Continuing Growth:

To keep users comfortable during a program's lifetime, its functional content must be continuously updated

Answer:

```
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.....continues
```

C)

```
#include<stdio.h>
int main()
{
char  ch [5];
printf("Enter the name: ");
scanf("%s", &ch);
printf( "The name you entered = %s", ch);
return 0;
}
```

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
double PI = 3.14159;
cout << fixed << setprecision(3) << PI <<endl;
return 0;
}
```

Output:

3.142

Answer:

Enter the name:

Dennis # entered name

The name you entered = **Denni** will be outputted on the screen.

Fundamental Law of Program Evolution:

Software development is governed by a dynamics that produces statistically definable patterns and invariances − enabling the software process and system to self-regulate

```
public class MyClass {
public static void main(String[] args) {
char[] x = {'A', 'B', 'C', 'D', 'E', 'F',
'G', 'H', 'I', 'J', 'K', 'L',
'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T',
'U', 'V', 'W', 'X', 'Y', 'Z'};
for(int i=0; i<x.length; i++) {
System.out.print(x[i]);
}
System.out.println();
}
}

// Output: ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Java Program

Instead of Dennis, only **Denni** will be displayed on the screen because of the statement

```
char  ch [5];
```

The statement:

```
char ch [5];
```

make provision only for 5 lettered name to be displayed on the screen.

If the statement:

```
char ch [5];
```

is replaced by the statement

```
char  ch [6];
```

Then the output on the screen is:

```
Enter the name:
Dennis # entered name
The name you entered = Dennis will be outputted on the screen.
```

```
#include <iostream>
using namespace std;

int main ()
{
string x = "Albert Einstein was a ";
string y = "German-born theoretical physicist.";
string z = x + y;
cout<<" "<< z;
return 0;
}
```

Output:

Albert Einstein was a German-born theoretical physicist.

- **Note:** %s implies the format specifier for string.

- **Program 4.4**

  Continue and break statements:

  i)

```
#include <stdio.h>
int main()
{
int i;
```

```
#include <iostream>
using namespace std;
int x = 5;
int main() {
static int a = 0;
if (cout << a << " " && a++ < x && main())
{ }
return 0;
}
```

Output:

0 1 2 3 4 5

```
for (i=1; i<=5; i++)
{
if (i==3)
{
continue;
}

printf("%d\n ", i);
}
return 0;
}
```

```
#include<iostream>
using namespace std;
int main() {
int x = 12, y = 12;
int z = -( -x-y );
cout << z << endl;
return 0;
}
```

Output:

24

Output on the screen:

1
2
4
5

```
#include<iostream>
using namespace std;
int main()
{
if (!(cout << "Albert "))
cout <<" Albert ";
else
cout << "Einstein";
return 0;
}
```

Output:

Albert Einstein

- **Note:**

```
i = 1
Is the condition (i<=5) is true?
Yes because i=1
The statement printf("%d\n ", i); is executed to print the output:
1
Now, the value of i is:
i =  1+1 = 2
Is the condition (i<=5) is true?
Yes because i=2
The statement printf("%d\n ", i); is executed to print the output:
2
Now, the value of i is:
i =  2+1 = 3
```

```
Is the condition (i<=5) is true?
Yes because i=3
The statement printf("%d\n ", i); is not executed to print the output:
3

Because of the statement:

if (i==3)
{
continue;
}

// Execution skips //

Now, the value of i is:
i = 3 + 1 = 4
Is the condition (i<=5) is true?
Yes because i=4
The statement printf("%d\n ", i); is executed to print the output:
4
Now, the value of i is:
i =  4+1 = 5
Is the condition (i<=5) is true?
Yes because i=5
The statement printf("%d\n ", i);  is executed to print the output:
5
and stop because the condition i<=5 is achieved.
```

```
#include <iostream>
using namespace std;
int a = 0;
int main() {
int a = 0;
::a = 11;
a = 12;
cout << ::a << ", " << a;
return 0;
}
```

Output:

11, 12

ii)

```
#include <stdio.h>
int main()
{
int i;
for (i=1; i<=5; i++)
{
if (i==3)
{
break;
```

```
#include <iostream>
using namespace std;
namespace num { int a = 18; }
int main() {
cout << num::a << endl;
return 0;
}
```

Output:

18

```
}

printf("%d\n ", i);
}
return 0;
}
```

```
#include<iostream>
using namespace std;
const char* name = "Albert";
int main() {
    const char* name = "Einstein";
    cout << name;
    return 0;
}
```

Output:

Einstein

Output on the screen:

1

2

```
#include <iostream>
using namespace std;
int main() {
    int x=20;
    x=!x>54;
    cout<<x;
    return 0;
}
```

Output:

0

- **Note:**

```
i = 1
Is the condition (i<=5) is true?
Yes because i=1
The statement printf("%d\n ", i); is executed to print the output:
1
Now, the value of i is:
i =  1+1 = 2
Is the condition (i<=5) is true?
Yes because i=2
The statement printf("%d\n ", i); is executed to print the output:
2
Now, the value of i is:
i = 2+1 = 3
Is the condition (i<=5) is true?
Yes because i=3
The statement printf("%d\n ", i); is not executed to print the output:
```

Because of the statement:

```
if (i==3)
{
break;
}
```

```
#include <iostream>
using namespace std;
int main() {
    int a=12,b=13;
    a=b+++a++;
    cout<<a<<" "<<b;
    return 0;
}
```

Output:

25 14

```
for loop: for (i=1; i<=5; i++)

is immediately terminated (even before the condition i<=5 is achieved) and program execution stops.
```

- **The goto statement:**

```
#include <stdio.h>
int main()
{
int i;
for(i=1; i<=5; i++)
{
if(i==3)
{
goto HAI;
}
printf("\n %d",i);
}
HAI: printf("\n Linux");
}
```

Output on the screen:

```
1
2
Linux
```

- **Note:**

```
i = 1
Is the condition (i<=5) is true?
Yes because i=1
The statement printf("\n %d ",i); is executed to print the output:
```

```
#include <stdio.h>
void num(void)  {
    auto int x = 11;
    x++;
    printf ("%d ", x);
}

int main() {
    num();
    num();
    num();
    num();
    return 0;
}
```

12 12 12 12

```
#include <stdio.h>
void  num(void)  {
    static int x = 11;
    x++;
    printf ("%d", x);
}

int main() {
    num();
    num();
    num();
    num();
    return 0;
}
```

12 13 14 15

Law of Declining Quality:

If a software system is not carefully managed and updated to a changing operational environment, its quality will appear to be declining

```
1
Now, the value of i is:
i = 1+1 = 2
Is the condition (i<=5) is true?
Yes because i=2
The statement printf("\n %d ",i); is executed to print the output:
2
Now, the value of i is:
i = 2+1 = 3
Is the condition (i<=5) is true?
Yes because i=3
The statement printf("%d\n ", i); is not executed to print the output:
3
```

Bookmark

A group of links to websites that are preserved in a web browser as shortcuts

```
Rather
The statement printf("\n Linux"); is executed to print the output:
Linux
```

```
Because of the statement:
if(i==3)
{
goto HAI;
}
for loop: for(i=1; i<=5; i++) is immediately terminated (even before the condition i<=5 is achieved)
and program execution stops.
```

Law of Feedback System:

Feedback must be gathered from a variety of sources during the complex process of software development in order to make substantial advancements

```
import java.util.*;
public class MyClass {
public static void main(String[] args) {
int[] x = {11, 12, 13, 14, 15, 16, 17};
int[] y = {11, 12, 13, 14, 15, 16, 17};
if(Arrays.equals(x, y))
System.out.println("The 2 arrays are equal.");
else
System.out.println("The 2 arrays are not equal.");
}
}  // Output: The 2 arrays are equal.
```

- **Program 4.5**

  C program to convert the upper case letter to lower case letter

```
#include<stdio.h>
int main()
{
char ch = 'A';
```

```
char b = tolower(ch);

printf("Upper case letter %c is converted to lower case letter %c", ch, b);
return 0;
}
```

Output on the screen:

```
Upper case letter A is converted to lower case letter a
```

If you want to enter the **character through the keyboard**, then the above program should take the form:

```
#include<stdio.h>

int main() {

char ch;

printf("Enter any character:");
scanf("%c", &ch);
char b = tolower(ch);

printf("Upper case letter %c is converted to lower case letter %c", ch, b);
return 0;
}
```

```
Enter any character:
C # entered character
Upper case letter C is converted to lower case letter c will be
outputted on the screen.
```

```
#include <cstring>
#include <iostream>
using namespace std;
int main() {
    char x[] = "0123456789";
    char y[] = "marchFirst1988";
    size_t z = strcspn(y, x);
    if (z < strlen(y))
    cout << " The first time a number appears in " << y << " is at position " << z;
    else
        cout << y << " doesn't include any numbers";

    return 0;
}
```

C++ program

```
# Output:
The first time a number appears in marchFirst1988 is at position 10
```

- **Program 4.6**

  C program to convert the lower case letter to upper case letter

Mainframe computer:

A powerful computer used to handle big data processing tasks

```
#include<stdio.h>
int main()
{
char ch = 'a';
char b = toupper(ch);
printf("Lower case letter %c is converted to upper case letter %c", ch, b);

return 0;
}
```

Output on the screen:

```
Lower case letter a is converted to upper case letter A
```

If you want to enter the character through the **keyboard**, then the above program should take the form:

Java program

```
import java.util.Scanner;
public class MyClass {
public static void main(String args[]) {
Scanner keyboard = new Scanner(System.in);
System.out.println(keyboard.nextLine());
keyboard.close();
}
}
```

```
Alan # entered string
Alan # Output
```

```
#include<stdio.h>
int main()
{
char ch;
printf("Enter any character:");
scanf("%c", &ch);
char b = toupper(ch);
printf("Lower case letter %c is converted to upper case letter %c", ch, b);
return 0;
}
```

```
#include <stdio.h>
int main () {
int x = 55, y = 100;
printf("Maximum number: %d\n", ((x + y) + abs(x - y)) / 2);
printf("Minimum number: %d\n", ((x + y) - abs(x - y)) / 2);
return 0;
}
```

Output:

Maximum number: 100

Minimum number: 55

Output on the screen:

```
Enter any character:
h  # entered character
Lower case letter h is converted to upper case letter H
will be outputted on the screen.
```

- **Program 4.7**

C program to test whether the entered character is upper case letter or not

```c
#include<stdio.h>
int main()
{
char ch = 'a';
if(isupper(ch))
printf("you have entered the upper case letter");
else
printf("you have entered the lower case letter");
return 0;
}
```

```c
#include<stdio.h>
int main() {
int a= 5, b = 5;
if (!(a ^ b))
printf("a = b");
else
printf("a ≠ b");
return 0;
}
```

Output:

a = b

Output on the screen:

```
you have entered the lower case letter
```

If the statement:

`char ch = 'a';` is replaced by the statement:

```c
char ch = 'A';
```

Then the **output on the screen** is:

```
you have entered the upper case letter
```

```c
#include<stdio.h>
#include<string.h>
int main() {
char x[] = "Albert";
char y[] = "Alan";
int z = strcmp(x, y);
if (z==0)
printf("2 strings are equal");
else
printf("2 strings are not equal");
return 0;
}
```

Output:

2 strings are not equal

- **Program 4.8**

  C program to test whether the entered character is lower case letter or not

```c
#include<stdio.h>
int main()
{
char ch = 'a';
if(islower(ch))
printf("you have entered the lower case letter");
```

Motherboard

The computer's main circuit board

```
else
printf("you have entered the upper case letter");
return 0;
}
```

Router

A device that provides access to the Internet by transferring data between computer networks

Output on the screen:

you have entered the lower case letter

Java Program:

```
public class HelloWorld {
public static void main(String[] args) {
boolean flag=false;
if(flag)  {
System.out.println("true");
}
else  {
System.out.println("false");
}

}
}
```

Output:

false

```
public class HelloWorld {
    public static void main(String[] args) {
   System.out.println('t' + 'a' + 'b');
   }
}
```

Output:

311

The **ASCII values** of 't', 'a', 'b' are:

t	116
a	97
b	98

116 + 97 + 98 = 311

Hence 311 is printed

- **Program 4.9**

  C program to print the value of tan inverse x (i.e., the value of $\tan^{-1}x$)

```
#include<stdio.h>
#include<math.h>
int main()
{
```

```
int x = 20;
printf("The value of tan inverse x = %f", atan(x));
return 0;
}
```

Spyware:

A piece of software that surreptitiously tracks a user's online activity and gathers private data about a user without the user's consent

Output on the screen:

```
The value of tan inverse x = 1.520838
```

- **Program 5.0**

C program to print the value of tan inverse x/y

```
#include<stdio.h>
#include<math.h>
int main()
{
int x,y;
x = 20;
y =20;
printf("The value of tan inverse x/y = %f", atan2(x, y));
return 0;
}
```

```
# Python program to convert all characters to lowercase

x = "aLBert"
print(x.casefold())
# Output: albert
```

Output on the screen:

```
The value of tan inverse x/y = 0.785398
```

Java Program:

```
public class MyClass {
static double x = 9.18;
public static void main(String args[]) {
x = x + 33.00;
System.out.println(x);
}
}

// Output: 42.18
```

- **Program 5.1**

C program to print the value of fmod(x, y)

```
#include<stdio.h>
#include<math.h>
int main()
```

```
{
float x = 20.500000;
float y =20.799999;
printf("The remainder of %f divided by %f is %f", x, y, fmod(x, y));
return 0;
}
```

Output on the screen:

```
The remainder of 20.500000 divided by 20.799999 is 20.500000
```

- **Program 5.2**

C program to print the value of ~x

```
#include<stdio.h>
int main()
{
int x, y;
x = 205;
y=~x;
printf("The value of y is:%d", y);
return 0;
}
```

Output on the screen:

```
The value of y is:-206
```

C++ Program:

```
#include <iostream>
using namespace std;
int main() {
int a;
for(a=1;a<=4;a++) {
switch(a) {
case 1: cout<<a<<endl;
break;
case 2: cout<<a<<endl;
break;
case 3: cout<<a<<endl;
break;
case 4: cout<<a<<endl;
break;
return 0;
}
}
}
```

Output:

```
1
2
3
4
```

If the statement: `y=~x;` is replaced by the statement:

```
y= -(~x);
```

Then the **output on the screen** is:

```
The value of y is: 206
```

- **Program 5.3**

C program to print the ASCII (American Standard Code for Information Interchange) value of the entered character

```
#include<stdio.h>
int main()
{
char ch ='A';
printf("The ASCII value of ch is: %d", ch);
return 0;
}
```

Python code:

```
x = 'albert einstein'

# print the number of occurrence of 'e'
print(x.count('e'))

# Output: 3
```

Output on the screen:

```
The ASCII value of ch is: 65
```

If the statement:

```
printf("The ASCII value of ch is: %d", ch);
```

is replaced by the statement:

```
printf("The ASCII value of ch is: %c", ch);
```

```
x = 'albert\t1905\tpapers'
print(x.expandtabs())
# Output: albert  1905    papers


x = 'Albert\n1905\nPapers'
print(x.splitlines())
# Output: ['Albert', '1905', 'Papers']
```

Then the **output on the screen** is:

```
The ASCII value of ch is: A
```

- **What will be the output of the following programs?**

```
x = "Alan"
print(x.zfill(4))
# Output: Alan
print(x.zfill(6))
# Output: 00Alan
print(x.zfill(8))
# Output: 0000Alan
```

i)

```
#include<stdio.h>
int main()
{
int i;
int num [5] ={16,18,19,20,21};
for(i=0;i<5;i++)
printf("\n Element = %d", num[i] +1);
return 0;
}
```

Answer:

```
Element = 17
Element = 19
Element = 20
Element = 21
Element = 22
```

```
// program to print the punctuation characters using ASCII value in C
#include <stdio.h>
#include <ctype.h>
int main() {
    int x;
    printf("Punctuation Characters in C: \n");
    for(x = 0; x <= 127; ++x) {
        if(ispunct(x)!= 0)
            printf("%c ", x); }
    return 0;
}
```

```
# Output:

All punctuations in C:

! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [ \ ] ^ _ ` { | } ~
```

ii)

```
#include<stdio.h>
int main()
{
int i = 54;
int y = i<<1;
printf("The value of y = %d", y);
return 0;
}
```

Answer:

```
The value of y = 108
```

If the statement: `i<<1` is replaced by the statement: `i<<2`

Then the **output on the screen** is:

```
The value of y = 216
```

Note:

- `i<<1` implies `54 * 2 = 108`
- `i<<2` implies `54 * 4 = 216`
- `i<<3` implies `54 * 6 = 324`
- `i<<4` implies `54 * 8 = 432`

Utility software:

A kind of system software that assists in the configuration, analysis, reinforcement, maintenance of the appropriate and efficient operation of a computer system

```c
// program to print the graphic characters using ASCII value in C
#include <stdio.h>
#include <ctype.h>
int main() {
    int x;
    printf("Graphic characters in C: \n");
    for(x=0;x<=127;++x) {
        if (isgraph(x)!=0)
            printf("%c ",x);
    }
    return 0;
}
```

 # Output:

```
Graphic characters in C:

! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8
9 : ; < = > ? @ A B C D E F G H I J K L M N O P
Q R S T U V W X Y Z [ \ ] ^ _ ` a b c d e f g h
i j k l m n o p q r s t u v w x y z { | } ~
```

iii)

```c
#include<stdio.h>
int main()
{
int i = 54;
int y = i>>1;
printf("The value of y = %d", y);
return 0;
}
```

Answer:

```
The value of y = 27
```

VoIP

An Internet-based phone connection

If the statement: `i>>1` is replaced by the statement: `i>>2`

Then the **output on the screen** is:

```
The value of y = 13
```

- **Note:**

  - i>>1 implies 54/2 = 27
  - i>>2 implies 54/4 = 13
  - i>>3 implies 54/6 = 9
  - i>>4 implies 54/8 = 6

  << **implies:** left shift operator

  >> **implies:** right shift operator

Python code:

```
# Get the documentation of math.pow

help('math.pow')
```

```
// Import the Scanner class
import java.util.Scanner;
public class MyClass {
  public static void main(String[] args) {
    // Create a Scanner object
    Scanner x = new Scanner(System.in);
    System.out.println("Enter a string: ");
    // Read the user input
    String y = x.nextLine();
    // Output user input
    System.out.println("Entered string is: " + y);
  }
}

// Output:
Enter a string:
Alan # entered string
Entered string is: Alan
```

- **Program 5.4**

  C program to print the length of the entered character (i.e., to print the length of the string)

```
#include<stdio.h>
#include<string.h>
int main()
{
char ch[4];
printf("Enter any word: ");
scanf("%c", &ch);
printf("The length of the string = %d", strlen(ch));
return 0;
}
```

```
import java.util.Scanner;
import static java.lang.System.in;
import static java.lang.System.out;
public class MyClass {
  public static void main(String[] args) {
    Scanner x = new Scanner(in);
    out.println("Enter a string: ");
    String y = x.nextLine();
   out.println("Entered string is: " + y);
  }
}
```

Output on the screen:

```
Enter any word:
dog # entered word

The length of the string = 3
```

will be displayed on the console screen because **there are only 3 letters** in the word "dog".

Suppose if you enter the word tech

```
The length of the string = 4
```

will be displayed on the console screen because **there are only 4 letters** in the word tech.

- **Program 5.5**

  C program to print the factorial of the entered number

```
#include<stdio.h>
int main()
{
int i, n, fact=1 ;
printf("Enter any number:");
scanf("%d", &n);
for(i=1; i<=n; i++)
fact = fact *i;
printf("\n Entered number is: %d", n);
printf("\n The factorial of the entered number %d is: %d", n, fact);
return 0;
}
```

```
# import the module
import keyword
# print the keywords
print("Python keywords:")
print(keyword.kwlist)
```

```
Python keywords are:
['False', 'None', 'True', 'and', 'as', 'assert', 'async',
'await', 'break', 'class', 'continue', 'def', 'del', 'elif',
'else', 'except', 'finally', 'for', 'from', 'global', 'if',
'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass',
'raise', 'return', 'try', 'while', 'with', 'yield']
```

```
public class MyClass {
public static void main(String[] args) {
double x = 6.54;
double y = 50.00, z = 0.99;
x += y;
System.out.println(x);
// Output: 56.54
x -= z * 2;
System.out.println(x);
// Output: 54.56
}
}
```

Output on the screen:

```
Enter any number:
2 # entered number
Entered number is: 2
The factorial of the entered number 2 is: 2
will be displayed on the screen.
```

Suppose if you enter the number 4

Entered number is: 4

The factorial of the entered number 4 is: 24

will be displayed on the screen.

Storage class	Purpose
auto	It is a default storage class
extern	It is a global variable
static	It is a local variable which is capable of returning a value even when control is transferred to the function call
register	It is a variable which is stored inside a Register

```cpp
#include <cstring>
#include <iostream>
using namespace std;
int main() {
  char x[] = "Albert Einstein";
  // Break the string when it encounters empty space
  char* i = strtok(x, " ");
  cout <<  i;
  return 0;
}  // Output: Albert
```

Python code

```python
class Math:

  def myfunc(x, y):
      return x * y

# create myfunc static method using staticmethod()
Math.myfunc = staticmethod(Math.myfunc)
print('15*3=', Math.myfunc(15, 3))
# Output: 15*3= 45
```

- **What will be the output of the following program?**

```cpp
#include <stdio.h>
int main()
{
printf("\nLinux \' linux ");
printf("\nLinux \? linux ");
return0;
}
```

Java Program:

```java
public class MyClass {
public static void main(String[] args) {
System.out.println("Hames".replace('H', 'J'));
// Output: James
}
}
```

Answer:

Linux ' linux

Linux ? linux

```c
#include<stdio.h>
#include<stdlib.h>
int main () {
printf("linux\n");
exit (0);
printf("php\n");
return 0;
}
```

```java
import java.lang.Math;
public class MyClass {
public static void main(String[] args) {
int x = 100, y = 70;
System.out.println(Math.subtractExact(x, y));
 }
}
// Output: 30
```

Answer:

linux

Algorithm Animation:

Computer program operations, data and interpretations are abstracted – and then animated graphical representations of these abstracts are produced.

```java
import java.lang.Math;
public class MyClass {
public static void main(String[] args) {
int x = 100, y = 70;
System.out.println(Math.multiplyExact(x, y));
 }
}
// Output: 7000
```

- **What will be the output of the following program?**

```c
#include<stdio.h>
int main() {
int a, b, c;
a=2;
b=2;
c = a ^ b;
printf( "The value of c = %d", c);
}
```

```java
import java.lang.Math;
public class MyClass {
public static void main(String[] args) {
int x = 100;
System.out.println(Math.incrementExact(x));
 }
}
// Output: 101
```

Answer:

The value of c = 0

Strategy or paradigm for implementing a process of computation

Algorithmic problem solving steps:

1. Understanding the problem
2. Deciding on computational means, exact versus approximate solving, data structures and **algorithm design technique**
3. **Design an algorithm**
4. Prove accuracy
5. Analyze the algorithm
6. Code the algorithm

Block-based coding

A type of programming language where instructions are mainly represented as blocks

Conditionals

Statements that only run under certain conditions

Function

A piece of code that we can easily call over and over again

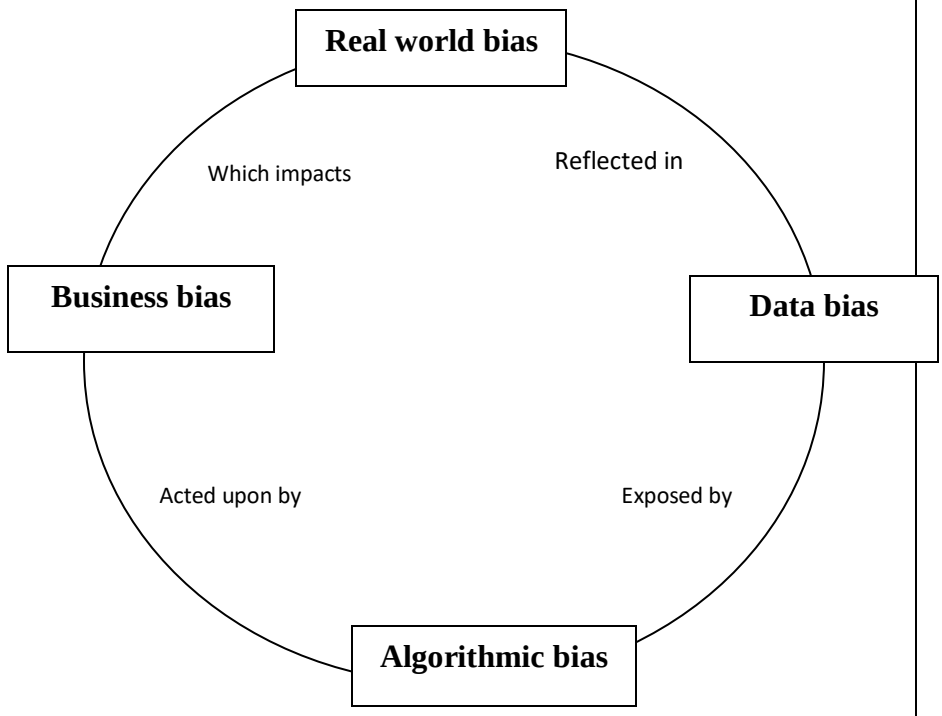
Function call

A piece of code that we add to a program to indicate that the program should run the code inside a function at a certain time.

Function definition

A code inside a function that instructs the program on what to do when the function is called

4 Stages of Ethical AI



Loop

The action of doing something over and over again

While loop: **A loop that continues to repeat while a condition is true**

Pattern matching

The process of checking whether a specific sequence of data exists among the given data

DNS (domain name service)

The service that translates URLs to IP addresses.

Packets

Small pieces of information that have been carefully formed from larger pieces of information

Automated Reasoning: Applying **reasoning** in the form of logic to define, approach and solve problems

- Define the problem the program will be required to solve.
- Decide what programming language, rules of logic and functions the program will use.
- Specify the process and tools that the program will use to analyze data and reach a conclusion.
- Create a control flow process to perform all these procedures efficiently.

Data

The information that is saved on a computer

To create a file in a 'C' program following syntax is used:

```
FILE *fp;
fp = fopen ("file_name", "mode");
```

```
#include <stdio.h>
int main() {
FILE *fp;
fp = fopen ("data.txt", "w");
}
```

Universal Resource Locator

(URL)

A way of accessing an Information resource on the Internet

- **WWW:** World Wide Web
- **WAN:** Wide Area Network
- **ZIP:** Zone Information Protocol

What is a Function in C?

Function in **C programming** is a reusable block of code that makes a program easier to understand, test and can be easily modified without changing the calling program. **Functions** divide the code and modularize the program for better and effective results. In short, a larger program is divided into various subprograms which arecalled as *functions*.

Java Development Kit = Java Runtime Environment + Development tools

Java Runtime Environment = Java Virtual Machine + Libraries to execute the application

Java Virtual Machine = Runtime environment to execute Java byte code

Java Program:

```
public class HelloWorld extends Thread {
   public void run(){
       System.out.println("Albert Einstein ...");
   }
   public static void main(String args[]){
       HelloWorld ab = new HelloWorld();
       ab.start();
   }
}
```

Implementing a thread by extending the Thread class

Output:

Albert Einstein ...

```
public class HelloWorld implements Runnable {
   public void run(){
       System.out.println("Albert Einstein ...");
   }
   public static void main(String args[]){
       Thread ab = new Thread(new HelloWorld ());
       ab.start();
   }
}
```

Implementing a thread using the method of Runnable interface

Output:

Albert Einstein ...

Java Program:

```
import java.lang.Math;
public class MyClass {
public static void main(String[] args) {
int x = 100;
System.out.println(Math.decrementExact(x));
}
}
// Output: 99
```

```
import java.lang.Math;
public class MyClass {
public static void main(String[] args) {
int x = -16;
int y = 35;
System.out.println(Math.negateExact(x));
// Output: 16
System.out.println(Math.negateExact(y));
// Output: -35
}
}
```

```
import java.lang.Math;
public class MyClass {
  public static void main(String[] args) {

    // create long variable
    long x = -54965L;
    long y = 198876L;

    // change long to int
    int a = Math.toIntExact(x);
    int b = Math.toIntExact(y);

    // print the int value
    System.out.println(a);
    // Output: -54965
    System.out.println(b);
    // Output: 198876

  }
}
```

Abstraction

↓

A simplified representation of something more complex

Data centre rationalization

↓

The restructuring and upgrading of IT resources to fulfill the enterprise organizational needs in terms of business, operations and technology

A bit is the single unit of information in a computer – typically represented as a 0 or 1

Computational thinking: A set of techniques for solving complex problems

Decomposition: Break a problem into parts

Pattern recognition:

Interpreting patterns and models

(Looking for similarities and trends)

Abstraction:

- Focusing on what is important and ignoring what is unnecessary
- Organizing data logically

Algorithm design: step by step instructions for solving the problem

- Designing and implementing algorithms

Debugging: Fixing errors within the algorithm

Generalization: Extending a solution for a particular problem to other kinds of problems

Python code:

```
print("{:,}".format(892887872878))

# Output: 892,887,872,878
```

C / C++ categorize statements into:

- **Selection [**if and switch**]**
- **Iteration [**while, for, do-while**]**
- **Jump [**break, continue, goto and return**]**
- **Label [** case and default**]**
- **Expression [** statements composed of a valid expression**]**
- **Block [**blocks of code. Each block begins with { and ends with } and referred to as compound statements**]**


```
import java.lang.Math;

public class MyClass {

public static void main(String[] args) {

double x = 8.68;

System.out.println(Math.nextDown(x));

// Output: 8.679999999999998

System.out.println(Math.nextUp(x));

// Output: 8.680000000000001

}

}
```

Suppressing input:

```
scanf("%d %*d %d", &a, &b, &c);
```

Input

a=15, b=50, c=55

Here 15 is stored in a, 50 skipped and 55 is stored in the b, since no data is available for c so it has garbage value. The character * is called the **suppression character**.

The infinite loop:

```
for( ; ; )

printf("This Loop will run forever.\n";)
```

```
for( ; ; ) {
ch = getchar(); ⟶ Get a character
if(ch=='B')
```

```
import java.lang.Math;

public class MyClass {

public static void main(String[] args) {

    double x = 2.693;

    double y = -3.896;

    System.out.println(Math.copySign(x, y));

    // Output: -2.693

}

}
```

```
break; ——————→ Exit the loop
}
printf("you entered B";)

// This loop will run until the user types a Letter B at the keyword.
```

C++ Programming

Paradigms	Multi-paradigm: procedural, functional, object-oriented, generic
Family	C
Designed by	**Bjarne Stroustrup**
Developer	ISO/IEC JTC1 (Joint Technical Committee 1) / SC22 (Subcommittee 22) / WG21 (Working Group 21)
First appeared	1985; 35 years ago
Stable release	C++17 (ISO/IEC 14882:2017) / 1 December 2017; 2 years ago
Preview release	C++20
Typing discipline	Static, nominative, partially inferred
OS	Most major
Filename extensions	.C, .cc, .cpp, .cxx, .c++, .h, .hh, .hpp, .hxx, .h++
Website	isocpp.org

Bjarne Stroustrup

Creator of C++ language

Major implementations
GCC, LLVM Clang, Microsoft Visual C++, Embarcadero C++Builder, Intel C++ Compiler, IBM XL C++, EDG
Influenced by

Ada, ALGOL 68, C, CLU, ML, Mesa, Modula-2, Simula, Smalltalk
Influenced
Ada 95, C#, C99, Chapel, Clojure, D, Java, JS++, Lua, Nim, Perl, PHP, Python, Rust, Seed7

```cpp
#include <iostream>
#include <string>
using namespace std;
int main() {
string x = "Albert";
cout << &x; // Output: 0x7ffd8e332470
return 0;
}
```

Memory address of the string "Albert"

Programming methodology that views a computer program as a combination of variables, functions and data structures called objects

An **Object-oriented** high level cross-platform general-purpose case-sensitive language developed by a Danish computer scientist **Bjarne Stroustrup** (at Bell Labs circa 1980) as an **extension of the C language** for a specific purpose[C was developed for programming Operating Systems]. Initially named **C with classes** which later named C ++ in 1983. As a **successor of C language**, C++ has been certified as a **99.9 percent pure standard** and possesses exceptional performance, efficiency and flexibility of use compared to C language. It well deserves the widely acknowledged nickname "**Swiss Pocket Knife of Languages**."

Advantages: C++ is extremely fast and has the power and extensibility to write large-scale programs and runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. Most famous software has their backbone in C++ and many programming languages depend on C++'s performance and reliability in their implementation [Examples include: JVM, JavaScript interpreters and Web frameworks]. C++ [a superset of C] is said to use static typing when type checking is performed during compile-time as opposed to run-time.

Limitation: Some tasks can be implemented in C++, though not very quickly. For example: designing GUI screens for applications. Other programming languages like VB, Python have GUI design elements built into them. Therefore, they are better suited for GUI type of task.

Uses: Used in the development of new programming languages [C#, Java, JavaScript, Perl, PHP, Python and Verilog], Apple Macintosh, PC running Windows, operating systems and Adobe Systems (like Photoshop, Acrobat etc), softwares for MRI machines, high-end CAD/CAM systems. C++ fully supports most important features of object-oriented programmingincluding the four pillars of **object-oriented development**:

- Encapsulation
- Data hiding
- Inheritance
- Polymorphism

Inheritance: The ability of a class (sub class) to derive properties and characteristics from another class (super class) is called **Inheritance**. Inheritance is one of the most important features of Object Oriented Programming.

- **Sub Class:** The class that inherits properties from another class is called Sub class or Derived Class.
- **Super Class:** The class whose properties are inherited by sub class is called Base Class or Super class.

```
// My First C++ Program ⟶ Single line Comment

#include<iostream> ⟶  Preprocessor command

int main() ⟶ Main function. Execution of C++ program begins from main()

{ ⟶  Beginning of the main function

std::cout<<"Hello, world!"; ⟶  Output statement

return 0; ⟶ Terminates the execution of the main function

} ⟶  End of the main function

/* you can still use C style comment */
```

Process of C++ program execution:

Every 'C++' program follows a basic structure**. A C++ program:**

```
#include<iostream> //Pre-processor directive
int main() //main function declaration
{
std::cout<<"Hello, world!"; // output the string on a display screen
return 0; //terminating function
```

```
}
```

is written using **Text Editor** such as :

- **Notepad** (in case of Windows Operating System),
- **vim** or **vi** (in case of Linux Operating System)

and saved with **[.cpp] Extension** – For example, hello.cpp. File Saved with [.cpp] extension is called **Source Program** or Source Code. C++ Source code with **[.cpp] Extension** is sent to preprocessor first. The preprocessor generates an expanded source code.

Expanded source code is given as input to **compiler** where the expanded source program is compiled (i.e., the program is entirely read and translated to instructions the computer can understand i.e., machine understandable or readable language i.e., to **machine code sequence** of 0s and 1s). If the C++ compiler finds any error during compilation, it provides information about the error to the programmer.

The programmer has to review code and re-edit the program. After re-editing program, **Compiler** again check for any error. If program is error-free then it is sent to assembler − where the code is assembled and converted into object code. Now a simple **.obj file** is generated.

The object code is sent to linker (where the object code is linked with appropriate libraries). Then it is converted into a single executable code. A simple **.exe file** is generated.

The executable code is sent to loader (where the executable code is loaded into memory and then it is executed). After execution, output:

```
Hello, world!
```

is displayed on the console screen.

Input	Program	Output
Source code	Preprocessor	Expanded code
Expanded code	Compiler	Assembly code
Assembly code	Assembler	Object code
Object code	Linker	Executable code
Executable code	Loader	Execution

`#include<iostream>` → preprocessor statement

This statement begins with # symbol and is also called preprocessor directive. This statement directs the preprocessor to include a section of standard C++ code [header iostream] that supports C++ style input output operations [<iostream> is to C++ what stdio.h is to C, allows to perform standard input and output operations – such as writing the output of this program "Hello, world!" to the console screen]. There is no .h extension to the name iostream. This is because that <iostream> is one of the modern style headers defined by standard C++.

- **iostream** means input output screen

  - i → input
  - o → output
  - stream → screen

iostream comprises input output functions like cout, cin etc.

- **cin** is a input function (**cin** means console input)
- **cout** is a output function (cout means console output)

and it is included into the C++ program by writing the statement **#include<iostream>**. The statement #include tell the compiler to include the contents of the file iostream before compilation.

If a program is written without the statement: #include<iostream> then the **C++ compiler** can't compile and a **compilation error** is displayed on the screen (because **C++ compiler** fails to recognize the functions such as **cin** and **cout**).

- main()→ main function

The function named main is a special function in all C++ programs; it is the function called when the C++ program is executed. As the name itself indicates this is the main function of every C++ Program. Parentheses " () " indicate a function and the word **main** indicate the name of the function.

`main()` **implies:** main function. Execution of C++ program begins from `main()`. No C++ program is executed without `main()`. It is case sensitive language: only lower case letters (or small letters) must be used and should not be enclosed by a semicolon. There must be one and only one `main()` function in every C++ program.

We can represent the main function in various forms, such as:

- `main()`
- `int main()`
- `void main()`
- `main(void)`
- `void main(void)`
- `int main(void)`

Classes and objects are supported by C++ but not by C

As we know **C++ is Platform dependent language**. So the Operating system needs to know when the program execution ends. So when there is value returns from the main function, the Operating System get to know that the program execution is over. **`int main()` implies:** `main()` should return integer value.

- If the main function **returns 0 to the operating system**, then the program has completedexecution successfully.
- If the main function **returns 1 to the operating system**, then the program has notcompleted execution successfully.

int main() {

}

Body of the **main function** within which the sequence of instructions to the computer to perform specific operations in the form of statements [input-output statements, arithmetic statements, control statements and other statements] are **written and executed**

string x = "Hi Alan";

String x in memory:

'H'	'i'	' '	'A'	'l'	'a'	'n'

Length: 7

```
#include <iostream>
#include <cstring>
using namespace std;

int main() {

  string x = "Hi Alan";
  cout << x.length();
  return 0;

}


// Output: 7
```

The left curly brace

```
{
```

implies: the beginning of the main function and the right curly brace

```
}
```

implies: the end of the main function

These braces can also be used to indicate the beginning and end of user-defined functions and compound statements.

`return 0;` → implies the exit status of execution of the program i.e., at this point, main function returns back the control of the computer to the operating system since the **execution is terminated** at this point and once a **return statement** i.e., `return 0;` is executed, no further instructions within the main function are executed.

For example:

```
#include<iostream>
int main() {
std::cout<<"Hello, world!";
return 0;
std::cout<<"Hello, world!";
}
```

```
#include <iostream>
using namespace std;
int main() {
    cout<<"Hello, World!";
    return 0;
    cout<<"Hello, World!";
}
```

Output on the screen:

```
Hello, world!
```

`;` → implies **semicolon** or **statement terminator** [*a delimiter of the declaration*] → All C++ statements must end with a semicolon character. One of the most common syntax errors in C++ is forgetting to end a statement with a semicolon. A program is a well-defined set of instructions and each well-defined instruction (in the form of a statement) is ended by a semicolon (which is **C++ language punctuation** − like a period in English i.e., in an English paragraph eachsentence is ended by a full stop which tells that one sentence ends and another begins, semicolon implies the end of one logical entity − that one instruction (or statement) ends and another begins).

`// My First C++ Program` → **Comment**

A good programmer who writes codes understood by a human is better than a programmer who generates codes understood only by the machine. So, it is highly recommended to insert comments. Comment is explanatory note on some instruction. Two slash signs indicate that the rest of the line is a comment inserted by the programmer but which has no effect on the behavior of the C++ program. Comment statement is not compiled and executed.

- "\n" is used in both C and C++ but "\n" occupies 1 byte memory
- endl is used in C++ but not in C
- endl doesn't occupy any memory

```cpp
#include <iostream>
using namespace std;

int main() {
    cout << "Alan"<<"\n";
    cout << "Albert";
    return 0;
}

// Output:
                Alan
               Albert
```

```cpp
#include <iostream>
using namespace std;

int main() {
    cout << "Alan"<<endl;
    cout << "Albert";
    return 0;
}

// Output:
                Alan
               Albert
```

cout→ implies **the standard way of displaying output on the screen and** output function of the C++ language which makes provision to print the output:

```
Hello, world!
```

on the console screen. The statement

```
std::cout<<"Hello, world!";
```

has three parts:

```cpp
#include <iostream>
using namespace std;
int main() {
// single insertion operator
cout<<"Hi\n";
// multiple insertion operators
cout<<"Alan"<< " "<<"Turing"<<endl;
return 0;
}
```

```
// Output:
Hi
Alan Turing
```

First, std::cout which allows us to send data to the console screen to be printed as text. std::cout is defined in the iostream header file.

- std → standard
- :: → scope resolution operator
- cout → console output or character output

> std::cout **basically means**: look in standard library and get cout function.

Second, the insertion operator (<<), which output the text:

Hello, world!

on the console screen using the **cout** function.

Finally, a string (Helloworld!) enclosed by the double quotation marks is the content inserted into the standard output.

C program:

```c
#include <stdio.h>
int main() {
    printf("%lu\n", sizeof(22));
    // Output: 4
    printf("%lu\n", sizeof(int));
   // Output: 4
    return 0;
}
```

C++ program:

```cpp
#include <iostream>
using namespace std;
int main() {
     cout<<sizeof(22)<<endl;
     // Output: 4
     cout<<sizeof(int)<<endl;
     // Output: 4
    return 0;
}
```

sizeof(22) and sizeof(int) are the same in C++ and in C

```c
#include <stdio.h>
int main() {
    printf("%lu\n", sizeof('v'));
    // Output: 4
    printf("%lu\n", sizeof(char));
   // Output: 1
    return 0;
}
```

sizeof('v') and sizeof(char) are the same in C++ but not in C

```cpp
#include <iostream>
using namespace std;
int main() {
     cout<<sizeof('v')<<endl;
     // Output: 1
     cout<<sizeof(char)<<endl;
     // Output: 1
    return 0;
}
```

If the statement:

```
using std::cout;
```

is added below the statement:

```
#include<iostream>
```

then the **program** takes the form:

```
#include<iostream>
using std::cout;
int main()
{
cout<<"Hello, crazy world!";
return 0;
}
```

C Program:

```
#include <stdio.h>
int main() {
int x, y;
for (x=1; x<=3; x++) {
for (y=0; y<=x; y++)
printf("%d", y);
printf("\n");
}
return 0;
}
```

Output:

01
012
0123

i.e., no need to include `std::` in the statement:

```
std::cout<<"Hello, crazy world!";
```

C++ does not have strict rules on indentation:

```
int main() { std::cout << " Hello World!\n"; } → No ERROR
```

C++ Keywords:

auto	break	case	char	const
double	else	enum	extern	float
int	long	register	return	short
struct	switch	typedef	union	unsigned

continue	default	do
for	goto	if
signed	sizeof	static
void	volatile	while

A list of 30 Keywords in C++ Language which are not available in C language:

asm	dynamic_cast	namespace
explicit	new	static_cast
operator	template	friend
this	inline	public
delete	mutable	protected
typeid	typename	using

reinterpret_cast	bool
false	catch
private	class
throw	const_cast
true	try
virtual	wchar_t

```
int main()
{
int i, j;
i=10;
j=i*2;
return j;
} ⟶ NO ERROR

int main()
{
int i;
i=10;
```

C Program:

```
#include<stdio.h>
enum names{Albert, Alan, John, David, Turing, Weinberg, Friedrich};
int main()
{
    enum names scientist;
    scientist = John;
    printf("%d", scientist);
    return 0;
}
```

Output:

2

```
int j;
j=i*2;
return j;
} ⟶ ERROR
```

C++ data types

Type	Keyword
Boolean	bool
Character	char
Integer	int
Floating point	float
Double floating point	double
Valueless	void
Wide character	wchar_t

```
#include <iostream>
using namespace std;

int main() {
   cout << "Size of char: " << sizeof(char) << endl;
   cout << "Size of int: " << sizeof(int) << endl;
   cout << "Size of short int: " << sizeof(short int) << endl;
   cout << "Size of long int: " << sizeof(long int) << endl;
   cout << "Size of float: " << sizeof(float) << endl;
   cout << "Size of double: " << sizeof(double) << endl;
   cout << "Size of wchar_t: " << sizeof(wchar_t) << endl;

   return 0;
}
```

The output on the screen:

```
Size of char: 1
Size of int: 4
Size of short int: 2
Size of long int: 4
Size of float: 4
Size of double: 8
Size of wchar_t: 4
```

```
#include <iostream>
using namespace std;
int main() {
    int y;
    unsigned char x=0x70;
    y=x<<1;
    cout<<y<<endl;
    return 0;
}
```

Output:

224

- **Program 1.1**

C++ program to print the word "hello Bill Gates" on screen

```
#include<iostream>
using std::cout;
int main()
{
cout<<"hello Bill Gates";
return 0;
}
```

The output on the screen:

```
hello Bill Gates
```

```
#include <iostream>
using namespace std;
void myfunc (int x) {
cout << "  " << x << endl;
}
void myfunc (double y) {
cout << "  " << y << endl;
}
void myfunc (char* z) {
cout << " " << z << endl;
}
int main() {
myfunc (13);
myfunc (13.12);
myfunc ("Thirteen");
return 0;
}
```

Output:

13

13.12

Thirteen

- **Program 1.2**

C++ program to print

```
    *
   *****
   *****
   *****
   *****
```

on screen

```
#include<iostream>
using std::cout;
int main()
{
cout<<"\n     *     ";
cout<<"\n   *****   ";
cout<<"\n   *****   ";
cout<<"\n   *****   ";
cout<<"\n   *****   ";
return 0;
}
```

```
#include <iostream>
using namespace std;
template <typename T>

int mult (T x,T y) {
return (x*y);
}
int main() {
cout<<mult<int>(2,4);
return 0;
}
```

Output:

8

The output on the screen:

```
     *
   *****
   *****
   *****
   *****
```

If **new line** `\n` is not included in the above program then the output on the screen is:

```
**********************
```

- `endl` can be used instead of **`\n`**:

```
#include<iostream>
using std::cout;
int main()
{
cout<<"         *     "<< endl;
cout<<"       *****   "<< endl;
cout<<"       *****   "<< endl;
```

```
#include <iostream>
using namespace std;
class Name {
public:
Name() {
cout << "Albert Einstein";
}
};

int main() {
Name obj;
return 0;
}
```

Output:

Albert Einstein

```
cout<<"       *****   "<< endl;
cout<<"       *****   "<< endl;
return 0;
}
```

The output on the screen:

```
// Error ⟶ endl was not declared in this scope //
```

If the above program is rewritten:

```
#include<iostream>
using std::cout;
using std::endl;
int main()
{
cout<<"       *     "<< endl;
cout<<"     *****   "<< endl;
cout<<"     *****   "<< endl;
cout<<"     *****   "<< endl;
cout<<"     *****   "<< endl;
return 0;
}
```

```
#include <bits/stdc++.h>
using namespace std;
int main(){
vector<int> num{
11,19,14,13,12,18,15,17};
sort(num.begin(), num.end());
for (auto a : num)
cout << a << "" ""<< endl;
return 0;
}
```

Output:

```
11
12
13
14
15
17
18
19
```

The output on the screen:

```
                    *
                  *****
                  *****
                  *****
                  *****
```

The single statement:

```
#include<iostream>
using namespace std;

int main() {
int x= 20, y= 50;
bool z, p;
z= (x == y);
p= (x < y);
cout << z <<endl;
cout << p << endl;
return 0;
}
```

Output:

```
0
1
```

```
using namespace std;
```

can be used instead of the statements:

```
using std::cout;
using std::endl;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
cout<<"       *     "<< endl;
cout<<"     *****   "<< endl;
cout<<"     *****   "<< endl;
cout<<"     *****   "<< endl;
cout<<"     *****   "<< endl;
return 0;
}
```

```
#include<iostream>
#include<vector>
using namespace std;

int main() {
vector <string> a;
a.push_back("Stephen ");
a.push_back("William Hawking");
for(vector <string>::iterator x=a.begin();x!=a.end();++x)
cout<<*x;
return 0;
}
```

Output:

Stephen William Hawking

The output on the screen:

```
      *
     *****
     *****
     *****
     *****
```

```
#include<iostream>
using namespace std;
int main(){
for( unsigned int x = 1; x <= 10; x++)
if( x & 0x00000001 )
cout << x << "  " <<endl;
return 0;
}
```

Print out all odd numbers from 1 to 10

C++'s Predefined Streams

Stream	Meaning	Default device
cin	Standard input	Keyboard
cout	Standard output	Screen
cerr	Standard error output	Screen
clog	Buffered version of cerr	Screen

C Program:

```
#include <stdio.h>
int main() {
    if(printf("Albert Einstein")) {
    }
    return 0;
}
```

Output:

Albert Einstein

```
#include <stdio.h>
void abc(void) {
    void abx(void);
    printf("\n Albert Einstein");
    abx();
}

int main() {
    abc();
    return 0;
}

void abx(void) {
    printf("\n Alan Turing");
}
```

Output:

Albert Einstein

Alan Turing

```
#include <stdio.h>
int main() {
    int a=2;
    if(a==0)
      return;
    printf("Value is not equal to Zero.");

}
```

Output:

Value is not equal to Zero.

Resource starvation

A situation in which a process does not receive the resources it requires to finish a task due to resource allocation to other processes

Java Program:

```
public class HelloWorld {
    static { System.out.println("Alan Turing");}
  public static void main(String args[]){
   System.out.println("Albert Einstein");
  }
}
```

Output:

Alan Turing

Albert Einstein

```
public class HelloWorld {
    public static void main(String a[]){
        String name = "Albert Einstein";
  name = name.replaceAll("Albert", "Lieserl");
        System.out.println(name);
    }
}
```

Output:

Lieserl Einstein

```
import java.util.ArrayList;
import java.util.Collections;

public class HelloWorld {
     public static void main(String[] args) {
         ArrayList<String> names= new ArrayList<String>(2);
         names.add("John");
         names.add("Albert");
         System.out.println("Old array list: " + names);

         names.ensureCapacity(4);
         names.add("Alan");
         names.add("Robert");
         System.out.println("New array list: " + names);
   }
}
```

Autonomous

Does not require human input to function

Output:

Old array list: [John, Albert]

New array list: [John, Albert, Alan, Robert]

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println(1900 + 05 + "Papers");
        System.out.println("Office" + 36 + 05);
    }
}
```

Output:

1905Papers

Office365

```
#include <iostream>
int main () {
  int n = -77;
  std::cout.width(6); std::cout << std::internal << n << '\n';
  std::cout.width(6); std::cout << std::left << n << '\n';
  std::cout.width(6); std::cout << std::right << n << '\n';
  return 0;
}
```

Java Program:

```
public class HelloWorld {
    public static void main(String[] args) {
       System.out.println(381* 05 + "Papers");
        System.out.println("Office" + 73 * 05);
    }
}
```

Output:

1905Papers

Office365

The output on the screen:

```
-   77
-77
   -77
```

- **Write a program to print the following outputs:**

(a)

*

****

****

****

*

(b)

```
#include <iostream>
using namespace std;
int main() {
    double Pi = 3.141592653589;
    cout.precision(4);
    cout << Pi << endl;
    // Output: 3.142
    cout.precision(5);
    cout << Pi << endl;
   // Output: 3.1416
    return 0;
}
```

In-Place Algorithm

An algorithm for transforming the input without the use of additional memory

****

* *

* Hello World! *

* *

****

(c)

```
#include <iostream>
using namespace std;
int main() {
    char x[] = "Albert";
    char y = 'A';
    cout.write(x, 3);
    cout<<endl;
    cout.put(y);
    return 0;
}
```

Output:

Alb

A

Braces come in pairs!

Comments come in pairs!

All statements end with a semicolon!

Spaces are optional!

Must have a main function!

C++ is done mostly in lowercase. It's a case-sensitive language

Answers:

(a)

C program:

```
#include <stdio.h>
int main()
{
    char a[50];
    int b = sprintf(a, "Albert Einstein grew up in Munich");
    printf("\nThe total number of characters in the string:\n %s is: %d.", a, b);
    return 0;
}
```

Output:

The total number of characters in the string:

Albert Einstein grew up in Munich is: 33.

```
#include<iostream>
using namespace std;
int main()
{
cout<<"\n           *     ";
cout<<"\n         ****    ";
cout<<"\n        *******  ";
cout<<"\n         ****    ";
cout<<"\n           *     ";
return 0;
}
```

Python code:

```
string = "Albert"
list = ["Computer", "Mouse", "Keyboard", "Camera", "Disk"]
tuple = ("Car", "Flight", "Ship")
dictionary = {
    "Albert": "Einstein",
    "John": "Turing",
    "David": "Hilbert",
    "Alan": "Mary",
}

print("The minimum value of string is: ", min(string))
print("The minimum value of list is: ", min(list))
print("The minimum value of tuple is: ", min(tuple))
print("The minimum value of dictionary is: ", min(dictionary))
```

(b)

Output:

The minimum value of string is: A
The minimum value of list is: Camera
The minimum value of tuple is: Car
The minimum value of dictionary is: Alan

```
#include<iostream>
using namespace std;
int main()
{
cout<<"\n           ***************         ";
cout<<"\n                 * *               ";
cout<<"\n           * Hello World! *         ";
cout<<"\n                 * *               ";
cout<<"\n           ***************         ";
return 0;
}
```

```
x = [11, 12, 13, 14, 15]
y = iter(x)
print(next(y))
```

(c)

Output:

11

```
#include<iostream>
using namespace std;
int main()
{
cout<<"\n Braces come in pairs!";
cout<<"\n Comments come in pairs!";
cout<<"\n All statements end with a semicolon!";
cout<<"\n Spaces are optional!";
cout<<"\n Must have a main function!";
cout<<"\n C++ is done mostly in lowercase. It's a case-sensitive language";
return 0;
}
```

```
def myfunc(x, y):
    print (x/y)

def awp_1(func):
    def awp (x, y):
        if x < y:
            x, y = y, x
        return func(x, y)
    return awp

myfunc = awp_1(myfunc)
myfunc(2, 8)
```

Output:

4.0

An old-style C++ program:

```
#include<iostream.h>
int main()
{
return 0;
}

// It includes the file iostream.h, not the header <iostream>.
// No namespace statement is present.
```

```
def myfunc():
    yield "a"
    yield "b"
    yield "c"
    yield "d"

for i in myfunc():
    print(i)
```

Output:

a

b

c

d

A modern-style C++ program that uses the new style headers and a namespace:

```
#include<iostream>
Using namespace std;
int main()
{
return 0;
}

// It includes C++ style header and specifies a namespace.
```

Integer overflow

A kind of an arithmetic overflow error results when an arithmetic operation outputs a numeric value that goes outside allocated memory space or overflows the range of the specified integer value

C Program:

```
#include <stdio.h>
int main() {
int i;
for(i=97; i<=122; i++)
{
   printf("%c ", (char)i);
}
  return 0;
}
```

Output:

a b c d e f g h i j k l m n o p q r s t u v w x y z

- **Program 1.3**

  C++ program to find the area of a circle

```
#include<iostream>
using namespace std;
main()
{
int r, area;
r = 2;
area = 3.14 * r * r;
cout<<"The area of the circle = "<< area;
return 0;
}
```

Informed Search	Uninformed Search
AI receives recommendations on how and where to find a solution to any issue	AI does not receives recommendations on how and where to find a solution to any issue

The output on the screen:

```
The area of the circle = 12
```

If the statement:

```
float r, area;
```

is used instead of

```
int r, area;
```

Duplex transmission

At the same time, data is transmitted in both directions

i.e.,

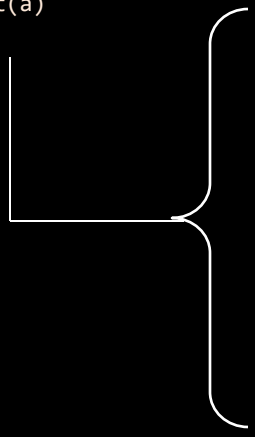
```
#include<iostream>
using namespace std;
int main()
{
float r, area;
r = 2;
area = 3.14 * r * r;
cout<<"The area of the circle = "<< area;
return 0;
}
```

Then the **output on the screen**:

```
The area of the circle = 12.56
```

Python code:

```
import pandas as pd
x={"col1":[1,2,3],"col2":['Albert','Mary','Alan']}
y=pd.DataFrame(x)
x={"col1":[4,5,6],"col2":['John','Hilbert','David']}
z=pd.DataFrame(x)
a=pd.concat([y,z],axis=0)
print(a)
```



	col1	col2
0	1	Albert
1	2	Mary
2	3	Alan
0	4	John
1	5	Hilbert
2	6	David

If you want to supply the *value for* r through the **key board**, then the statement

```
float r = 2;
```

should be replaced by the statements

```
cout<<"Enter any number:";
cin>>r;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
float r, area;
cout<<"Enter any number:";
```

```
import pandas as pd
x={"col1":[1,2,3],"col2":['Albert','Mary','Alan']}
y=pd.DataFrame(x)
x={"col1":[4,5,6],"col2":['John','Hilbert','David']}
z=pd.DataFrame(x)
a=pd.concat([y,z],axis=1)
print(a)
```

Output:

	col1	col2	col1	col2
0	1	Albert	4	John
1	2	Mary	5	Hilbert
2	3	Alan	6	David

```
cin>>r;
area = 3.14 * r * r;
cout<<"The area of the circle = "<< area;
return 0;
}
```

```
x={"b1":11,"b2":12,"b3":13}
for a in x.keys():
   x[a]=x[a]+1
   print(x[a])

# Output:
                    12
                    13
                    14
```

The output on the screen:

```
Enter any number:
If you the number 2
The area of the circle = 12.56

will be outputted on the screen.
```

Desk check

An informal manual technique for testing the programming and logic of the algorithm before a program is launched

- **cout** is an output function and **cin** is an input function.

```
# Check if each word start with an upper case letter
x = "Albert Einstein"
print(x.istitle())

# Output: True
```

The statement:

```
cout<<"Enter any number:";
```

make provision to print the text

Enter any number:

on the screen.

```
for x in range(5):
  if x==2: break
  print(x)

# Output:
                0
                1
```

- **cin>> r;** is to C++ what **scanf("%d", &r);** is to C

```
for x in range(5):
  if x==2: continue
  print(x)
```

0
1
3
4

If you write the statement:

```
area = 3.14 * r ^ 2;
```

instead of

```
area = 3.14 * r * r;
```

Then ***compilation error*** will be displayed on the console screen because like in *C Language* – there is no operator for performing exponentiation operation − so the statement

```
area = 4 * 3.14 * r ^ 2;
```

is invalid.

```
if 5>3:
   print("Alan")
   print("Turing")
```

Output:

Alan
Turing

- cout and cin are not part of **C++ language** but they are part of **iostream file**. Hence the statement **#include<iostream>** should be included in the C++ program otherwise **cout** and **cin** will not work and the **compilation error** will be displayed on the console screen.
- **Right shift operator** >> denote **stream extraction operator** (extract data entered through the keyboard).
- **Left shift operator** << denote **stream insertion operator** (insert data into an output screen)
- << **and** >> are termed **overloaded operators** and the file iostream defines these operators.

As told earlier: when you enter an integer for x through the **keyboard**, this integer will be stored in the computer memory. If you yearn to know the ***storage size*** of the integer in **computer memory** (i.e., space occupied by the entered integer in the computer memory), you need to appeal to the following program:

```
#include<iostream>
using namespace std;
```

```
int main()
{
int x;
x=10;
cout<<"size of r = "<< sizeof(r);
return 0;
}
```

```
# returns the hash value of x
x=hash(6.7)
print(x)

names=['Albert','Alan','John','Mary','David']
x=' '.join(names)
print(x)

# Output: Albert Alan John Mary David
```

The output on the screen:

```
size of x = 4
```

i.e., integer entered for r i.e., 10 has occupied a space of 4 bytes in the computer memory.

C++ Type Qualifiers:

- **const:** Objects of type const cannot be changed by your program during execution.
- **volatile:** The modifier volatile tells the compiler that a variable's value may be changed in ways not explicitly specified by the program.
- **restrict:** A pointer qualified by restrict is initially the only means by which the object it points to can be accessed. Only C99 adds a new type qualifier called restrict.

- **Write a program to print the circumference of the circle (given r = 2.5)**

Answer:

```
#include<iostream>
using namespace std;
int main()
{
float r, area;
r = 2.5;
```

```
x=6.5%1.2
print(x)
# Output: 0.5000000000000002

x=2.2+3.3>5.5
print(x)
# Output: False
```

```
x={1,3,4,4}=={1,3,4}
print(x)
# Output: True

x=3^2
print(x)
# Output: 1
```

```
circumference = 2* 3.14 * r;
cout<<"The circumference of the circle = "<< circumference;
return 0;
}
```

- **Write a program to print the area of the rectangle (given l = 2.5 and b = 3)**

Answer:

```
#include<iostream>
using namespace std;
int main()
{
float l, b, area;
l = 2.5;
b = 3;
area = 1*b;
cout<<"The area of the rectangle = "<< area;
return 0;
}
```

```
x={11:11,12:14,13:19,14:18,15:28}
print(type(x))
# Output: <class 'dict'>

def myfunc():
    """ Albert """
    print("Einstein")
myfunc()
```

Output:

Einstein

- **Program 4.6**

C++ program to find the sum of two numbers

```
#include<iostream>
using namespace std;
int main()
{
int a, b, sum;
a=1;
b=2;
sum = a + b;
cout<<"the sum of a and b = "<< sum;
return 0;
```

```
print('z' in {'x':11,'y':12,'z':13}.values())
print(13 in {'x':11,'y':12,'z':13}.values())

# Output:
                                False
                                True
```

```
}
```

The output on the screen:

```
the sum of a and b = 3
```

If you assign the **floating point values** 1.5 and 2.6 *for* a and b, then the statement:

```
int a, b, sum;
```

should be replaced by the statement

```
float a, b, sum;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
float a, b, sum;
a=1.5;
b=2.6;
sum = a + b;
cout<<"the sum of a and b = "<< sum;
return 0;
}
```

```
print('albERt'.swapcase())
# Output: ALBerT

print(bytes([12,14,18]))
# Output: b'\x0c\x0e\x12'
```

```
# Infinite loop in python

def myfunc(x):
  while(x==7):print(x)
myfunc(5)
```

Press **Ctrl + C** to break an infinite loop

The output on the screen:

```
the sum of a and b = 4.1
```

The statement:

```
cout<<"the sum of a and b = "<< sum;
```

```
x="a"
y=5
print(x + "{0:.3f}".format(y))

# Output: a5.000
```

make provision to print the output:

```
the sum of a and b = 4.1
```

on the console screen. And if the **statement**:

```
cout<<"the sum of a and b = "<< sum;
```

```
x="a"
y=5
print("x={0} but y={1}".format(x,y))

# Output: x=a but y=5

a={x**3 for x in range(4,0,-2)}
print(a)

# Output: {64, 8}
```

is omitted from the **C ++ program**, then the program will be successfully executed but there will be *no display of the output* on the console screen.

If you want to supply the values ***for*** a and b through the key board, then the statements:

```
a=1.5;
b=2.6;
```

should be replaced by the statements:

```
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
float a, b, sum;
cout<<"Enter any two numbers:";
cin>>a;
```

```
x ="John"
print (x * 2)

# Output: JohnJohn

x ="John"
print (x + "Turing")

# Output: JohnTuring

x = [1905, 'Albert']
print (x * 2)

# Output: [1905, 'Albert', 1905, 'Albert']
```

```
cin>>b;
sum = a+ b;
cout<<"the sum of a and b = "<< sum;
return 0;
}
```

The output on the screen:

```
Enter any two numbers:
If you enter two numbers 2.9 & 3.6
the sum of a and b = 6.5
will be outputted on the screen.
```

```
x = (1927, 'George')
print (x * 2)

# Output: (1927, 'George', 1927, 'George')

print (3/ 2) # Output: 1.5
print (3 // 2) # Output: 1

import pandas as pd
x = pd.Series([11, 12, 13, 14, 15, 16, 17, 18, 19, 20])
y = pd.Series([20, 19, 18, 17, 16, 15, 14, 13, 12, 11])
print (sum((x - y)**2)**0.5)

# Output: 18.16590212458495
```

The statement:

```
cout<<"Enter any two numbers:";
```

make provision to print

Enter any two numbers:

on the screen and the statements:

```
cin>>a;
cin>>b;
```

make provision to read the two numbers 2.9 and 3.6 entered through the **keyboard** and store them in the computer memory.

If the statements:

```
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
```

```
def main():
    print('Albert Einstein')
if __name__ == '__main__':
    main()
```

Output:

Albert Einstein

are replaced by the statements:

```
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
```

Then the **output on the screen** is:

```
Enter any number:
If you enter the number 2.9
Enter any number:
If you enter the number 3.6
the sum of a and b = 6.5
will be outputted on the screen.
```

If the statement:

```
cout<<"the sum of a and b = "<< sum;
```

is replaced by the statement:

```
cout<< a <<" + "<< b <<" = "<< sum;
```

Then the **output**:

2.9 + 3.6 = 6.5

will be displayed on the console screen.

```
x=15
if x ==15:
    print("Albert")
else:
    pass

# Output: Albert

import json
x = json.dumps([11, 62,33, "albert", "beer"])
print(x)

# Output: [11, 62, 33, "albert", "beer"]

import pickle
x = pickle.dumps([11, 62,33, "albert", "beer"])
print(pickle.loads(x))

# Output: [11, 62, 33, 'albert', 'beer']
```

```
# prints the length of each string
a = ['Jo', 'hn']
print(len(list(map(list, a))))
```

Output:

2

```
def X(i):
    def Y():
        print(i)
    return Y

X(1905)()
```

Output:

1905

- **What will be the output of the following program:**

```
#include<iostream>
using namespace std;
int a = 5;
int main()
{
int a =2;
cout<< a;
return 0;
}
```

```
print(6.0 <= 6)
# Output: True

import numpy
print(numpy.array([]))
print(numpy.empty(shape=(0,0)))

# Output:
                        []
                        []
```

Answer:

2

- 2 is a **local variable** (variable declared within the body of the **main function**)

The statement:

```
int a = 2;
```

imply: **local variable** declaration.

- 5 is a **global variable** (variable declared outside the body of the main function)

The statement:

```
int a = 5;
```

imply: global variable declaration.

If the statement:

```
x = "Albert"
for i in x:
    print(i)

# Output:
                    A
                    l
                    b
                    e
                    r
                    t
```

```
cout<< a;
```

is replaced by the statement:

```
cout<< :: a; (where :: denote scope resolution operator)
```

i.e.,

```
#include<iostream>
using namespace std;
int a = 5;
int main()
{
int a =2;
cout<< ::a;
return 0;
}
```

```
def myfunc(n):
        x=1
        while(x<=n):
            yield x**3
            x+=1
for x in myfunc(3):
        print(x)
```

Output:

1
8
27

Then the **output on the screen** is:

5

i.e., **global variable** will be outputted on the screen.

```
x = [[11,12]] * 5
print(x)

# Output:
[[11, 12], [11, 12], [11, 12], [11, 12], [11, 12]]

x = [[11,12] for _ in range(3)]
print(x)

# Output:
        [[11, 12], [11, 12], [11, 12]]
```

If the same program is written in **C language**

i.e.,

```
#include<stdio.h>
int a = 5;
int main()
{
int a =2;
print("%d", ::a);
```

```
return 0;
}
```

Then the **compilation error** will be outputted on the screen because **scope resolution operator** is not defined in the C language (i.e., **C does not hold scope resolution operator**).

```
x = open('1.txt')
for i in x:
    i = i.strip('\n')
    print(i)

Output:

Albert Einstein was a theoretical physicist who was born in Germany and is largely
regarded as one of the greatest and most influential physicists of all time.
```

Content of 1.txt

- **Whether the following program will be successfully outputted or not:**

```
#include<iostream>
using namespace std;
int main()
{
int a, b, c;
a=3;
b=2;
c= a+b;
cout<<" sum of two numbers = 6"<< c;
return 0;
}
```

```
x = [[11,12,13,14]]
x[0][3] = 59
print(x)

# Output: [[11, 12, 13, 59]]

x = []
for i in range(4):
    x.append(lambda: i)
print ([y() for y in x])

# Output: [3, 3, 3, 3]
```

Answer:

Yes, the output on the screen is:

```
sum of two numbers = 65
```

Executable

The binary file that is formed when the source code is built

- **Program 4.7**

  C ++ program to convert the temperature in Celsius to Fahrenheit

```
#include<iostream>
using namespace std;
int main()
{
float C, F;
C=38.5;
F = 9*C/5 +32;
cout<<"temperature in Fahrenheit= "<< F;
return 0;
}
```

```
x = []
for i in range(4):
    x.append(lambda i = i : i)
print ([y() for y in x])

# Output: [0, 1, 2, 3]


print([ [ ] ] * 3)

# Output: [[], [], []]
```

The output on the screen:

```
temperature in Fahrenheit = 101.3
```

If you want to supply a value **16 digits** after decimal point i.e., 36.5555555555555555 for C, then the statement:

```
double C, F;
```

should be used instead of the statement:

```
float C, F;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
double C, F;
C=38.5555555555555555;
F = 9*C/5 +32;
cout<<"temperature in Fahrenheit= "<< F;
return 0;
}
```

```
x = ['a', 'l', 'a', 'n']
print (x[2:])
print (x[7:])

# Output:
            ['a', 'n']
            []
```

If you want to supply the value for C through the key board, then the statement:

```
C=38.5;
```

should be replaced by the statements:

```
cout<<"Enter any number:";
cin>>C;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
float C, F;
cout<<"Enter any number:";
cin>>C;
F = 9*C/5 +32;
cout<<"temperature in Fahrenheit= "<< F;
return 0;
```

```
for x in '1905':
 print ("albert",x,)

# Output:
                  albert 1
                  albert 9
                  albert 0
                  albert 5

x = {'Albert': 1905}
x.update({"Tesla":1927})
print(x)

# Output: {'Albert': 1905, 'Tesla': 1927}
```

```
}
```

The output on the screen:

```
Enter any number:

If you enter the number 23.6
temperature in Fahrenheit = 74.48

will be outputted on the screen.
```

```
import array
x = array.array('i', [50,25,15])
print(x[1])
```

Output:

25

```
x = open('1.txt')
y = x.readlines()
type(y)
print(y)

Output:

    ['Albert Einstein was a theoretical physicist who was born in Germany and is largely
       regarded as one of the greatest and most influential physicists of all time.']
```

```
import array
x = array.array('i', [50,25,15])
x.insert(3, 5)
print(x)

# Output: array('i', [50, 25, 15, 5])
```

- **Program 4.8**

C++ program to find the product of two numbers

```
import array as myarray
x = myarray.array('i', [500, 300, 200])
x.remove(300)
print(x)

# Output: array('i', [500, 200])
```

```
#include<iostream>
using namespace std;
int main()
```

```
{
int a, b, product;
a=1;
b=2;
product = a * b;
cout<<"the product of a and b = "<< product;
return 0;
}
```

```
import array as myarray
x = myarray.array('i', [12,22, 32])
x.reverse()
print(x)

# Output: array('i', [32, 22, 12])
```

The output on the screen:

```
the product of a and b = 2
```

If you want to insert a **10 digit number** ***for*** a and b i.e.,

```
a=1000000000
b=3000000000
```

```
x = "Albert Einstein"
y = "Alan Turing"
print (x[1])
print (y[1:4])

# Output:
                    l
                   lan
```

, then the statement:

```
int a, b, product;
```

should be replaced by the statement:

```
long int a, b, product;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
long int a, b, product;
a=1;
b=2;
```

```
import numpy as np
x = np.array([[15, -50, 33], [51, -56, 65], [-42, 79, 82]])
print(x)
```

```
[[ 15 -50  33]
 [ 51 -56  65]
 [-42  79  82]]
```

```
product = a * b;
cout<<"the product of a and b = "<< product;
return 0;
}
```

Package declaration

A declaration that is used to identify a package

The output on the screen:

```
the product of a and b = 3000000000000000000
```

If you want to supply the **integer values *for*** a and b through the key board, then the statements:

a=1;

b=2; should be replaced by the statements:

```
cout<<"Enter any two numbers:";
cin >> a;
cin >> b;
```

```
import numpy as np
x = np.array([[13, 16, 19], [15, -20, 25], [-17, 24, 56]])
y = np.array([[91, -15, 87], [10, 28, 53], [63, -18, 38]])
z = x - y
print(z)
```

```
[[-78  31 -68]
 [  5 -48 -28]
 [-80  42  18]]
```

i.e.,

```
#include<iostream>
using namespace std;
int main ()
{
int a,  b,  product;
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
product = a* b;
cout<<"the product of a and b = "<< product;
return 0;
}
```

```
import numpy as np
x = np.array([[13, 16, 19], [15, -20, 25], [-17, 24, 56]])
y = np.array([[91, -15, 87], [10, 28, 53], [63, -18, 38]])
z = x.dot(y)
print(z)
```

```
[[ 2540   -89  2701]
 [ 2740 -1235  1195]
 [ 2221   -81  1921]]
```

The output on the screen:

```
Enter any two numbers:
If you enter two numbers 2 & 3
the product of a and b = 6
will be outputted on the screen
```

```
x = [12,13,14,13,20,13,15,16,13]
y = x.count(13)
print(y)
```

Output:

4

The numeric expressions are composed of the following terms:

- Numbers
- Operators +, −, /, *,^,%, =
- Parentheses
- Variables

- **Program 4.9**

C++ program to find the square of a number

```
#include<iostream>
using namespace std;
int main()
{
int a, b;
a=2;
b = a * a;
cout<<"the square of a = "<< b;
return 0;
}
```

```
for x in range(0, 5):
    print('*', end="")

# Output: *****

print(help('modules'))
```

Prints all available modules

The output on the screen:

```
the square of a = 4
```

If you want to supply the integer value for a through the **key board**, then the statement:

```
a=2;
```

should be replaced by the statements:

```
cout<<"Enter any number:";
cin>>a;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
int a, b;
cout<<"Enter any number:";
cin>>a;
b = a * a;
cout<<"the square of a = "<< b;
return 0;
}
```

The output on the screen:

```
Enter any number:
If you enter a number 3
the square of a = 9
will be outputted on the screen.
```

```
import math
print(math.pi)
math.pi = 15
print(math.pi)
```

Output:

3.141592653589793

15

```
x = ['Car', 'House', 'Phone', 'Computer']
print(x[-1][-1])

# Output: r

y = [15, 20, 25, 30]
for i, x in enumerate(y, start=10):
   print(i, x)

# Output:
                    10 15
                    11 20
                    12 25
                    13 30
```

- **Write a program to print the cube of a number**

Answer:

```
#include<iostream>
using namespace std;
int main()
{
int a, b;
cout<<"Enter any number:";
cin>>a;
b = a * a*a;
cout<<"the cube of a = "<< b;
return 0;
}
```

```
print(sum([x * x for x in range(1, 10)]))

# Output: 285


y = {'age': 102, 'name': 'Mary', 'Car': 'Audi'}
x = y.get('age')
print(x)

# Output: 102
```

- **Write a program to print the force applied to the mass m.**

Answer:

```
#include<iostream>
using namespace std;
int main()
{
int m, a, F;
cout<<"Enter the mass:";
cin>>m;
cout<<"Enter acceleration:";
cin>>a;
F = m * a;
cout<<"the force applied to the mass = "<< F;
return 0;
}
```

```
y = {'age': 102, 'name': 'Mary', 'Car': 'Audi'}
x = y.setdefault('name')
print(x)

# Output: Mary
```

```
def x(name, money):
   print(f"{name} had a net worth equal to ${money / 10:.5f} thousand at the time of his death in 1955.")
x("Albert Einstein", 650)
```

Output: Albert Einstein had a net worth equal to $65.00000 thousand at the time of his death in 1955.

```
x = open('1.txt')
y = x.readline()
while y!= "":
    print(y)
    y = x.readline()
```

Output:

Albert Einstein was a theoretical physicist who was born in Germany and is largely regarded as one of the greatest and most influential physicists of all time.

```
from collections import Counter
x = "Albert Einstein was born \
and raised in Munich".split()
y = Counter(x)
print(y)
```

Output:

```
Counter({'Albert': 1, 'Einstein': 1, 'was': 1, 'born': 1, 'and': 1, 'raised': 1, 'in': 1, 'Munich': 1})
```

- **Program 5.0**

  C ++ program to find the greatest of two numbers using ***if - else*** statement

  The syntax of *if - else* statement is:

```
if (this condition is true)
{
print this statement;
}
else
{
print this statement;
```

```
y = [16, 12, 13, 11]
i=1;
for x in y:
  i = i* x
print (i)

# Output: 27456
# 16 × 12 × 13 × 11 = 27456
```

```
}
```

```
#include<iostream>
using namespace std;
int main()
{
int a, b;
a = 2;
b = 3;
if(a>b)
{
cout<<"a is greater than b";
}
else
{
cout<<"b is greater than a";
}
return 0;
}
```

```
y = [22, 26, 66, 46, 46, 70, 46, 52]
x = []
[x.append(i) for i in y if i not in x]
print (x)
```

Output:

```
[22, 26, 66, 46, 70, 52]
```

The output on the screen:

```
b is greater than a
```

- **Program 5.1**

  C++ program to find the greatest of three numbers using ***else-if*** statement

  The syntax of *else-if* statement:

```
if (this condition is true)
{
print this statement;
}
else if (this condition is true)
```

```
y = [24, 28, 13, 20, 15, 16]
x = [i for i in y if i%2 != 0]
print(x)

# Output: [13, 15]
```

```
{
print this statement;
}
else
{
print this statement;
}
```

```
y = [24, 28, 13, 20, 15, 16]
x = [i for i in y if i%2 == 0]
print(x)

# Output: [24, 28, 20, 16]
```

```
x = [11, 12, 13, 14, 15, 16]
x.sort(reverse=True)
print(x)
```

Output:

[16, 15, 14, 13, 12, 11]

```
#include<iostream>
using namespace std;
int main()
{
int a, b, c;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
cout<<"Enter any number:";
cin>>c;
if(a>b&&a>c)
{
cout<< a<<" is greater than"<< b<<" and "<<c;
}
else if (b>a&&b>c)
{
cout<< b<<" is greater than"<< a <<" and "<<c;
}
else
{
cout<< c<<" is greater than"<< b<<" and "<< a;
}

return 0;
}
```

```
x = ["Haw", "w", "a", "physi"]
y= ["king", "as", "n", "cist"]
z = [a + b for a, b in zip(x, y)]
print(z)
```

Output:

['Hawking', 'was', 'an', 'physicist']

The output on the screen:

```
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 4
4 is greater than 3 and 2 will be outputted on the screen.
```

- **What will be the output of the following program?**

```
#include <iostream>
using namespace std;
int main()
{
int a, b;
a=2;
b=2;
if(a>b || a==b)
cout<<"a is greater than or equal to b";
else
cout<<"b is greater than a";
return 0;
}
```

```
x = ["Haw", "w", "a", "physi"]
y= ["king", "as", "n", "cist"]
z = [(a, b) for a in x for b in y]
c = [a + ' ' + b for (a, b) in z]
print(c)
```

Output:

['Haw king', 'Haw as', 'Haw n', 'Haw cist', 'w king', 'w as', 'w n', 'w cist', 'a king', 'a as', 'a n', 'a cist', 'physi king', 'physi as', 'physi n', 'physi cist']

Answer:

```
a is greater than or equal to b
```

- Symbol || denote OR i.e., **a>b || a == b** denote ***a is greater than or a is equal to b***.

- **Program 5.2**

  C ++ program to find the average of 10 numbers

```
#include<iostream>
using namespace std;
int main()
{
int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, X;
cout<<"Enter any 10 numbers:";
cin>>N1;
cin>>N2;
cin>>N3;
cin>>N4;
cin>>N5;
cin>>N6;
cin>>N7;
cin>>N8;
cin>>N9;
cin>>N10;
X = (N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9 + N10) /10;
cout<<"the average of 10 numbers = "<< X;
return 0;
}
```

```
x = [-15, 57, 500, -41, -20, -810, 266, -97]
y = [x for x in x if x >= 0]
print(y)
```

Output:

```
[57, 500, 266]
```

The output on the screen:

```
Enter any 10 numbers:
If you enter ten numbers 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10
the average of 10 numbers = 5
will be outputted on the screen.
```

C++ equivalent *mathematical expression* is same as C equivalent *mathematical expression*

For example:

Mathematical expression	C equivalent expression	C++ equivalent expression
$log_{10}x + bx$	log10 (x) + b * x	log10 (x) + b * x

- **Program 5.3**

  C ++ program to find the square root of a number

```
x, y = 12, 20
print(x if x < y else y)

# Output: 12

x=12
x,y=x+1,x+2
print(x,y)

# Output: 13 14
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main()
{
int a, b;
cout<<"Enter any number:";
cin>> a;
b = sqrt (a);
cout<<"the square root of a number = "<< b;
return 0;
}
```

The output on the screen:

```
x = []
for i in range(5):
  x.append(i*2)
print (x)

# Output: [0, 2, 4, 6, 8]
```

```
Enter any number:
If you enter the number 16
the square root of a number = 4
will be outputted on the screen.
```

```
Since b = sqrt(a) is written:
```

the statement: **#include<cmath>** must be included in the above program because **cmath file** defines the mathematical functions like sqrt().

If the statement: **#include<cmath>** is not included in the above program:

```
#include<iostream>
using namespace std;
int main()
{
int a, b;
cout<<"Enter any number:";
cin>> a;
b = sqrt (a);
cout<<"the square root of a number = "<< b;
return 0;
}
```

```
def upto(x):
  for a in range(x-1):
    yield a
for i in upto(5):
  print(i)
```

Output:

0
1
2
3

Then the ***compilation error*** will be displayed on the console screen.

Note:

#include<math.h> is used in C

whereas #include<cmath> is used in C ++

```
x = {11:12, 13:14}
y = {**x}
print(y)

# Output: {11: 12, 13: 14}
```

- **Write a program to print the cube root of a number:**

Answer:

```
#include<iostream>
#include<cmath>
```

```
x = {11:12, 13:14}
y = {13:15, 15:16}
z = {**x, **y}
print(z)

# Output: {11: 12, 13: 15, 15: 16}
```

```
using namespace std;
int main()
{
cout<<"the cube root of a number = "<< cbrt (8);
return 0;
}
```

- **Program 5.4**

  C++ program to find the simple interest

```
#include<iostream>
using namespace std;
int main()
{
int P,T, R, SI;
P = 1000;
T = 2;
R = 3;
SI = P*T*R/100;
cout<<"the simple interest = "<< SI;
return 0;
}
```

```
x = []
y = []
z = x
print(x == y)
print(x is y)
print(x is z)
```

Output:

True
False
True

The output on the screen:

```
the simple interest = 60
```

If you want to supply the **integer values** *for* P, T and R through the **key board**, then the statements:

```
P = 1000;
```

```
T = 2;
R = 3;
```

should be replaced by the statements:

```
print(any([False, False, False]))
# Output: False

print(any([True, False, False]))
# Output: True

print(any([True, True, True]))
# Output: True
```

```
cout<<"Enter principal amount:";
cin>>P;
cout<<"Enter time:";
cin>>T;
cout<<"Enter rate of interest:";
cin>>R;
```

i.e., the above program should take the form:

```
x= [15, 52, 53]
y = x
y[0] = 50
print(y)

# Output: [50, 52, 53]


names = ["Alan", "Albert", "Mary"]
for x in range(len(names)):
    y = names[x]
    print("{} is at {}. place".format(y, x + 1))

# Output:
                 Alan is at 1. place
                Albert is at 2. place
                 Mary is at 3. place
```

```
#include<iostream>
using namespace std;
int main()
{
int P,T, R, SI;
cout<<"Enter principal amount:";
cin>>P;
cout<<"Enter time:";
cin>>T;
cout<<"Enter rate of interest:";
cin>>R;
SI = P*T*R/100;
cout<<"the simple interest = "<<SI;
return 0;
}
```

The output on the screen:

```
Enter principal amount:
```

```
If you enter the principal amount 1000
Enter time:
If you enter the time 2
Enter rate of interest:
If you enter the rate of interest 3
the simple interest = 60
will be outputted on the screen.
```

```
try:
    x = 51 + "three"
except:
    print("An exception has been discovered! The
default value will be used.")
    x = 25
finally:
    print("This is always the case.")
print(x)
```

```
An exception has been discovered! The
default value will be used.
This is always the case.
25
```

- **Program 5.5**

  C++ program to find the senior citizen

```
#include<iostream>
using namespace std;
int main()
{
int age;
age=20;
if(age > = 60)
{
cout<<"senior citizen";
}
if(age<60)
{
cout<<"not a senior citizen";
}
return 0;
}
```

```
from itertools import permutations
x = [11, 12, 13]
y = permutations(x)
print(list(y))
```

```
[(11, 12, 13), (11, 13, 12), (12, 11, 13), (12, 13, 11), (13, 11, 12), (13, 12, 11)]
```

```
x = {"John": 42, "Joseph": 20, "James": 10}
print("James" in x)

# Output: True
```

The output on the screen:

```
not a senior citizen
```

- (age > = 60) **means:** **age greater than or equal to 60.**

If you want to supply the value for age through the key board, then the statement:

```
age = 20;
```

should be replaced by the statements:

```
cout<<"Enter age:";
cin>>age;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
int age;
cout<<"Enter age:";
cin>>age;
if(age>60)
{
cout<<"senior citizen";
}
if(age<60)
{
cout<<"not a senior citizen";
}
return 0;
}
```

Python code:

```
string = "John"
list = ["Albert", "John", "Mary", "David", "Joseph"]
tuple = ("Einstein", "Hilbert", "Newton")
dictionary = {
    "David": "Hilbert",
    "Stephen": "Hawking",
    "Isaac": "Newton",
    "Niels": "Bohr",
}

print("The maximum value of string is: ", max(string))
print("The maximum value of list is: ", max(list))
print("The maximum value of tuple is: ", max(tuple))
print("The maximum value of dictionary is: ", max(dictionary))
```

Output:

The maximum value of string is: o

The maximum value of list is: Mary

The maximum value of tuple is: Newton

The maximum value of dictionary is: Stephen

The output on the screen:

```
Enter age:

If you enter the age 60

senior citizen
```

Abstract method

A method whose declaration does not need implementation

will be outputted on the screen.

Suppose if you enter the age 31

```
not a senior citizen
```

will be outputted on the screen.

Default constructor

A constructor that can be invoked without specifying any parameters

Information hiding

The idea that development decisions should be concealed by the modules in which they are used

- **Program 5.6**

  C ++ program to get marks for 3 subjects and declare the result.

  If the marks >= 35 in all the subjects the student passes else fails.

```
#include<iostream>
using namespace std;
int main()
{
int M1, M2, M3;
M1 = 38;
M2= 45;
M3 = 67;
if(M1 >= 35 && M2>= 35 && M3>= 35)
{
cout<<"candidate is passed";
}
else
{
cout<<"candidate is failed";
}
return 0;
}
```

Python code:

```
import array as arr
x=arr.array('i',[11,12,13,14,15])
print (x[::-1])
```

Output:

array('i', [15, 14, 13, 12, 11])

```
from random import shuffle
a = ['Albert', 'Einstein', 'is', 'a', ' theoretical', 'physicist']
shuffle(a)
print(a)
```

Output:

[' theoretical', 'Einstein', 'is', 'a', 'Albert', 'physicist']

The output on the screen:

```
candidate is passed
```

If you want to supply the integer values ***for marks*** M1, M2 and M3 through the **key board**, then the statements:

```
M1 = 38;
M2= 45;
M3 = 67;
```

should be replaced by the statements:

```
cout<<"Enter any three marks:";
cin>> M1;
cin>> M2;
cin>> M3;
```

i.e.,

```
#include<iostream>
using namespace std;

int main()
{
int M1, M2, M3;
cout<<"Enter any three marks:";
cin>> M1;
cin>> M2;
cin>> M3;
if(M1 >= 35 && M2>= 35 && M3>= 35)
{
cout<<"candidate is passed";
}
else
{
```

```
x='ALBERT'
print(x.lower())
# Output: albert

x='ALBERT'
print(len(x))
# Output: 4

x="Albert Einstein"
print(x.split())
# Output: ['Albert', 'Einstein']

def myfunc(x):
    for i in range(x):
        print(''*(x-i-1)+'*'*(2*i+1))
myfunc(6)

# Output:
     *
    ***
   *****
  *******
 *********
***********

x = 13
if x > 1:
    for a in range(2, int(x/2)+1):
        if (x % a) == 0:
            print(x, "is not a prime number")
            break
    else:
        print(x, "is a prime number")
else:
    print(x, "is not a prime number")

# Output: 13 is a prime number
```

```
cout<<"candidate is failed";
}
return 0;
}
```

The output on the screen:

```
Enter any three marks:
26, 28, 39 # entered marks
candidate is failed
will be outputted on the screen.
```

- **Program 5.7**

  C ++ program to find profit or loss

```
#include<iostream>
using namespace std;
int main()
{
int CP, SP, loss, profit;
cout<<"Enter cost price:";
cin >> CP;
cout<<"Enter selling price:";
cin>>SP;
if ( SP > CP )
{
cout<<"profit= "<< SP-CP;
}
else
{
cout<<"loss = "<< CP-SP;
}
return 0;
```

```
x='535'
y=x[::-1]
if x==y:
  print("palindrome")
else:
  print("Not a Palindrome")

# Output: palindrome

x = ["12", "42", "50", "16", "19"]
x = [int(a) for a in x]
x.sort()
print (x)

# Output: [12, 16, 19, 42, 50]

x = dict(zip(('a','b','c','d','e'),(11,12,13,14,15)))
print(x)
# Output: {'a': 11, 'b': 12, 'c': 13, 'd': 14, 'e': 15}


x = range(3)
y = [[i, i*i] for i in x]
print(y)

# Output: [[0, 0], [1, 1], [2, 4]]

import numpy as np
x = np.array([1, 3, 2, 4, 5])
print(x.argsort()[-3:][::-1])

# Output: [4 3 1]

x = 'Albert'
y = "Albert"
z = '''''Albert'''
print(x)
print(y)
print(z)
```

Output:

Albert

Albert

"Albert

```
}
```

The output on the screen:

```
Enter cost price:
If you enter the cost price 25
Enter selling price:
If you enter the selling price 26
profit = 1
will be outputted on the screen.
```

```
x = True + 5
y = False + 8
print(x)
print(y)

# Output:
              6
              8

x = [ 6, 11, 8, 2, 10, 6 ]
print(x)
x.remove(6)
print(x)

# Output:
             [6, 11, 8, 2, 10, 6]
              [11, 8, 2, 10, 6]
```

- **Program 5.8**

  C++ program to convert inches into centimeter

```
#include<iostream>
using namespace std;
int main()
{
float I, C;
I=3.5;
C = 2.54*I;
cout<<"length in centimeters = "<< C;
return 0;
}
```

```
x = [ 3, 11, 8, 2, 10, 3 ]
print(x)
del x[3]
print(x)

# Output:
             [3, 11, 8, 2, 10, 3]
              [3, 11, 8, 10, 3]

x = "lber"
y = "At"
print(x.join(y))

# Output: Albert
```

The output on the screen:

```
length in centimeters = 8.89
```

```
# Save an jpg image locally from an URL address

import urllib.request
urllib.request.urlretrieve("https://parade.com/wp-content/uploads/2021/08/albert-einstein-quotes.jpg", "img1.jpg")
```

- **Note:** float is used instead of int because **I = 3.5** if int is used instead of float then the result will not be clearly outputted i.e., instead of 8.89 the computer displays only 8.

If you want to supply the value *for* I through the **key board**, then the above program should take the form:

```
#include<iostream>
using namespace std;
int main()
{
float I, C;
cout<<"Enter the length in inches:";
cin >> I;
C = 2.54*I;
cout<<"length in centimeters= "<< C;
return 0;
}
```

```
x = 10
y = 14
z = x if x < y else y
print(z)

# Output: 10

x, y = 15, 15
print(x==y)
print(x<y)
print(x<=y)
print(x!=y)

# Output:
```

True
False
True
False

The output on the screen:

```
Enter the length in inches:
```

If you enter the value for I i.e., 25.5

```
length in centimeters = 64.9 will be outputted on the screen.
```

- **Program 5.9**

  C++ program to find the incremented and decremented values of two numbers

```
#include<iostream>
using namespace std;
int main()
```

Constructor

↓

A class function that creates new objects within the class

```
{
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
cout<<"the incremented value of a = "<< c;
cout<<"the incremented value of b = "<< d;
cout<<"the decremented value of a = "<< e;
cout<<"the decremented value of b = "<< f;
return 0;
}
```

```
x=13
print(x)
x += 3
print(x)
x -= 3
print(x)
x *= 3
print(x)
x **=3
print(x)


x = True
y = False
print(x and y)

# Output: False
```

Output:

13
16
13
39
59319

The output on the screen:

```
the incremented value of a = 11 the incremented value of b = 13 the decremented value
of a = 9 the decremented value of b = 11
```

If the statements:

```
cout<<"the incremented value of a = "<< c;
cout<<"the incremented value of b = "<< d;
cout<<"the decremented value of a = "<< e;
cout<<"the decremented value of b = "<< f;
```

```
x = True
y = False
print(x or y)

# Output: True

x = True
y = False
print(not y)

# Output: True
```

are replaced by the statements:

```
cout<<"\n the incremented value of a = "<< c;
cout<<"\n the incremented value of b = "<< d;
cout<<"\n the decremented value of a = "<< e;
cout<<"\n the decremented value of b = "<< f;
```

Then the **output on the screen** is:

```
the incremented value of a = 11
the incremented value of b = 13
the decremented value of a = 9
the decremented value of b = 11
```

If the statements:

```
cout<<"the incremented value of a = "<< c;
cout<<"the incremented value of b = "<< d;
cout<<"the decremented value of a = "<< e;
cout<<"the decremented value of b = "<< f;
```

are replaced by the statements:

```
cout<<"the incremented value of a = "<< c << endl;
cout<<"the incremented value of b = "<< d << endl;
cout<<"the decremented value of a = "<< e << endl;
cout<<"the decremented value of b = "<< f << endl;
```

Then the **output on the screen**:

```
the incremented value of a = 11
the incremented value of b = 13
the decremented value of a = 9
the decremented value of b = 11
```

```
x = [22,24,26,27,33,34]
print(35 in x)

# Output: False

x = ("Alan","John")
print("Albert" not in x)

# Output: True

x = 50
y = 24
print(x is y)

# Output: False

x = 50
y = 24
print(x is not y)

# Output: True


x = 60
y = 74
print(x & y)
print(x | y)
print(x ^ y)
print(~x)
```

Output:

8
126
118
-61

If you want to supply the values *for* a ***and*** b through the **key board**, then the above program should take the form:

```
#include<iostream>
```

Data sink

A storage device that collects and stores data for an unlimited amount of time

```
using namespace std;
int main()
{
int a, b, c, d, e, f;
cout<<"Enter any number:";
cin>> a;
cout<<"Enter any number:";
cin>> b;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
cout<<"\n the incremented value of a = "<< c;
cout<<"\n the incremented value of b = "<< d;
cout<<"\n the decremented value of a = "<< e;
cout<<"\n the decremented value of b = "<< f;
return 0;
}
```

```
x = ("\u0123", "\u2665", "\U0001f638", "\u265E", "\u265F", "\u2168")
print (x)

# Output: ('ģ', '♥', '😸', '♞', '♟', 'Ⅸ')
```

The output on the screen:

```
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3

the incremented value of a = 3
the incremented value of b = 4
the decremented value of a = 1
the decremented value of b = 2

will be outputted on the screen.
```

```
x= "Albert Einstein profoundly changed physics"
print(x[3:15])

# Output: ert Einstein
```

```
with open("1.txt", "r") as fp:
    x = fp.read()
# prints the contents of the 1.txt file
print(x)
```

```
x = ["A","L","B","E","R","T"]
print (list(enumerate(x)))

# Output:

  [(0, 'A'), (1, 'L'), (2, 'B'), (3, 'E'), (4, 'R'), (5, 'T')]
```

- **What will be the output of the following program:**

```
#include<iostream>
using namespace std;
int main()
{
float T1, T2, A;
cout<<"Enter any number:";
cin >>T1;
cout<<"Enter any number:";
cin >>T2;
A = (T1 + T2) / 2;
cout<<"the average temperature of the day = "<< A;
return 0;
}
```

```
import array as arr
x = [2, 'pqr', 2.55]
print (x)

# Output: [2, 'pqr', 2.55]
```

Answer:

```
Enter any number:
If you enter the number:
2
Enter any number:
If you enter the number:
3
the average temperature of the day = 2.5
will be displayed on the console screen.
```

```
import os
os.remove('hello.txt')


import os
os.unlink('hello.txt')
```

Python code to delete a file

- **Program 6.0**

  The percentage marks are entered and the grades are allotted as follows:

  percentage >= 60 First Class

  percentage >=50 and per <= 60 Second Class

  percentage >= 40 and per <= 50 Pass Class

  percentage < 40 Fail

```
x, y = 10, 30
print(x, y)
x, y = y, x
print(x, y)

# Output:
                10 30
                30 10
```

Write a C++ program for the above:

Mutator

A function that modifies the state of an object

```
#include<iostream>
using namespace std;
main()
{
int P;
cout<<"Enter the percentage:";
cin>>P;
if(P >= 60)
{
cout<<"first class";
}
if(P>=50&&P <60)
{
cout<<"second class";
}
if(P>=40&&P<=50 )
{
cout<<"pass class";
}
if(P<40)
{
cout<<"fail";
}
return 0;
}
```

The output on the screen:

```
Enter the percentage:
If you enter the percentage 35
fail
will be outputted on the screen.
```

```
import array as arr
x=arr.array('b',[11, 12, 13, 14, 15])
print(x[::-1])

import array
x = array.array('b', [11, 12, 13, 14, 15])
print(x[::-1])


from array import *
x =array('b',[11, 12, 13, 14, 15])
print(x[::-1])
```

Output:

array('b', [15, 14, 13, 12, 11])

```
def myfunc():
   return 14, 15, 16, 17
a, b, c, d = myfunc()
print(a, b, c, d)

# Output: 14 15 16 17

i = 4
z = "Albert ";
print(z * i)
```

Output:

Albert Albert Albert Albert

- **Program 6.1**

  C++ program to calculate the discounted price and the total price after discount

  Given:

  - If purchase value is greater than 1000, 10% discount
  - If purchase value is greater than 5000, 20% discount
  - If purchase value is greater than 10000, 30% discount

- **discounted price**

```
#include<iostream>
using namespace std;
int main()
{
double PV, dis;
cout<<"Enter purchased value:";
cin>>PV;
if(PV>1000)
{
cout<<"dis= "<< PV* 0.1;
}
else if(PV>5000)
{
cout<<"dis= "<< PV* 0.2;
}
else
{
cout<<"dis= "<< PV* 0.3;
}
return 0;
}
```

```
# Check the memory usage of 'a'
import sys
a = 200
print(sys.getsizeof(a))

# Output: 28

x = [11, 12, 13, 19, 12, 17, 13, 15, 19, 19, 19]
print(max(set(x), key = x.count))

# Output: 19

from collections import Counter
def myfunc(a, b):
     return Counter(a) == Counter(b)
print(myfunc('Alan', 'Albert'))
print(myfunc('Alan', 'Alan'))
```

Output:

False

True

The output on the screen:

```
Enter purchased value:
If you enter the purchased value 6500
dis = 1300.000000
will be outputted on the screen.
```

Accessor

A function that accesses information in an object without causing any observable changes to the object

- **total price**

```
#include<iostream>
using namespace std;
int main()
{
double PV, total;
cout<<"Enter purchased value:";
scanf("%lf", &PV;
if(PV<1000)
{
cout<<"total= "<< PV - PV* 0.1;
}
else if(PV<5000)
{
cout<<"total = "<< PV- PV* 0.2;
}
else
{
cout<<"total= "<< PV- PV* 0.3;
}
return 0;
}
```

```
import pandas as pd
names=["Alan","John","Albert","Joseph","Mary"]
jobs=["Chemist","Biologist","Physicist","Engineer","Teacher"]
y={"jobs":jobs,"names":names}
x=pd.DataFrame(y)
print(x)
```

Output:

	jobs	names
0	Chemist	Alan
1	Biologist	John
2	Physicist	Albert
3	Engineer	Joseph
4	Teacher	Mary

```
a=1
x=2
while a<=5:
    print(x,"*", a, "=", x*a)
    a=a+1
```

2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10

The output on the screen:

```
Enter purchased value:
If you enter the purchased value 650
```

```
total = 585.000000
will be outputted on the screen.
```

- **Now, Combing both the programs (above), we can write:**

```
#include<iostream>
using namespace std;
int main()
{
double PV, dis, total;
cout<<"Enter purchased value:";
cin>>PV;
if(PV>1000)
{
cout<<"dis= "<< PV* 0.1;
cout<<"total= "<< PV - PV* 0.1;
}
else if(PV>5000)
{
cout<<"dis = "<< PV* 0.2;
cout<<"total= "<< PV - PV* 0.2;
}
else
{
cout<<"dis= "<< PV* 0.3;
cout<<"total= "<< PV - PV* 0.3;
}
return 0;
}
```

The output on the screen:

```
Enter purchased value:
```

```
import pandas as pd
names=["Alan","John","Albert","Joseph","Mary"]
jobs=["Chemist","Biologist","Physicist","Engineer","Teacher"]
y={"jobs":jobs,"names":names}
x=pd.DataFrame(y)
q=[1,2,3,4,5]
x.index=q
print(x.loc[4])
```

Output:

jobs	Engineer
names	Joseph

Name: 4, dtype: object

```
import pandas as pd
x=[11,12,13]
y=[21,31,51]
a={"num1":x,"num2":y}
z=pd.DataFrame(a)
z["Sum"]=z["num1"]+z["num2"]
z["Difference"]=z["num1"]-z["num2"]
print(z)
```

Output:

	num1	num2	Sum	Difference
0	11	21	32	-10
1	12	31	43	-19
2	13	51	64	-38

```
If you enter the purchased value 850

dis = 85.000000

total = 765.000000

will be outputted on the screen.
```

```
import pandas as pd
x={"col1":[1,2,3],"col2":["John","Alan","Mary"]}
y=pd.DataFrame(x)
print(y)
```

	col1	col2
0	1	John
1	2	Alan
2	3	Mary

- **Program 6.2**

  C++ program to print the first ten natural numbers using *for* loop statement

```
#include<iostream>
using namespace std;
int main()
{
int i;
for (i=1; i<=10; i++)
cout<<"value of i = "<< i;
return 0;
}
```

```
import pandas as pd
x={"col1":[1,2,3],"col2":["John","Alan","Mary"]}
y=pd.DataFrame(x)
y=y.drop(["col1"],axis=1)
print(y)
```

	col2
0	John
1	Alan
2	Mary

The output on the screen is:

```
value of i = 1 value of i = 2 value of i = 3 value of i = 4  value of i= 5  value of
i= 6 value of i = 7 value of i= 8  value of i= 9  value of i= 10
```

- **When *for loop* executes, the following occurs:**

```
for x in range(4):
    for a in range(x):
        print(x, end=" ")
    print("\n")
```

```
1

2 2

3 3 3
```

```
i = 1
Is the condition (i<=10) is true?
Yes because i=1
The statement cout<<"value of i = "<< i; is executed to print the output:
```

Data corruption

A breach of data security

```
value of i = 1
Now, the value of i is:
i = 1+1 = 2
Is the condition (i<=10) is true?
Yes because i=2
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 2
Now, the value of i is:
i = 2+1 = 3
Is the condition (i<=10) is true?
Yes because i=3
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 3
Now, the value of i is:
i = 3+1 = 4
Is the condition (i<=10) is true?
Yes because i=4
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 4
Now, the value of i is:
i = 4+1 = 5
Is the condition (i<=10) is true?
Yes because i=5
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 5
Now, the value of i is:
i = 5+1 = 6
Is the condition (i<=10) is true?
Yes because i=6
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 6
Now, the value of i is:
i = 6+1 = 7
Is the condition (i<=10) is true?
Yes because i=7
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 7
Now, the value of i is:
i = 7+1 = 8
Is the condition (i<=10) is true?
Yes because i=8
```

Concept phase:

The first phase of a software development process during which user requirements are documented and reviewed

```
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 8
Now, the value of i is:
i = 8+1 = 9
Is the condition (i<=10) is true?
Yes because i=9
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 9
Now, the value of i is:
i = 9+1 = 10
Is the condition (i<=10) is true?
Yes because i=10
The statement cout<<"value of i = "<< i; is executed to print the output:
value of i = 10
and stop because the condition i<=10 is achieved.
```

If the statement:

```
cout<<"value of i = "<< i;
```

is replaced by the statement:

```
cout<<"\n value of i = "<< i;
```

Then the output on the screen is:

```
value of i = 1
value of i = 2
value of i = 3
value of i = 4
value of i = 5
value of i = 6
value of i = 7
value of i = 8
```

```
import pandas as pd
x={"col1":[1,2,3],"col2":["John","Alan","Mary"]}
y=pd.DataFrame(x)
y.dropna(inplace=True)
y=y[y.col1!=1]
print(y)
```

Output:

	col1	col2
1	2	Alan
2	3	Mary

```
from functools import reduce
x = [2, 4, 8, 10, 12, 50]
z = reduce (lambda a, b: a*b, x)
print(z)
```

Output: 384000

```
# 2 × 4 × 8 × 10 × 12 × 50 = 384000
```

```
value of i = 9
value of i = 10
```

```
from functools import reduce
x = [2, 4, 8, 10, 12, 50]
z = reduce (lambda a, b: a+b, x)
print(z)
```

Output:

86

```
# 2 + 4 + 8 + 10 + 12 + 50 = 86
```

If the ***for loop*** statement:

```
for (i=2; i<=10; i++)
```

is written instead of the statement:

```
for (i=1; i<=10; i++)
```

then the output on the screen is:

```
value of i = 2 value of i = 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7
value of i= 8 value of i = 9 value of i= 10
```

```
import random
x = random.randint(1,6)
print(x)

# Output: 5
```

If the *for loop* statement:

`for(i=1; i<10; i++)`

is written instead of the statement:

`for (i=1; i<=10; i++)`, then the output on the screen is:

Complexity: The extent to which a system's design or implementation is difficult to comprehend and validate

```
value of i = 1 value of i = 2 value of i= 3 value of i= 4 value of i= 5 value of i= 6
value of i = 7 value of i= 8 value of i = 9
```

- **Note:** the condition `i<=10` tells to print till value of i=10 but the condition `i<10` tells to print till value of i=9.

If the statement:

```
for(i=1; i=10; i++)
```

is written instead of the statement:

```
for(i=1; i<=10; i++)
```

then the output on the screen is:

```
value of i = 10  value of i = 10 value of i = 10 value of  i = 10 value of i= 10
value of i= 10 value of i = 10 value of i= 10  value of i = 10   value of i = 10
value of i = 10   value of i = 10  value of i = 10    value of i = 10   value of i = 10

continues ....
```

```
import numpy as np
x=np.array([1,2,3,4,5])
y=np.array([5,6,7,8,9])
print(np.vstack((x,y)))
```

Output:

[[1 2 3 4 5]
[5 6 7 8 9]]

```
x  = ("S", "t", "e", "p", "h", "e", "n", "h", "a", "w", "k", "i", "n", "g")
y = slice(2,5)
print(x[y])

# Output: ('e', 'p', 'h')
```

- **Note:**

If the statement:

```
cout<<"value of i = "<< i;
```

is replaced by the statement:

```
cout<<"\n "<< i;
```

```
x  = ("S", "t", "e", "p", "h", "e", "n", "h", "a", "w", "k", "i", "n", "g")
y = slice(0, 14, 2)
print(x[y])

# Output: ('S', 'e', 'h', 'n', 'a', 'k', 'n')
```

Then the output on the screen is:

Computer system audit: An analysis of a computer system's procedures in order to assess their efficacy and accuracy and make suggestions for improvements

C Program:

```
#include <stdio.h>
    int main() {
       printf("Albert Einstein\n");
       printf("Line: %d\n",__LINE__);
       printf("Line: %d\n",__LINE__);
       printf("1905 Papers\n");
       return 0;
}
```

Output:

Albert Einstein
Line: 4
Line: 5
1905 Papers

C Program:

```
#include <stdio.h>
int main() {
       printf("Albert Einstein\n");
       printf("Line: %d\n",__LINE__);
       #line 21
       printf("Line: %d\n",__LINE__);
       printf("1905 Papers\n");
       return 0;
}
```

Output:

Albert Einstein
Line: 4
Line: 21
1905 Papers

```
#include <stdio.h>
 int a = 2;
 int main(void)
{
  printf("%d\n", a);
  return 0;
}
```

Output:

2

```
#include <stdio.h>
 int a = 2;
 int main(void)
{
 int a =3;
  printf("%d\n", a);
  return 0;
}
```

Output:

3

- **What will be the output of the following program:**

```
#include<iostream>
using namespace std;
int main()
{
int i;
for (i =1; i<=5; i ++)
cout<<"\n Linux is not portable";
return 0;
}
```

Java program:

```
public class HelloWorld {
public static void main(String[] args) {
int[] array = new int[16];
int x = array.length;
// prints the Length of array
System.out.println(x);
}
}
```

Answer:

```
Linux is not portable
Linux is not portable
Linux is not portable
Linux is not portable
Linux is not portable
```

```
public class HelloWorld {
public static void main(String[] args) {
String a = "Albert";
char[] x = a.toCharArray();
for(int i=0; i< x.length; i++){
System.out.print(x[i]);
}
}
}
```

Output:

Albert

- **C++ program to print the first ten natural numbers using *while loop* statement**

The syntax of *while loop* statement is:

```
while (this is the condition)
{
execute this statement;
}
```

```
public class HelloWorld {
public static void main(String[] args) {
String x = "Albert#Einstein%profoundly^changed*physics&.";
String y = x.replaceAll("[^a-zA-Z0-9]",  " ");
System.out.println(y);
}
}
```

Output:

Albert Einstein profoundly changed physics

```
#include<iostream>
using namespace std;
int main()
{
int i = 1;
while (i<=10)
{
cout<<"\n "<< i++;
}
return 0;
}
```

The output on the screen is:

```
1
2
3
4
5
6
7
8
9
10
```

(i<=10) is the condition and

The statement

```
cout<<"\n "<< i++;
```

is repeatedly executed as long as a given condition `(i<=10)` is true.

If the statement:

```
public class HelloWorld {
public static void main(String[] args) {
for (int x=1; x<=5; x++){
    for (int y=1; y<=x; y++)
    System.out.print("*");
    System.out.println(" ");
}
}
}
```

Output:

```
*
**
***
****
*****
```

```
public class HelloWorld {
public static void main(String[] args) {
for (int x=5; x>=1; x--){
    for (int y=1; y<=x; y++)
    System.out.print("*");
    System.out.println(" ");
}
}
}
```

Output:

```
*****
****
***
**
*
```

Inheritance

The technique of building a new piece of code on a previously written piece of code

is replaced by the statement:

```
int i;
```

> **Project-based Learning**
>
> A method of learning that involves the creation of real-world projects

Then the **compilation error** will be displayed on the console screen because initialization is not defined [i.e., from where to start is not declared].

If the statement:

```
int i = 1;
```

is replaced by `int i = 0;`

Then the output on the screen is:

```
0
1
2
3
4
5
6
7
8
9
10
```

```
public class HelloWorld {
public static void main(String[] args) {
String x= "Albert Einstein profoundly changed physics.";
String y=x.replaceAll("[AEIOUaeiou]", "");
System.out.println(y);
}
}
```

Output:

lbrt nstn prfndly chngd physcs.

```
public class HelloWorld {
public static void main(String[] args) {
System.out.println("Albert Einstein");
System.exit(0);
}
}
```

Output:

Albert Einstein

Similarly if the statement `int i = 0;` is replaced by the `int i = 7;`

Then the output on the screen is:

7

> **Blockly**
>
> **Code.org's** block programming language for teaching children to code

8

9

10

Angular.js

A website development framework based on JavaScript

- **C++ program to print first 10 numbers using *do while loop* statement**

  The syntax of do while loop statement is:

```
do
{
execute this statement;
}
while(this is the condition);
```

```
#include<iostream>
using namespace std;
int main()
{
int i =1;
do
{
cout<<" \n i= "<< i++;
} while (i<=10);
return 0;
}
```

The output on the screen is:

```
i=1
i=2
i=3
i=4
i=5
i=6
```

```
public class HelloWorld {
void mult(int a, int b) {
System.out.println(a*b);
}
void mult(double a, double b) {
System.out.println(a*b);
}
void mult(double a, int b) {
System.out.println(a*b);
}
public static void main(String args[]) {
HelloWorld x = new HelloWorld ();
x.mult(20,30);
x.mult(30.12,12.24);
x.mult(50.15,9);
}
}
```

Output:

600

368.66880000000003

451.34999999999997

```
public class HelloWorld {
	public static void main(String[] args) {
	    String x = "005-065777";
	    String[] y = x.split("-");
	    String a = y[0];
	    String b = y[1];
	    System.out.println(a);
	    System.out.println(b);
	}
}
```

Output:

005

065777

```
i=7
i=8
i=9
i=10
```

```java
import java.time.LocalDateTime;

public class HelloWorld {
	public static void main(String[] args) {
		LocalDateTime x = LocalDateTime.now();
		System.out.println(x);
	}
}
```

Output:

2022-06-15T07:41:26.512844

The statement:

```
cout<<" \ni= "<< i++;
```

is executed and then condition **(i<=10)** is checked. If condition (i<=10) is true then

The statement:

```
cout<<" \ni= "<< i++;
```

is executed again. This process repeats until the given **condition (i<=10)** becomes false.

- **Write a program to print**

  When in doubt use brute force

  100 times using for loop statement.

Answer:

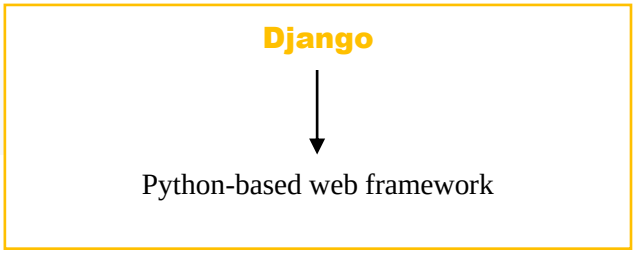
```
#include<iostream>
using namespace std;
int main()
{
int i;
for(i=0; i<=99; i++)
cout<<"\n When in doubt use brute force";
return 0;
```

```java
import java.util.List;
import java.util.stream.Stream;
import java.util.stream.Collectors;

public class HelloWorld {
public static void main(String[] args) {
List<Integer> x = Stream.of(11, 12, 13).collect(Collectors.toList());
System.out.println(x);
}
}
```

Output:

[11, 12, 13]

```
}
```



- **Program 6.3**

  C++ program to print the characters from A to Z using *for loop*, *do while loop* and *while loop* statement.

  - **C ++ program to print the characters from A to Z using *for loop* statement:**

```
#include<iostream>
using namespace std;
int main()
{
char a;
for( a='A'; a<='Z'; a++)
cout<<" \n"<< a;
return 0;
}
```

```
public class HelloWorld {
        public static void main(String[] args) {
        int x=20, y=60;
        if((x < y) && (y+1 < 105)) {
        System.out.println("Albert Einstein");
        } else {
            System.out.println(y);
}   }
}
```

Output:

Albert Einstein

The output on the screen:

```
A
B
C
D
E
F
G
H
I
```

```
public class HelloWorld {
public static void main(String[] args) {
if(2 + 2 + 2 + 2 + 2 == 10){
    System.out.print("True");
} else {
    System.out.print("False");
} }
}
```

Output:

True

```
J
K
L
M
N
O
P
Q
R
S
T
W
X
Y
Z
```

```
public class HelloWorld {
public static void main(String[] args) {
int x = 0;
for (int i = 1; i <= 50; i++) {
if (i % 2 == 1) {
x = x + 1;
}
}
System.out.println("There are " + x  + " odd numbers from 1 to 50");
}
}
```

Output:

There are 25 odd numbers from 1 to 50

If the statement: `for( a=A; a<=z; a++)` is written instead of the statement `for( a='A'; a<='z'; a++)` i.e., A is used instead of 'A' and Z is used instead of 'Z', then the **compilation error** will be displayed on the screen.

- **C ++ program to print the characters from A to Z using while loop statement:**

```
#include<iostream>
using namespace std;
int main()
{
char a = 'A';
while (a<='Z')
{
cout<<" \n"<< a++;
}
return 0;
}
```

```
public class HelloWorld {
public static void main(String[] args) {
int x = 0;
for (int i = 1; i <= 100; i++) {
if (i % 2 == 0) {
x = x + 1;
}
}
System.out.println("There are " + x  + " even numbers from 1 to 100");
}
}
```

Output:

There are 50 odd numbers from 1 to 100

- **C ++ program to print the characters from A to Z using do while loop statement:**

```
#include<iostream>
using namespace std;
int main()
{
char a = 'A';
do
{
cout<<" \n"<< a++;
} while (a<='Z');
return 0;
}
```

```
public class HelloWorld {
public static void main(String[] args) {
    String[] months = new String[] {
        "January",
        "February",
        "March",
        "April",
        "May",
        "June",
        "July",
        "August",
        "September",
        "October",
        "November",
        "December"
    }; System.out.println(months[0]);
}
}
```

Output:

January

- **Program 6.4**

C++ program to print the given number is even or odd.

```
#include<iostream>
using namespace std;
int main()
{
int a;
cout<<"Enter any number:";
cin>>a;
if(a%2 = = 0)
{
cout<<"the number is even";
}
else
{
cout<<"the number is odd";
}
return 0;
}
```

```
public class HelloWorld {
public static void main(String[] args) {
    String[] months = new String[] {
        "January",
        "February",
        "March",
        "April",
        "May",
        "June",
        "July",
        "August",
        "September",
        "October",
        "November",
        "December"
    };

    for(int x = 0; x < months.length; x++){
        System.out.println( x + " : " + months[x]);
    }
}
}
```

0 : January
1 : February
2 : March
3 : April
4 : May
5 : June
6 : July
7 : August
8 : September
9 : October
10 : November
11 : December

Output

The output on the screen:

```
Enter any number:
If you enter the number 6
the number is even
will be outputted on the screen.
```

Express.js

The back end web application framework for **Node.js**

- **Program 6.5**

  C++ program to print the remainder of two numbers

```
#include<iostream>
using namespace std;
int main()
{
int a, b, c;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
c = a % b;
cout<<"the remainder of a and b = "<< c;
return 0;
}
```

```
public class HelloWorld {
public static void main(String[] args) {
for (char x = 'a'; x <= 'c'; x++) {
for (char y = 'a'; y <= x; y++) {
System.out.print(y +" ");
}
System.out.println();
}
}
}
```

Output:

a

a b

a b c

The output on the screen:

```
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 2
the remainder of a and b = 1

will be outputted on the screen.
```

Python code:

```
a = "Albert"
print(type(a))
# Output: <class 'str'>

a = True
b = False
print(type(a))
print(type(b))
# Output:
<class 'bool'>
<class 'bool'>
```

If the statement:

```
cout<<"the remainder of a and b = "<< c;
```

HAML

(HTML Abstraction Markup Language)

A templating framework for making HTML cleaner and simpler by avoiding writing inline code in a web document

is replaced by the statement:

```
cout <<" the remainder of "<<a <<"and"<< b <<"=  "<< c;
```

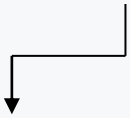
i.e.,

```
#include<iostream>
using namespace std;
int main()
{
int a, b, c;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
c = a % b;
cout <<" the remainder of "<<a <<"and"<< b <<"= "<< c;
return 0;
}
```

The output on the screen:

```
Enter any number:

If you enter the number 3

Enter any number:

If you enter the number 2

the remainder of 3 and 2 = 1

will be outputted on the screen.
```

```
string = "Albert"
list = ["Albert", "John", "Mary", "David", "Ram"]
tuple = ("1", "9", "0", "5")
dictionary = {
    "Albert": "Einstein",
    "Alan": "Turing",
    "Stephen": "Hawking",
    "John": "Calvin",
}
print("The length of string is: ", len(string))
print("The length of list is: ", len(list))
print("The length of tuple is: ", len(tuple))
print("The length of dictionary is: ", len(dictionary))
```



Output:

The length of string is: 6
The length of list is: 5
The length of tuple is: 4
The length of dictionary is: 4

```
x = ["Car", "Book", "Pencil", "House", "Box"]
y = ["Old " + i for i in x]
print(y)
```

Output:

['Old Car', 'Old Book', 'Old Pencil', 'Old House', 'Old Box']

C Program:

```c
#include <stdio.h>

int main()
{
    int x = 0;
    while (1) {
        printf("%d\n", x++);
        if (x == 4)
            break;
    }
    return 0;
}
```

Output:

0
1
2
3

C Program:

```c
#include <stdio.h>

int main()
{
    int x = 0, flag=0;
    while (0) {
        printf("%d\n", x++);
        if (x == 4)
            break;
    }
    if (flag==0)
     printf ("while loop will never get executed");
     return 0;
}
```

Output:

while loop will never get executed

- **Program 6.6**

  C++ program to check equivalence of two numbers

```c
#include<iostream>
using namespace std;
int main()
{
int x, y;
cout<<"Enter any number:";
cin>>x;
cout<<"Enter any number:";
cin>>y;
```

Checked Exception	Unchecked Exception
An exception thrown at compile time when the source code is translated into an executable code	An exception thrown at run-time when the executable code starts running

```
if(x-y==0)
{
cout<<"The two numbers are equivalent";
}
else
{
cout<<"The two numbers are not equivalent";
}
return 0;
}
```

Java program:

```
public class HelloWorld {
public static void main(String[] args) {
String x = "115";
System.out.println( Integer.parseInt( x ));
}
}
```

Output:

115

The output on the screen:

```
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 2
The two numbers are equivalent
will be outputted on the screen.
```

```
public class HelloWorld {
public static void main(String[] args) {
char x = '5';
int y = Integer.parseInt(String.valueOf(x));
System.out.println(y);
}
}
```

Output:

5

- **Program 6.7**

C ++ program to print whether the given number is positive or negative.

```
#include<iostream>
using namespace std;
int main()
{
int  a;
a = -35;
if(a>0)
{
cout<<"Number is positive";
}
else
```

```
import java.util.*;

public class HelloWorld {
public static void main(String[] args) {
List<Integer> x = new ArrayList<Integer>();
                x.add(19);
                x.add(16);
                x.add(15);
                x.add(12);
                x.add(10);
System.out.println("Smallest element in the collection:  "+Collections.min(x));
      }
  }
```

Output:

10

```
{
cout<<"Number is negative";
}
return 0;
}
```

```
import java.util.*;

public class HelloWorld {
public static void main(String[] args) {
List<Integer> x = new ArrayList<Integer>();
                x.add(19);
                x.add(16);
                x.add(15);
                x.add(12);
                x.add(10);
System.out.println("Largest element in the collection:  "+Collections.max(x));
      }
  }
```

Output:

Largest element in the collection: 19

The output on the screen:

```
Number is negative
```

- **Program 6.8**

  C++ program to print the sum of the first 10 digits using **for loop** statement

```
#include<iostream>
using namespace std;
int main()
{
int i, sum = 0;
for(i=1; i<=10; i++)
sum = sum + i;
cout<<"sum of the first 10 digits = "<< sum;
return 0;
}
```

```
import java.util.*;

public class HelloWorld {
public static void main(String[] args) {
List<Integer> x = new ArrayList<Integer>();
                x.add(19);
                x.add(16);
                x.add(15);
                x.add(12);
                x.add(10);
System.out.println(x);
      }
}
```

Output:

[19, 16, 15, 12, 10]

The output on the screen:

```
sum of the first 10 digits = 55
```

- **How the sum of the first 10 digits = 55 is outputted on the screen through the for Loop statement?**

```
i=1 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<=10 true?
Yes, do this
sum = sum + i = 0 +1 =1
Now,
i=2 (sum = 1)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +2 =3
Now,
i=3 (sum = 3)
Is i<=10 true?
Yes, do this
sum = sum + i = 3 +3 = 6
Now,
i=4 (sum = 6)
Is i<=10 true?
Yes, do this
sum = sum + i = 6 + 4= 10
Now,
i=5 (sum = 10)
Is i<=10 true?
Yes, do this
sum = sum + i = 10 + 5= 15
Now,
i=6 (sum = 15)
Is i<=10 true?
Yes, do this
sum = sum + i = 15 + 6 = 21
Now,
i=7 (sum = 21)
Is i<=10 true?
Yes, do this
sum = sum + i = 21 + 7 = 28
Now,
i=8 (sum = 28)
Is i<=10 true?
Yes, do this
sum = sum + i = 28 + 8 = 36
Now,
```

```
public class HelloWorld {
public static void main(String[] args) {
byte x =65;
char y = (char) x;
 // prints 65
System.out.println(x);
// prints A (ASCII is 65 for A)
System.out.println(y);
}
}
```

```
public class HelloWorld {
public static void main ( String args[]){
String x;
int y = 6;
x = (y == 3) ? "Alan" : "Albert";
System.out. println(x);
}
}
```

Output:

Albert

```
public class HelloWorld {
public static void main(String[] args) {
                int x;
                int[] num = new int[]{1, 3, 4, 5, 6, 7};
                x = 7;
                int y = x * ((x + 1) / 2);
                int z = 0;
                for (int i: num) {
                 z += i; }
                System.out.print( "Missing Number: " + (y - z));
}
}
```

Output:

Missing Number: 2

```
i=9 (sum = 36)
Is i<=10 true?
Yes, do this
sum = sum + i = 36 + 9 = 45
Now,
i=10 (sum = 45)
Is i<=10 true?
Yes, do this
sum = sum + i = 45 + 10 = 55
stops because the condition i<=10 is achieved
```

```
public class HelloWorld {
public static void main(String[] args) {
int[] num = { 11, 12, 13, 14, 15 };
int sum = 0;
for (int x : num)
sum += x;
System.out.println(sum);
}
}
```

Output:

65

The statement:

```
cout<<"sum of the first 10 digits = "<< sum;
```

is executed to print the output:

```
sum of the first 10 digits = 55
```

```
public class HelloWorld {
public static void main(String[] args) {
        String x = "Albert";
        int a = 1;
        char b = 3;
        System.out.println(a + b);
        System.out.println(x.substring(a, b));
        }
}
```

Output:

4

lb

If the statement:

```
int i, sum = 0;
```

is replaced by `int i, sum = 1;`

Then the output on the screen is:

```
sum of the first 10 digits = 56
```

- **What will be the output if the for loop statement `for(i=1; i<=10; i++)` is replaced by the statement `for(i=2; i<10; i++)`?**

Answer: sum of 10 digits = 44

Knowledge engineering

A branch of AI that develops rules to apply to data in order to mimic a human expert's thought process

If the statement `int i, sum, sum = 0;` is written instead of `int i, sum = 0;`

Then the *compilation error message* will be displayed on the screen (stating that sum is twice declared).

If the *for loop* is ended with a semicolon i.e.,

```
for( i=1; i<=10; i++);
```

Then the ***compilation error*** will be displayed on the console screen.

```
import java.text.SimpleDateFormat;
import java.util.Date;

public class HelloWorld {
public static void main(String[] args) {
String pattern = "MM-dd-yyyy";
SimpleDateFormat simpleDateFormat = new
SimpleDateFormat(pattern);
String x = simpleDateFormat.format(new Date());
System.out.println(x);
}
}
```

Output:

06-14-2022

- **Program 6.9**

  C++ program to print the average of the first 10 numbers using ***for loop*** statement

```
#include<iostream>
using namespace std;
int main()
{
int i, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
cout<<"sum of the first 10 numbers = "<< sum;
cout<<"average of the first 10 numbers = "<< avg;
return 0;
}
```

The output on the screen:

```
sum of the first 10 numbers = 55
average of the first 10 numbers = 5
```

```
public class HelloWorld {
public static void main(String[] args) {
int a = 20*20-20;
System.out.println(a);
}
}
```

Output:

380

```
public class HelloWorld {
public static void main(String[] args) {
double x = 1.66666666;
String y = Double.toString(x);
System. out. println(y);
}
}
```

Output:

1.66666666

The average of the first10 numbers = 55/10 = 5.5 not 5. But the **output on the screen** is:

```
average of the first 10 numbers = 5
```

because int is used instead of float.

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;

public class HelloWorld {
public static void main(String[] args) {
List<Integer> x = new ArrayList<>(Arrays.asList(11, 20, 11, 12, 12, 13,
50, 13, 23, 14, 25, 25));
// print the list with duplicates
System.out.println(x);
List<Integer> y = x.stream().distinct().collect(Collectors.toList());
// print the list with duplicates removed
System.out.println(y);
}
}
```

Output:

[11, 20, 11, 12, 12, 13, 50, 13, 23, 14, 25, 25]

[11, 20, 12, 13, 50, 23, 14, 25]

If the ***data type*** float is used i.e.,

```
#include<iostream>
using namespace std;
int main()
{
float i, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
cout<<"sum of the first 10 numbers = "<< sum;
cout<<"average of the first 10 numbers = "<< avg;
return 0;
}
```

The output on the screen:

```
sum of the first 10 numbers = 55
average of the first 10 numbers = 5.5
```

10,000-year clock

A precise, long-lasting mechanical clock that ticks once a year for 10,000 years

- **Program 7.0**

  C++ program to print the product of the first 10 digits using ***for loop*** statement

```
#include<iostream>
using namespace std;
int main()
{
```

```
int i, product = 1;
for( i=1; i<=10; i++)
product = product * i;
cout<<"the product of the first 10 digits =%d", product;
return 0;
}
```

Algorithmic accountability

The idea that enterprise organizations should be held answerable for the unfair outcomes of their pre-programmed algorithms

The output on the screen:

```
the product of the first 10 digits = 3628800
```

- **How the product of the first 10 digits = 3628800 is outputted on the screen through the for Loop statement?**

```
i=1 (product = 1 because the product is initialized to 1 in the statement int i,
product = 1;)
Is i<=10 true?
Yes, do this
product = product *  i = 1 * 1 =1
Now,
i=2 (product = 1)
Is i<=10 true?
Yes, do this
product = product *  i = 1 * 2 = 2
Now,
i=3 (product = 2)
Is i<=10 true?
Yes, do this
product = product *  i = 2 * 3 = 6
Now,
i=4 (product = 6)
Is i<=10 true?
Yes, do this
product = product *  i = 6 * 4 = 24
Now,
i=5 (product =24)
```

```
import java.util.*;
public class HelloWorld {
public static void main(String[] args) {
        String x = "Alan";
        String y = "Turing";
        // print String before swapping
        System.out.println( x + " " + y);

        x = x + y;
        y = x.substring(0, x.length() - y.length());
        x = x.substring(y.length());
        // print String after swapping
        System.out.println(x + " " + y);
        }
}
```

Output:

Alan Turing
Turing Alan

```
Is i<=10 true?
Yes, do this
product = product *  i = 24 * 5 =120
Now,
i=6 (product =120)
Is i<=10 true?
Yes, do this
product = product *  i = 120 * 6 = 720
Now,
i=7 (product =720)
Is i<=10 true?
Yes, do this
product = product *  i = 720 * 7 = 5040
Now,
i=8 (product =5040)
Is i<=10 true?
Yes, do this
product = product *  i = 5040 * 8 = 40320
Now,
i=9 (product = 40320)
Is i<=10 true?
Yes, do this
product = product *  i = 40320 * 9 = 362880
Now,
i=10 (product = 362880)
Is i<=10 true?
Yes, do this
product = product *  i = 362880 * 10 = 3628800

stops because the condition i<=10 is achieved.
```

```
public class HelloWorld {
        public static void main(String[] args) {
                String x = "alan turing";
                x = x.replace("tu", "");
                System.out.println(x);
        }
}
```

Output:

alan ring

```
public class HelloWorld {
        public static void main(String[] args) {
        for(int x=4; x>=0; x--) {
            System.out.println();
            for(int y=x; y<4; y++) {
            System.out.print(" a ");
            } } System.out.println();
}

}
```

Output:

a
a a
a a a
a a a a

The statement:

Data point: A discrete unit of information

```
cout<<"the product of the first 10 digits = "<< product;
```

is executed to display the output:

```
the product of the first 10 digits = 3628800
```

```
public class HelloWorld {
public static void main(String[] args) {
double x = 85.78;
System.out.println(Math.round(x));
float y = 7777.658999f;
System.out.println(Math.round(y));
}
}
```

Output:

86

7778

If the statement:

`int i, product = 1;` is replaced by `int i, product = 0;`

Then the output on the screen is:

```
the product of the first 10 digits = 0
```

If the statement:

`for(i=1; i<=10; i++)` is replaced by

```
for(i=5; i<=8; i++)
```

Cache algorithm

An algorithm used to manage a computer's cache of information

Then the output on the screen is:

```
the product of the first 10 digits = 1680
```

- **Program 7.1**

  C++ Program to print the table of a number using the ***for loop*** statement

```
#include<iostream>
using namespace std;
int main()
{
```

Domain generation algorithm

A computer algorithm that produces a large number of domain names

```
int n, i;
cout<<"Enter any number:";
cin>>n;
for( i=1; i<=5; i++)
cout<< n <<"  *   "<< i <<" = "<< n*i;
return 0;
}
```

```
public class HelloWorld {
public static void main(String args1[]) {
char[] x= {'a', 'l', 'a', 'n', ' ', 't', 'u', 'r', 'i', 'n', 'g'};
String b = new String(x);
System.out.println(b);
}
}
```

Output:

alan turing

The output on the screen:

```
Enter any number:
If you enter the number 2 (i.e., n=2)
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
will be outputted on the screen.
```

```
public class HelloWorld {
public static void main(String args1[]) {
char[] x= {'a', 'l', 'a', 'n', ' ', 't', 'u', 'r', 'i', 'n', 'g'};
String b = String.valueOf(x);
System.out.println(b);
}
}
```

Output:

alan turing

- **How the execution takes its Way through the *for Loop* statement**

```
Since you entered the number 2, therefore: n=2.
i=1
Is i<=5 true?
Yes, print this
2 * 1 = 2
using the statement cout<< n <<"  *   "<< i <<"  =  "<< n*i;

Now,
i=2
Is i<=5 true?
Yes, print this
2 * 2 = 4
using the statement cout<< n <<"  *   "<< i <<"  =  "<< n*i;

Now,
```

CRUSH

(Controlled Replication Under Scalable Hashing)

A hash-based algorithm for determining how and where information should be stored and retrieved in a distributed object-based storage cluster

```
i=3
Is i<=5 true?
Yes, print this
2 * 3 = 6
using the statement cout<< n <<"  *   "<< i <<"  =  "<< n*i;

Now,
i=4
Is i<=5 true?
Yes, print this
2 * 4 = 8
using the statement cout<< n <<"  *   "<< i <<"  =  "<< n*i;

Now,
i=5
Is i<=5 true?
Yes, print this
2 * 5 = 10
using the statement cout<< n <<"  *   "<< i <<"  =  "<< n*i;

stop Now because the condition i<=5 is achieved.
```

Algorithmic transparency

The idea that individuals should be able to see or understand the aspects that influence algorithmic decisions

If the symbol * is replaced by +

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
int n, a;
cout<<"Enter any number:";
cin>>n;
for( i=1; i<=5; i++)
cout<< n <<"  +   "<< i <<" = "<< n + i <<endl;
return 0;
}
```

```
public class HelloWorld {
public static void main(String args1[]) {
String x = "550";
int b = Integer.decode(x);
System.out.println(" "+ b);
}
}
```

Output:

550

Then the **output on the screen** is:

```
Enter any number:

If you enter the number 2 (i.e., n=2)

2 + 1 = 3

2 + 2 = 4

2 + 3 = 5

2 + 4 = 6

2 + 5 = 7

will be outputted on the screen.
```

Data Vault Modeling

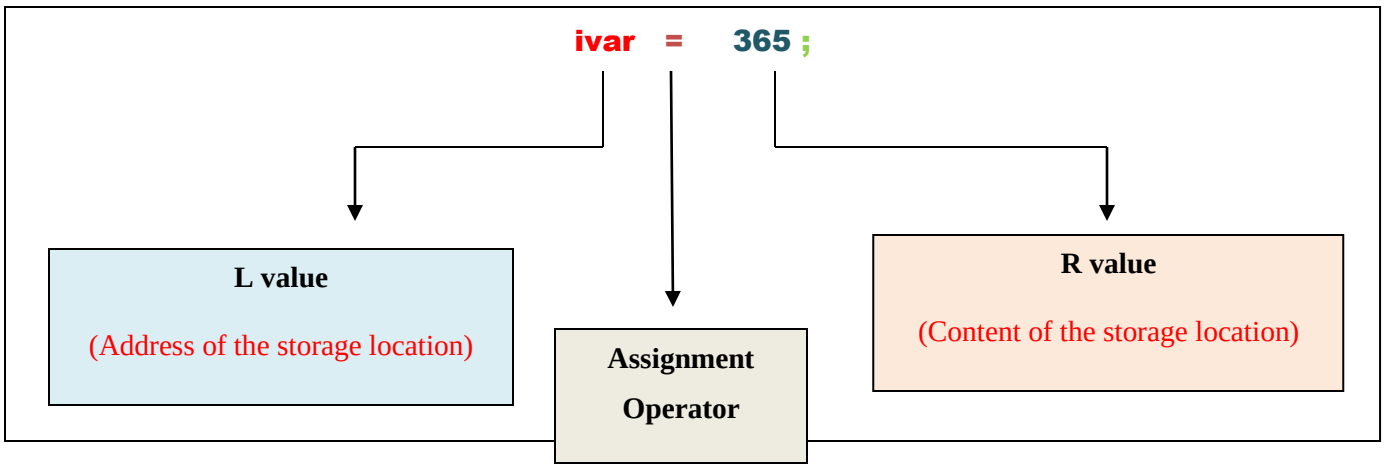
A database modelling technique for storing historical data from numerous operating systems over a lengthy period of time

Data regarding physical things that are tied to a geographical place is collected, displayed and manipulated

Geospatial Analytics

```
[manju@localhost ~]$ time perl -le '$n=0;foreach (1..10) {$n++}'

real    0m0.015s
user    0m0.000s
sys     0m0.004s

[manju@localhost ~]$ time perl -le '$n=0;foreach (1..10) {$n+1}'

real    0m0.016s
user    0m0.000s
sys     0m0.005s
```

n++ executes faster than n+1



C Program:

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main()

{

int b;

char a[] = "1905Papers";

b = atoi(a);

printf("%d\n", b);

return (0);

}
```

Output:

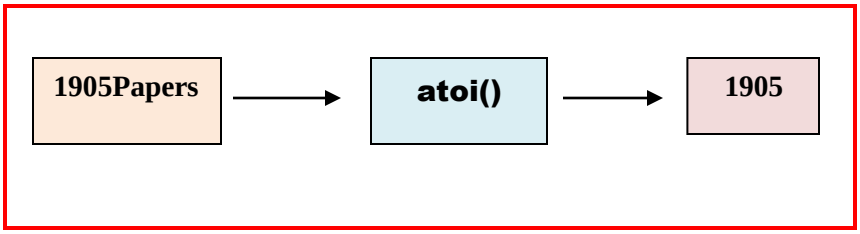
1905

- **Program 7.2**

  C++ program:

  If you enter a character M

  Output must be: ch = M




```
#include<iostream>
using namespace std;
int main()
{
char M;
cout<<"Enter any character:";
cin>>M;
cout<<"ch= "<< M;
return 0;
}
```

The output on the screen:

```
Enter any character:

If you enter the character M

ch = M

will be outputted on the screen.
```

Python code:

```
myfunc = lambda x,y : x/y
print(myfunc(16, 2))
```

Output:

8.0

- **Note:**

If we replace the statement: `cin>>M;` by the statement:

```
M = getchar();
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
char M;
cout<<"Enter any character:";
M = getchar();
cout<<"ch= "<< M;
return 0;
}
```

```
def myfunction(x):
    return lambda y : y * x
a = myfunction(16)
print(a(3))
```

Output:

48

There will be no change in the output on the screen i.e., The output on the screen is:

```
Enter any character:

If you enter the character K

ch = K

will be outputted on the screen.
```

```
a = [[10,20,30],[40,50,60],[70,80,90]]
b = [x for i in a for x in i]
print(b)
```

Output:

[10, 20, 30, 40, 50, 60, 70, 80, 90]

If we replace the statement: `cout<<"ch= "<< M;` by the statement:

```
putchar(M);
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
char M;
cout<<"Enter any character:";
cin>>M;
putchar(M);
return 0;
}
```

```
for x in range(2):
    print (x)
```

Output:

0

1

```
for x in range(2,6):
    print (x)
```

Output:

2

3

4

5

There will be **no change in the output** on the screen i.e., The output on the screen is:

```
Enter any character:
If you enter the character M
M will be outputted on the console screen.
```

If we replace the statement: `cin>>M;` by the statement:

```
M = getchar();
```

and the statement:

```
cout<<"ch= "<< M;
```

by the statement:

`putchar(M);` i.e.,

```
for x in range(2,6,3):
    print (x)
```

Output:

2

5

```
#include<iostream>
using namespace std;
int main()
{
char M;
cout<<"Enter any character:";
M = getchar();
putchar(M);
return 0;
}
```

```
a = "Albert Einstein was born in Ulm"
b = a.split(' ')
print(b)
print(' '.join(b))
```

Output:

['Albert', 'Einstein', 'was', 'born', 'in', 'Ulm']

Albert Einstein was born in Ulm

The output on the screen:

```
Enter any character:
If you enter the character S
S will be outputted on the screen.
```

- **Write a program to print the absolute value of a number**

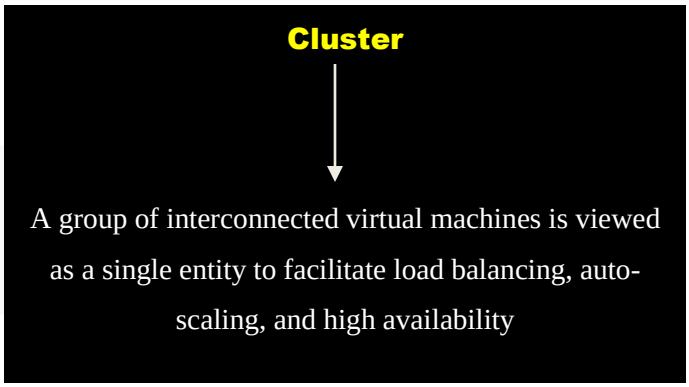
Answer:

```
# Python Program to delete a file

import os
os.remove("1.txt")
print("1.txt file does not exist")
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main()
{
int a, b;
a= - 2;
b= abs(a);
cout<<" absolute value of a = "<< b<< endl;
return 0;
}
```

The output on the screen:

```
absolute value of a = 2
```



- **Program 7.2**

  C ++ program to print the first 5 numbers starting from one together with their squares

```
#include<iostream>
using namespace std;
int main()
{
int i;
for( i=1; i<=5; i++)
cout<<"\n number = "<< i <<"its square = "<< i*i;
return 0;
}
```

The output on the screen:

```
number=1 its square=1
number=2 its square=4
number=3 its square=9
number=4 its square=16
number=5 its square=25
```

```
def myfunc(a):
    a.append(15)
a = [12, 13, 14]
print(a)
myfunc(a)
print(a)
```

Output:

[12, 13, 14]

[12, 13, 14, 15]

- **How the execution takes its way through the for loop statement**

```
i=1
Is i<=5 true?
```

```
Yes, print this
number=1 its square=1
using the statement cout<<"\n number = "<< i <<"its square =  "<< i*i;

Now,
i=2
Is i<=5 true?
Yes, print this
number=2 its square=4
using the statement cout<<"\n number = "<< i <<"its square =  "<< i*i;

Now,
i=3
Is i<=5 true?
Yes, print this
number=3 its square=9
using the statement cout<<"\n number = "<< i <<"its square =  "<< i*i;

Now,
i=4
Is i<=5 true?
Yes, print this
number=4 its square=16
using the statement cout<<"\n number = "<< i <<"its square =  "<< i*i;

Now,
i=5
Is i<=5 true?
Yes, print this
number=5 its square=25
using the statement cout<<"\n number = "<< i <<"its square = "<< i*i;

stop Now because the condition (i<=5) is achieved.
```

- **Note:**

If the statement

Cron job

A Linux command that allows us to schedule tasks to be executed at a later time in future

```
cout<<"\n number = "<< i <<"its square = "<< i*i;
```

is replaced by the statement:

```
cout<<"\n number =   "<< i <<"\t its square =  "<< i*i;
```

Then the **output on the screen** is:

```
number=1     its square=1
number=2     its square=4
number=3     its square=9
number=4     its square=16
number=5     its square=25
```

```
def myfunc(x, y, *argv):
    z = x * y
    for i in argv:
        z *= i
    return z
print(myfunc(4, 8, 12, 16, 20))
```

Output:

122880

tab /t is included because to leave space between

number=1 and its square=1

Suppose `cout<<"\n number =   "<< i <<"\t its square = "<< i*i;` is replaced by the statement

```
cout<<"\n number =   "<< i <<"\n its square =  "<< i*i;
```

Then the **output on the screen** is:

```
number=1
its square=1
number=2
its square=4
```

Bare-metal

The hard disc of a computer

↓

{The storage media on which the operating system (OS) is installed}

```
number=3
its square=9
number=4
its square=16
number=5
its square=25
```

Bastion host

A specialized server that is intentionally accessible to the public network and is configured to survive cyberattacks

- **Write a program to print the first 10 numbers starting from one together with their squares and cubes?**

Answer:

```
#include<iostream>
using namespace std;
int main()
{
int i;
for( i=1; i<=10; i++)
cout<<"number =  "<< i <<" its square =  "<< i*i <<" its cube =  "<< i*i*i<< endl;
return 0;
}
```

```
def myfunc(**kwargs):
    for i, x in kwargs.items():
        print(i + ": " + x)
myfunc(physicist = "Stephen", cosmologist = "William",  author = "Hawking")
```

Output:

physicist: Stephen
cosmologist: William
author: Hawking

- **Program 7.3**

C ++ program to print the address of x and the value assigned to x

```
#include<iostream>
using namespace std;
int main()
{
int x, *p;
cout<<"Enter any integer:";
cin>>x;
```

```
import pandas as pd
a = pd.Series([3, 4, 13, 30, 13])
b = pd.Series([13, 13, 30, 27, 29])
a=a[~a.isin(b)]
print(a)
```

Output:

0 3
1 4
dtype: int64

```
p = &x;
cout<<"The address of the variable x = "<< p;
cout<<"The value of the variable x = "<< *p;
return 0;
}
```

```
# union of series

import pandas as pd
import numpy as np
a = pd.Series([3, 4, 13, 30, 13])
b = pd.Series([13, 13, 30, 27, 29])
c= pd.Series(np.union1d(a, b))
print(c)
```

Output:

```
0    3
1    4
2   13
3   27
4   29
5   30
dtype: int64
```

The output on the screen:

```
Enter any integer:
If you enter the integer 1
The address of the variable x = 0x7fffc60478a4
The value of the variable x = 1
will be outputted on the screen.
```

If the statements:

```
cout<<"The address of the variable x = "<< p;
cout<<"The value of the variable x = "<< *p;

are replaced by the statement:

cout<<"The address of the variable x = "<< p <<"its value = "<< *p;
```

i.e.,

```
#include<iostream>
using namespace std;
int main()
{
int x, *p;
cout<<"Enter any integer:";
cin >> x;
p = &x;
```

```
# intersection of series

import pandas as pd
import numpy as np
a = pd.Series([3, 4, 13, 30, 13])
b = pd.Series([13, 13, 30, 27, 29])
c= pd.Series(np.intersect1d(a, b))
print(c)
```

Output:

```
0   13
1   30
dtype: int64
```

```
cout<<"The address of the variable x = "<< p <<"its value = "<< *p;
return 0;
}
```

Then the **output on the screen** is:

The address of the variable x = 0x7fff78508cc4 its value = 2

C++ program to print the sum of two numbers using pointers

```
#include<iostream>
using namespace std;
int main()
{
int x, y, *p, *q, sum;
cout<<"Enter any number:";
cin >> x;
cout<<"Enter any number:";
cin >> y;
p = &x;
q = &y;
sum = *p + *q;
cout<<"\n sum of entered numbers = "<< sum;
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
int x =25;
(x>=56)? (printf("Albert")) : (printf("Einstein"));
return 0;
}
```

Output:

Einstein

The output on the screen:

```
Enter any number:
If you enter the number 4
Enter any number:
If you enter the number 3
```

```
#include <stdio.h>
int main() {
int x =151;
(x>=56)? (printf("Albert")) : (printf("Einstein"));
return 0;
}
```

Output:

Albert

```
sum of entered numbers = 7

will be outputted on the screen.
```

- **C++ program to print the product, subtraction and division of two numbers using pointers**

```
#include<iostream>
using namespace std;
int main()
{
int x, y, *p, *q, product, subtract, div;
cout<<"Enter any number:";
cin>> x;
cout<<"Enter any number:";
cin>> y;
p = &x;
q = &y;
product = *p * *q;
subtract = *p - *q;
div= *p / *q;
cout<<"\n product of entered numbers = "<< product;
cout<<"\n subtract of entered numbers = "<< subtract;
cout<<"\n division of entered numbers =  "<< div;
return 0;
}
```

The output on the screen:

```
Enter any number:
If you enter the number 4
Enter any number:
If you enter the number 2
product of entered numbers = 8
```

```
#include<stdio.h>
#define solve(s,t,u,m,p,e,d) m##s##u##t
#define start solve(a,n,i,m,a,t,e)

int start() {
    printf("Albert Einstein");
   return 0;
}
```

Output:

Albert Einstein

```
#include <stdio.h>
int main() {
    float x = 6.77;
    int y = (int)x+1;
    printf("%d\n", y);
    return 0;
}
```

Output:

7

```
subtract of entered numbers = 2
division of entered numbers = 2

will be displayed on the screen.
```

- **C++ program to find the greatest of two numbers using pointers**

```
#include<iostream>
using namespace std;
int main()
{
int x, y, *p, *q;
cout<<"Enter any integer:";
cin>> x;
cout<<"Enter any integer:";
cin>> y;
p = &x;
q = &y;
if(*p>*q)
{
cout<<"x is greater than y";
}
else
{
cout<<"y is greater than x";
}
return 0;
}
```

```
#include<stdio.h>
#include<string.h>
#include<ctype.h>

int main() {
    char a[]="Albert";
    char b[]=" Einstein";
    printf("%s", strcat(a, b));
    return 0;
}
```

Output:

Albert Einstein

The output on the screen:

```
Enter any integer:
If you enter the integer 10
Enter any integer:
```

```
If you enter the integer 16
y is greater than x
will be outputted on the screen.
```

- **What is the output of the following programs?**

A)

```
#include <iostream>
using namespace std;
int main() {
  string x[2][4] = {
    {"a", "b", "c", "d"},
    {"e", "f", "g", "h"}
  };
cout << x[0][1];
return 0;
}
```

```
#include<stdio.h>
int main() {
int x = 3;
if(x == (2, 4)) {
printf("Albert");
} else {
printf("Einstein");
}
return 0;
}
```

Output:

Einstein

Answer:

b

B)

```
#include <iostream>
using namespace std;
int main()
{
int x, t, c;
x=12;
t=2;
c = x/t;
cout<<"velocity = "<< c <<"m/s";
return 0;
```

Cycle Stealing

↓

A technique for accessing RAM without disrupting the CPU

```
}
```

Answer:

```
velocity = 6 m/s
```

- **Program 7.4**

  C++ program to print the sum of two numbers using functions

```
#include<iostream>
using namespace std;
int addition();
int main()
{
int answer;
answer = addition();
cout<<"The sum of two numbers is: "<<answer;
return 0;
}

int addition()
{
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
return x+y;
}
```

Single-bit error:

A single bit of a data is altered from 1 to 0 or 0 to 1

Burst Error:

2 or more bits of data are altered from 0 to 1 or 1 to 0

```
#include<stdio.h>
int main() {
int x = 7;
if(x = x - 7 > 3) {
printf("Albert");
}
else {
printf("Einstein");
}
return 0;
}
```

Output:

Einstein

The output on the screen:

```
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
sum of two numbers = 8
will be displayed on the screen.
```

Python code:

```
a = ('Albert', 1905, 'Papers')
b = ['Albert', 1905, 'Papers']
print(a[0])
print(b[1])
```

Output:

Albert

1905

- **C++ program to print the product of two numbers using functions**

```
#include<iostream>
using namespace std;
int multiplication();
int main()
{
int answer;
answer = multiplication();
cout<<"The product of two numbers is: "<<answer;
return 0;
}

int multiplication()
{
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
return x*y;
}
```

```
a = [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
print(a[1 : : 2])
```

Output:

[12, 14, 16, 18, 20]

```
a = 50 # Global-scope variable
def myfunc():
   a = 100 # Local-scope variable
   print(a)
print(a)
myfunc()
print(a)
```

Output:

50

100

50

The output on the screen:

```
Enter any integer:
```

```
If you enter the integer 3

Enter any integer:

If you enter the integer 5

product of two numbers = 15

will be outputted on the screen.
```

```
b = [1, 2, 3, 4, 5]
a = [i**2 for i in b]
print(a)
```

Output:

[1, 4, 9, 16, 25]

- **C++ program to print the greatest of two numbers using functions**

- **Bit rate:** The number of bits transmitted in one second
- **Baud rate:** The number of signal units per second required to represent those bits

$$\text{Baud rate} = \frac{\text{Bit rate}}{\text{Number of bits represented by each signal shift}}$$

```
#include<iostream>
using namespace std;
int largest();
int main()
{
int answer;
answer = largest();
cout<<"The largest of two numbers is: "<<answer;
return 0;
}

int largest()
{
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
if(x>y)
return x;
if(y>x)
return y;
}
```

```
b = [1, 2, 3, 4, 5]
a = [i%2 for i in b]
print(a)
```

Output:

[1, 0, 1, 0, 1]

The output on the screen:

```
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
largest of two numbers= 5
will be outputted on the screen.
```

Retransmission:

A method in which the receiver identifies an error and requests that the sender resend the message

- **C++ program to print the greatest of three numbers using functions**

```
#include<iostream>
using namespace std;
int largest();
int main()
{
int answer;
answer = largest();
cout<<"largest of three numbers= "<< answer;
return 0;
}
int largest()
{
int x, y, z;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
cout<<"Enter any integer:";
cin>>z;

if(x>y&& x>z)
return x;
if(y>x&& y > z)
return y;
if(z>x && z>y)
return z;
}
```

Unicast transmission

The data is sent from a single source to a single recipient

```
b = [1, 2, 3, 4, 5]
a = {i:i**2 for i in b}
print(a)
```

Output:

{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

The output on the screen:

```
Enter any integer:

If you enter the integer 3

Enter any integer:

If you enter the integer 5

Enter any integer:

If you enter the integer 10

largest of three numbers = 10

will be outputted on the screen.
```

```
p = [3, 5, 7]
q = [9, 11, 13]
r = [(a + b) for (a,b) in zip(p,q)]
print(r)
```

Output:

[12, 16, 20]

```
p = [3, 5, 7]
q = [9, 11, 13]
r = [(a * b) for (a,b) in zip(p,q)]
print(r)
```

Output:

[27, 55, 91]

- **C++ program to print the square of the number using functions**

```
#include<iostream>
using namespace std;
int square(;
int main()
{
int  answer;
answer = square();
cout<<"square of the number = "<< answer;
return 0;
}
int square()
{
int x;
cout<<"Enter any integer:";
cin>>x;
return x*x;
}
```

```
p = [3, 5]
q = [9, 11]
r = [(a,b) for a in p for b in q]
print(r)
```

Output:

[(3, 9), (3, 11), (5, 9), (5, 11)]

The output on the screen:

```
Enter any integer:
If you enter an integer 5
square of the number = 25
will be outputted on the screen.
```

Shift left testing

A practice aimed at detecting and correcting errors considerably earlier in the software development lifecycle

- **What is the output of the following program:**

```
#include<iostream>
using namespace std;
int main()
{
int x;
x=6;
cout<<"The address of x = "<<&x;
return 0;
}
```

Answer:

```
The address of x = 0x7ffd80d2c06c
```

```
#include <iostream>
using namespace std;

int main()
{
    int a = 32;
    int& b = a;
     b = 62;
    cout << "a = " << a << endl;
    return 0;
}
```

Output:

a = 62

- **Program 7.5**

  C++ has an inbuilt multiple branch selection statement [called ***switch***] which successively tests the value of an expression against a list of integer or character constants. When a match is found, the statement sequences associated with that constant are executed. `Switch(case)` allows us to make decision from the number of choices i.e., from the number of cases.

For example:

```
#include<iostream>
using namespace std;
int main()
{
char ch;
cout<<"Enter any character:";
cin>>ch;
switch(ch)
{
case 'R':
cout<<"Red";
break;
case 'W':
cout<<"White";
break;
case 'Y':
cout<<"Yellow";
break;
case 'G':
cout<<"Green";
break;
default:
cout<<"Error";
break;
}
return 0;
```

Self-Serving Data

Making data accessible to data users for statistical and research purposes, allowing them to generate and build their own data analysis, as well as modify and adapt judgments based on it

```
}
```

The output on the screen:

```
Enter any character:

If you enter a character R

Red

will be outputted on the screen.
```

If the statements:

```
case 'R':
cout<<"Red";
break;
case 'W':
cout<<"White";
break;
case 'Y':
cout<<"Yellow";
break;
case 'G':
cout<<"Green";
break;
default:
cout<<"Error";
break;
```

are replaced by the statements:

```
case 'R':
cout<<"Red";
case 'W':
cout<<"White";
```

C program:

```
#include <stdio.h>
#include <ctype.h>
int main() {
    char a = 'k';
    printf("%c", toupper(a));
    char b = 'K';
    printf("\n %c", tolower(b));
    char x = '2';
    printf("\n %c", toupper(x));
    return 0;
}
```

Output:

K
k
2

```
#include<stdio.h>
int main() {
    int a;
    a=printf("Albert");
    printf("%d", a);
    return 0;

}
```

Output:

Albert6

```
#include<stdio.h>
int main() {
    int a;
    a=printf("Alan");
    printf("%d", a);
    return 0;

}
```

Output:

Alan4

```
case 'Y':
cout<<"Yellow";
break;
case 'G':
cout<<"Green";
break;
default:
cout<<"Error";
break;
```

```
#include <stdlib.h>
#include <stdio.h>

int main() {
    char a = 'K';
    if(a =='A') {
        exit(0);
        } else {
    printf("K");
    }
    return 0;
    }
```

Output:

K

Then the **output on the screen** is:

```
Red

White

Yellow
```

i.e., the output will be printed till yellow even though you have entered the **character R**.

- In C, a switch can have at least 257 case statements. C++ recommends that at least 16,384 case statements be supported.

Why do we need arrays?

Arrays provide a more convenient way of storing variables of a similar data type together instead of storing them separately. Each value of the array will be accessed separately.

- **Program 7.6**

  C++ program to print the output:

```
Element [0] = 16

Element [1] = 18
```

```
#include<stdio.h>
int main(){
int const x = 12;
printf("%d", x+1);
return 0;
}
```

Output:

13

```
Element [2] = 20

Element [3] = 25

Element [4] = 36
```

using arrays:

```
#include<iostream>
using namespace std;
main()
{
int i;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
cout<<"Element ["<< i <<" ] = "<< num[i] << endl;
return 0;
}
```

```
#include<stdio.h>
int main() {
int a = 20;
printf("%d\n%d\n%d\n", a++,a,++a);
return 0;
}
```

Output:

21

22

22

The output on the screen:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
```

```
#include <stdio.h>

int main() {
int a;
int b= 1;
int m[3] = { 1, 2, 3};
a = 5 * 6 + m[b+1] - (6 / b);
printf("%d", a);
return 0;
}
```

Output:

27

Suppose the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[i] << endl;
```

is replaced by the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[0] << endl;
```

Then the **output on the screen**:

```
Element [0] = 16

Element [1] = 16

Element [2] = 16

Element [3] = 16

Element [4] = 16
```

```
#include<stdio.h>

int main() {
    char c = 'x';
    int n = 0, y;
    y = c + n;
    printf("The ASCII value of x is: %d\n", y);
    return 0;

}
```

Output:

The ASCII value of x is: 120

Suppose the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[i] << endl;
```

is replaced by the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[1] << endl;
```

The output on the screen:

```
Element [0] = 18
Element [1] = 18
Element [2] = 18
Element [3] = 18
Element [4] = 18
```

```
#include <stdio.h>
int main(void) {
printf("Ala\bn");
return 0;
}
```

Output:

Ala•n

```
#include <stdio.h>
int main()
{
while (printf("Albert Einstein"))
return 0;
}
```

Output:

Albert Einstein

Suppose the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[i] << endl;
```

is replaced by the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[2] << endl;
```

The output on the screen:

```
Element [0] = 20
Element [1] = 20
Element [2] = 20
Element [3] = 20
Element [4] = 20
```

```
#include <stdio.h>
int main(){
    extern int x;
    printf("%d", x);
   return 0;
}
    int x = 52;
```

Output:

52

Suppose the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[i] << endl;
```

is replaced by the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[3] << endl;
```

The output on the screen:

```
Element [0] = 25
Element [1] = 25
Element [2] = 25
Element [3] = 25
Element [4] = 25
```

```
#include <stdio.h>
#define SQUARE(y) (y)*(y)

int main() {
    printf("%d\n", SQUARE(2));
    int z = 6;
    printf("%d\n", SQUARE(z+1));
  return 0;
}
```

Output:

4

49

Suppose the statement:

`cout<<"Element ["<< i <<" ] = "<<  num[i] << endl;` is replaced by the statement:

```
cout<<"Element ["<< i <<" ] = "<<  num[4] << endl;
```

The output on the screen:

```
Element [0] = 36
Element [1] = 36
Element [2] = 36
Element [3] = 36
Element [4] = 36
```

Data Culture

The collective behaviors, views, attitudes and convections of individuals in an enterprise organization who respect, value, practice and support the use of data to improve data-driven decision-making and solve data-related issues

If the condition:

`i<5`

is replaced by the condition:

`i<=5`

Parallel Programming

(The programming paradigm)

Divides a task into subtasks and focuses on completing them all at once

Then the **output on the screen** is:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
Element [5] = 3656

3656 is the number stored in the memory i.e., any number stored in the memory will be displayed.
```

If the statement:

`int num [5] = {16, 18, 20, 25, 36};` is replaced by the statement:

```
int num [i] = {16, 18, 20, 25, 36};
```

Then the **compilation error** will be displayed on the screen because there are 5 elements within the braces {} not `i` elements.

C Program:

```
#include <stdio.h>

int main() {
    int a =13;
    (a&1)?printf("Odd"):printf("Even");
    return 0;

}
```

Output:

Odd

C++ Program:

```
#include <iostream>
using namespace std;

int mult(int x, int y){
    return x*y;
}

int mult(int x, int y, int z)
{
    return x*y*z;
}

int main() {
    cout<<mult(4,5)<<endl;
    cout<<mult(2,6,1)<<endl;
return 0;

}
```

Output:

20

12

```
Java Program:
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Albert Einstein");

    }
    public static void main(String args) {
        System.out.println("Alan Turing");
    }
}
```

Output:

Albert Einstein

- **C++ program to print the sum of the elements in array.**

```
#include<iostream>
using namespace std;
int main()
{
int i, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
cout<<"Sum of the Elements in the array = "<< sum;
return 0;
}
```

Management Information Systems

The method of delivering appropriate data to conduct day-to-day business operations and track an enterprise organization's growth

The output on the screen:

```
Sum of the Elements in the array = 115
i.e., 16 + 18 + 20 + 25 + 36 = 115
```

Collective intelligence

The act of a large group of individuals gathering and sharing their ideas, opinions, knowledge, experiences, information and abilities in order to solve diverse problems in society

If the statement:

```
int i, sum = 0;
```

is replaced by `int i, sum = 1;`

Then The **output on the screen**:

```
Sum of the Elements in the array = 116
```

- **C++ program to print the average of the elements in array**

```
#include<iostream>
using namespace std;
int main()
{
int i, avg, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num [i];
avg = sum/5;
cout<<"Sum of the Elements in the array = "<< sum;
cout<<"average of the elements in the array= "<< avg;
return 0;
}
```

Python code:

```
x = [11, 12, 13, 14, 15, 16]
print(x[-1])
print(x[-2])
```

Output:

```
16
15
```

The output on the screen:

```
Sum of the Elements in the array = 115

average of the elements in the array = 23
```

- **Write a program to print:**

Einstein [0] = E

Einstein [1] = I

Einstein [2] = N

Einstein [3] = S

```
import pandas as pd
a = {'Albert' : 1879, 'Turing' : 1912, 'Newton' : 1643}
b = pd.Series(a)
print (b)
```

```
Albert    1879
Turing    1912
Newton    1643
dtype: int64
```

```
import pandas as pd
a = {'Albert' : 18.79, 'Turing' : 19.12, 'Newton' : 16.43}
b = pd.Series(a)
print (b)
```

```
Albert    18.79
Turing    19.12
Newton    16.43
dtype: float64
```

Einstein [4] = T

Einstein [5] = E

Einstein [6] = I

Einstein [7] = N

using arrays

C program:

```c
#include <stdio.h>

int main() {
    int x=0;
    while (10 - x++)
    printf ("\n %d", x);
    return 0;
}
```

Output:

1
2
3
4
5
6
7
8
9
10

Answer:

```
#include<iostream>
using namespace std;
int main()
{
int i;
char name [8] = {'E' , 'I', 'N', 'S', 'T', 'E', 'I', 'N'};
for(i=0; i<8; i++)
cout<<"Einstein ["<< i <<" ] = "<< name[i] << endl;
return 0;
}
```

- **What will be the output of the following programs?**

i)

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
cout<<""<< cbrt(27);
return 0;
```

```
}
```

Answer:

3

ii)

```
#include <iostream>
using namespace std;
int main()
{
char i;
char body [4] = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
cout<<"\n body ["<<body[i] <<" ] = "<< body[i] << endl;
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
    int a = 19, b = 2;
    double c;
    c = (double) a / b;
    printf("%f\n", c );
    return 0;
    }
```

Output:

9.500000

Answer:

body [b] = b

body [o] = o

body [d] = d

body [y] = y

iii)

```
#include <stdlib.h>
#include <stdio.h>
enum {false, true};
int main() {
    int x = 10;
    do {
        printf("%d\n", x); x+1;
        if (x <= 12)
        continue;
    } while (false);
    getchar();
    return 0;

}
```

Output:

10

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main()
{
int x=2;
cout<<""<< malloc (200*sizeof(x));
return 0;
}
```

Answer:

8183824

- **Program 7.7**

  C++ program to print the output:

```
Name of the book = B
Price of the book = 135.00
Number of pages = 300
Edition of the book = 8
```

  using structures

```
#include<iostream>
using namespace std;
int main()
{
struct book {
char name;
float price;
int pages;
```

```
#include <stdio.h>
int main() {
    for (int a=1; a<=3; a++)
    printf ("%d", a*a);
    return 0;
}
```

```
#include <stdio.h>
int main() {
    int a=1;
    while (a<=3) {
        printf ("%d", a * a);
        a++;
    }
    return 0;
}
```

Output:

1 4 9

```
#include <stdio.h>
#include <string.h>
int main () {
   int x, y;
   x=strlen("Albert");
   y=sizeof("Albert");
   printf("%d %d", x, y);
   return 0;
}
```

Output:

6 7

```
int edition;
};
struct book b1= {'B', 135.00, 300, 8};
cout<<"Name of the book = "<< b1.name<< endl;
cout<<"Price of the book = "<< b1.price<<endl;
cout<<"Number of pages = "<< b1.pages<<endl;
cout<<"Edition of the book = "<< b1.edition<< endl;
return 0;
}
```

The output on the screen:

```
Name of the book = B
Price of the book = 135.00
Number of pages = 300
Edition of the book = 8
```

- **What will be output of the following programs?**

A)

```
#include<iostream>
using namespace std;
struct book {
char name;
float price;
int pages;
int edition;
};
int main()
{
struct book b1;
b1.name = 'C';
b1.price = 135.00;
b1.pages = 300;
```

```
#include <stdio.h>
void num(int, int);
int main() {
int x=56, y=106;
num(x,y);
printf("%d \n", x);
printf("%d \n", y);
return 0;
}
void num(int p, int q) {
p=25;
q=36;
}
```

Calling a function by passing the values of variables

Output:

56

106

```
#include <stdio.h>
void num(int*,int*);
int main() {
int x=56,y=106;
num(&x, &y);
printf("%d \n", x);
printf("%d \n", y);
return 0;
}
void num(int *p, int *q) {
*p=25;
*q=36;
}
```

Calling a function by passing references of variables

Output:

25

36

```
b1.edition = 8;
cout<<"Name of the book = bulgarian "<< b1.name << endl;
cout<<"\n Price of the book = "<< b1.price;
cout<<"\n Number of pages = "<< b1.pages<<endl;
cout<<"\n Edition of the book = "<< b1.edition;
}
```

Answer:

```
Name of the book = B

Price of the book = 135.000000

Number of pages = 300

Edition of the book = 8
```

C program:

```
#include <stdio.h>
#define m(a, b) a##b

int main() {
int ab = 560;
printf("%d", m(a, b));
return 0;
}
```

Output:

560

B)

```
#include <iostream>
using namespace std;
int main() {


for( ; ; ) {
cout<<"This loop will run forever.\n";
}

return 0;
}
```

Answer:

```
This loop will run forever.
This loop will run forever.
This loop will run forever.
```

C program:

```
#include <stdio.h>

int main() {
    int a = 32;
    if (a && ((a & a-1) == 0)) {
    printf ("%d is a power of 2", a);
    }
    else {
        printf ("%d is not a power of 2", a);

    }
    return 0;
}
```

Output:

32 is a power of 2

```
This loop will run forever.
This loop will run forever.
This loop will run forever.

......... continues
```

- **Program 7.8**

  Continue and break statements:

  i)

```
#include <iostream>
using namespace std;
int main()
{
int i;
for (i=1; i<=5; i++)
{
if (i==3)
{
continue;
}

cout<<"\n "<< i;
}
return 0;

}
```

C Program:

```
#include <stdio.h>
#define xyz main

int xyz ()
{
    printf("Albert Einstein");
    return 0;

}
```

Output:

Albert Einstein

Output on the screen:

1

2

4

5

ii)

```
#include <iostream>
using namespace std;
int main()
{
int i;
for (i=1; i<=5; i++)
{
if (i==3)
{
break;
}
cout<<"\n "<< i;
}
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
int num[3]={11,12,13};
for(int x=0;x<3;x++) {
printf("%d", num[x]);
}
return 0;
}
```

Output:

11 12 13

Output on the screen:

```
1
2
```

Access	Public	Protected	Private
Same class	yes	yes	yes
Derived classes	yes	yes	no
Outside classes	yes	no	no

- **Program 7.9**

  C++ program to convert the upper case letter to lower case letter

```
#include<iostream>
using namespace std;
int main()
{
char ch = 'A';
char b = tolower(ch);
cout<<" upper case letter "<< ch <<" is converted to lower case letter "<< b;
return 0;
}
```

Python code:

```
a = "manjunath5496".isalnum()
print(a)
```

Output:

True

Output on the screen:

```
upper case letter A is converted to lower case letter a
```

If you want to enter the ***character*** through the keyboard, then the above program should take the form:

```
#include<iostream>
using namespace std;
int main()
{
char ch;
cout<<"Enter any character:";
cin>>ch;
char b = tolower(ch);
cout<<" upper case letter "<< ch <<" is converted to lower case letter "<< b;
return 0;
}
```

Python code:

```
a = "$manju@123".isalnum()
print(a)
```

Output:

False

Output on the screen:

```
Enter any character:
If you enter the character C
```

```
import re

a = bool(re.match('[A-Za-z0-9]+$','manju1988'))
print(a)

b = bool(re.match('[A-Za-z0-9]+$','$manju@1988'))
print(b)
```

Output:

True

False

```
upper case letter C is converted to lower case letter c will be outputted on the
screen.
```

- **Program 8.0**

  C++ program to convert the lower case letter to upper case letter

```
#include<iostream>
using namespace std;
int main()
{
char ch = 'a';
char b = toupper(ch);
cout<<" lower case letter "<<ch<<" is converted to upper case letter "<<b;
return 0;
}
```

```
import re
def m(date):
    return re.sub(r'(\d{4})-(\d{1,2})-(\d{1,2})', '\\3-\\2-\\1', date)
d = "1988-03-01"
print(m(d))
```

Output:

01-03-1988

Output on the screen:

```
lower case letter a is converted to upper case letter A
```

If you want to enter the character **through the keyboard**, then the above program should take the form:

```
#include<iostream>
using namespace std;
int main()
{
char ch;
cout<<"Enter any character:";
cin>>ch;
char b = toupper(ch);
cout<<" lower case letter "<<ch<<" is converted to upper case letter "<<b;
```

```
from datetime import datetime

a = datetime.strptime("1988-03-01", "%Y-%m-%d").strftime("%d:%m:%Y")
print(a)
```

Output:

01:03:1988

```
return 0;
}
```

```
from collections import Counter

a ={'Albert': 25, 'John': 50, 'Alan': 75}
b = {'Newton': 100, 'Einstein': 125, 'Turing': 150}
c = Counter(a) + Counter(b)
print(c)
```

Output on the screen:

```
Enter any character:

If you enter the character h

lower case letter h is converted to upper case letter H

will be outputted on the screen.
```

```
Counter({'Turing': 150, 'Einstein': 125, 'Newton': 100, 'Alan': 75, 'John': 50, 'Albert': 25})
```

Following is the list of operators which can be overloaded:

+	-	*	/	%	^
&	|	~	!	,	=
<	>	<=	>=	++	--
<<	>>	= =	!=	&&	||
+ =	– =	/=	%=	^=	&=
| =	*=	<<=	>>=	[ ]	()
–>	–>*	new	new [ ]	delete	delete [ ]

Following is the list of operators which cannot be overloaded:

::	.*	.	?:

- **Program 8.1**

C++ program to test whether the entered character is upper case letter or not

```
#include<iostream>
using namespace std;
int main()
{
char ch = 'a';
if(isupper(ch))
cout<<"you have entered the upper case letter";
else
cout<<"you have entered the lower case letter";
return 0;
}
```

Output on the screen:

```
you have entered the lower case letter
```

If the statement:

```
char ch = 'a';
```

is replaced by the statement:

```
char ch = 'A';
```

Then the output on the screen is:

you have entered the upper case letter

C program:

```
#include <stdio.h>
int main() {
int x = 4, y = 4;
printf("!(x == y) is %d \n", !(x == y));
return 0;
}
```

Output:

!(x = = y) is 0

- **Program 8.2**

  C++ program to test whether the entered character is lower case letter or not

```
#include<iostream>
using namespace std;
int main()
{
char ch = 'a';
if(islower(ch))
cout<<"you have entered the lower case letter";
else
cout<<"you have entered the upper case letter";
return 0;
}
```

```
#include <stdio.h>
int main() {
int x = 4, y = 4, z = 5;
printf("(x != y) || (z < y) is %d \n", (x != y) || (z < y));
return 0;
}
```

Output:

(x != y) || (z < y) is 0

Output on the screen:

```
you have entered the lower case letter
```

- **Program 8.3**

C++ program to print the value of tan inverse x (i.e., the value of $\tan^{-1} x$)

```
#include<iostream>
#include<math.h>
using namespace std;
int main()
{
int x = 20;
cout<<"the value of tan inverse x = "<< atan(x);
return 0;
}
```

```
#include <stdio.h>
int main() {
int x = 4, y = 4, z = 5;
printf("(x == y) || (z < y) is %d \n", (x == y) || (z < y));
return 0;
}
```

Output:

(x = = y) || (z < y) is 1

Output on the screen:

```
the value of tan inverse x = 1.520838
```

- **Program 8.4**

C++ program to print the value of tan inverse $\frac{x}{y}$ (i.e., the value of $\tan^{-1}\frac{x}{y}$)

```
#include<iostream>
#include<math.h>
using namespace std;
int main()
{
int x, y;
x = 20;
y =20;
cout<<"the value of tan inverse x/y = "<< atan2(x,y);
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
int x = 4, y = 4, z = 5;
printf("(x == y) && (z > y) is %d \n", (x == y) && (z > y));
return 0;
}
```

Output:

(x = = y) && (z > y) is 1

Output on the screen:

```
the value of tan inverse x/y = 0.785398
```

- **Program 8.5**

C++ program to print the value of fmod(x, y)

```
#include<iostream>
#include<math.h>
using namespace std;
int main()
{
float x = 20.500000;
float y =20.799999;
cout<<" the remainder of "<<x <<" divided by "<<y <<" is: "<< fmod(x,y);
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
int x = 4, y = 4, z = 5;
printf("%d <= %d is %d \n", x, y, x <= y);
printf("%d <= %d is %d \n", x, z, x <= z);
return 0;
}
```

Output:

4 <= 4 is 1

4 <= 5 is 1

Output on the screen:

```
the remainder of 20.500000 divided by 20.799999 is 20.500000
```

- **Program 8.6**

  C++ program to print the value of ~x

```
#include<iostream>
using namespace std;
int main()
{
int x, y;
x = 205;
y=~x;
cout<<"the value of y is: "<< y;
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
int x = 4, y = 4, z = 5;
printf("%d >= %d is %d \n", x, y, x >= y);
printf("%d >= %d is %d \n", x, z, x >= z);
return 0;
}
```

Output:

4 >= 4 is 1

4 >= 5 is 0

Output on the screen:

```
the value of y is:-206
```

C program:

```
#include <stdio.h>
int main() {
int x = 4, y = 4, z = 5;
printf("(x == y) && (z < y) is %d \n", (x == y) && (z < y));
return 0;
}
```

If the statement:

`y=~x;` is replaced by the statement:

`y= -(~x);`

Then the output on the screen is:

Output:

(x == y) && (z < y) is 0

```
the value of y is: 206
```

What will be the output of the following programs:

i)

```
#include<iostream>
using namespace std;
int main()
{
int i = 54;
int y = i<<1;
cout<<"The value of y = "<< y;
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
    int x = 4, y = 4, z = 5;
    printf("%d == %d is %d \n", x, y, x == y);
    printf("%d == %d is %d \n", x, z, x == z);
    return 0;
}
```

Output:

4 = = 4 is 1

4 = = 5 is 0

Answer:

`The value of y = 108`

If the statement:

`i<<1` is replaced by the statement: `i<<2`

Then the **output on the screen** is:

```
The value of y = 216
```

ii)

```
#include<iostream>
using namespace std;
int main()
```

C program:

```
#include <stdio.h>
int main() {
    int x = 4, y = 4, z = 5;
    printf("%d > %d is %d \n", x, y, x > y);
    printf("%d > %d is %d \n", x, z, x > z);
    return 0;
}
```

Output:

4 > 4 is 0

4 > 5 is 0

```
{
int i = 54;
int y = i>>1;
cout<<"The value of y = "<< y;
return 0;
}
```

C program:

```
#include <stdio.h>
int main() {
    int x = 4, y = 4, z = 5;
    printf("%d < %d is %d \n", x, y, x < y);
    printf("%d < %d is %d \n", x, z, x < z);
    return 0;
}
```

Output:

```
4 < 4 is 0
4 < 5 is 1
```

Answer:

The value of y = 27

If the statement:

i>>1 is replaced by the statement: i>>2

Then the **output on the screen** is:

```
The value of y = 13
```

C program:

```
#include <stdio.h>
int main() {
    int x = 4, y = 4, z = 5;
    printf("%d != %d is %d \n", x, y, x != y);
    printf("%d != %d is %d \n", x, z, x != z);
    return 0;
}
```

Output:

```
4 != 4 is 0
4 != 5 is 1
```

- **Program 8.7**

  C++ program to print the length of the entered character (i.e., to print the length of the string)

```
#include<iostream>
#include<string.h>
using namespace std;
int main()
{
```

```
char ch[4];
cout<<"Enter any word: ";
cin>>ch;
cout<<"The length of the string = "<< strlen(ch);
return 0;
}
```

Dogpile effect

(A sort of cascading failure)

occur

When a cache expires and a website receives a large number of requests at the same time

Output on the screen:

```
Enter any word:
If you enter the word dog

The length of the string = 3

will be displayed on the console screen because there are three letters in the word dog.
```

Suppose if you enter the **word** tech

The length of the string = 4

will be displayed on the console screen because there are four letters in the word tech.

Resilience testing

A technique of software testing that focuses on how well an software application performs under pressure or in uncontrolled chaotic situations

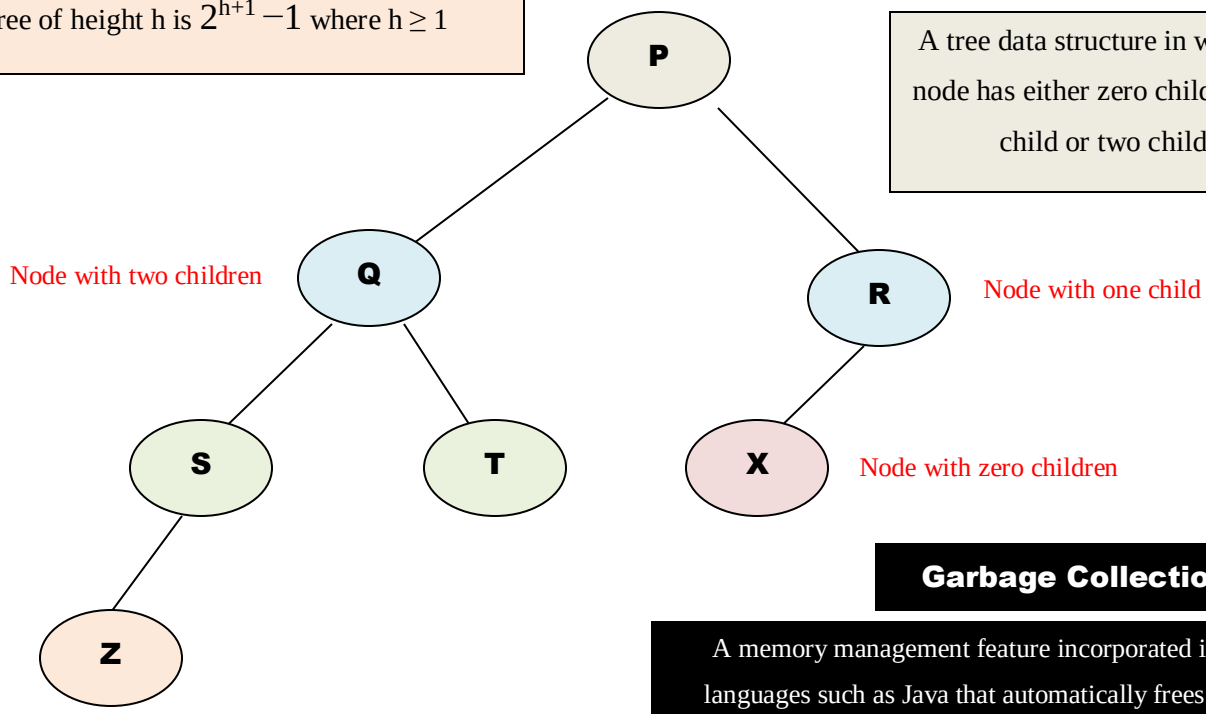
Database Sharding

The method of dividing huge tables into smaller bits known as shards and distributing them across several servers

Binary tree

The **maximum number of nodes** in a binary tree of height h is $2^{h+1} - 1$ where $h \geq 1$

A tree data structure in which each node has either zero children or one child or two children



Garbage Collection

A memory management feature incorporated into programming languages such as Java that automatically frees up memory space assigned to objects that the program no longer requires

Unlabeled Binary Tree → nodes are unlabeled

Number of unlabeled binary trees possible with 'x' unlabeled nodes = ${}^{2x}C_x$ / x +1

For x = 3:

$${}^{2x}C_x = \frac{2x!}{x!\,(2x-x)!} = \frac{6!}{3!\,(6-3)!} = \frac{6\times5\times4\times3\times2\times1}{(3\times2\times1)\times(3\times2\times1)} = 20$$

Number of unlabeled binary trees possible with '3' unlabeled nodes = $\frac{20}{(3+1)}$ = 5

Labeled Binary Tree → nodes are labeled

Number of labeled binary trees possible with 'x' labeled nodes = $({}^{2x}C_x$ / x +1) × x!

For x = 3:

Number of labeled binary trees possible with '3' labeled nodes = 5 × 3! = 5 × 3 × 2 × 1 = 30

- **Program 8.8**

  C++ program to print the factorial of the entered number

```
#include<iostream>
using namespace std;
int main()
{
int i, n, fact=1 ;
cout<<"Enter any number:";
cin>>n;
for(i=1; i<=n; i++)
fact = fact *i;
cout<<"\n Entered number is: "<< n;
cout<<"\n The factorial of the entered number"<<n<<"is:"<< fact;
return 0;
}
```

C program:

```
#include <stdio.h>
int main () {
    int p = 14;
    float q = p;
    printf("%f", q);
    return 0;
}
```

Output:

14.000000

Output on the screen:

```
Enter any number:
If you enter the number 2
Entered number is: 2
The factorial of the entered number 2 is: 2
will be displayed on the screen.
```

C program:

```
#include <stdio.h>
int main () {
    float p = 14.11;
    int q = (int) p;
    printf("%d", q);
    return 0;
}
```

Output:

14

Suppose if you enter the number 4

```
Entered number is: 4
The factorial of the entered number 4 is: 24
```

will be displayed on the screen.

Each 1 or 0 in a binary number is called a bit

1 nibble = 4 bits

1 byte = 2 nibbles = 8 bits

1 kilobyte (KB) = 1024 bytes

1 megabyte (MB) = 1024 KB

1 gigabyte (GB) = 1024 MB

The Single Responsibility Principle

↓

Each module or class should do one thing and one thing only

The Dilbert Principle:

Ineffective employees are frequently promoted to managerial positions in order to remove them from the workforce

Chesterton's Fence:

Even if it appears redundant or inaccurate at first look, one should try to thoroughly understand the meaning of the code before altering or deleting it

Set x = x + 1

Take the value of x, add 1, and store the result back in the same variable.

Errors

Syntax Errors – The program will not run at all

- **Include:** Forgetting punctuation, misspelling keyword

Logic Errors – The program runs, but does not produce the expected results

- **Include:** Using an incorrect formula, incorrect sequence of statements

If we emailed a marketing survey link to 520 people and only 250 responded, then 520 is the sample survey and 250 is the sample

The Interface Segregation Principle

↓

Clients should not be forced to use interfaces that they are unfamiliar with

Problem solving steps

- Comprehend the issue
- Devise a plan
- Carry out the plan
- Audit the outcomes

The Scout Rule:

Always try to leave the code in a better state than when you found it

Program development steps

- Analyze the problem
- Plan the program
- Code the program
- Test the program

Software

System Software – Programs written for computer systems

(Operating System, Interpreters, Compilers)

Application Software – Programs written for computer users

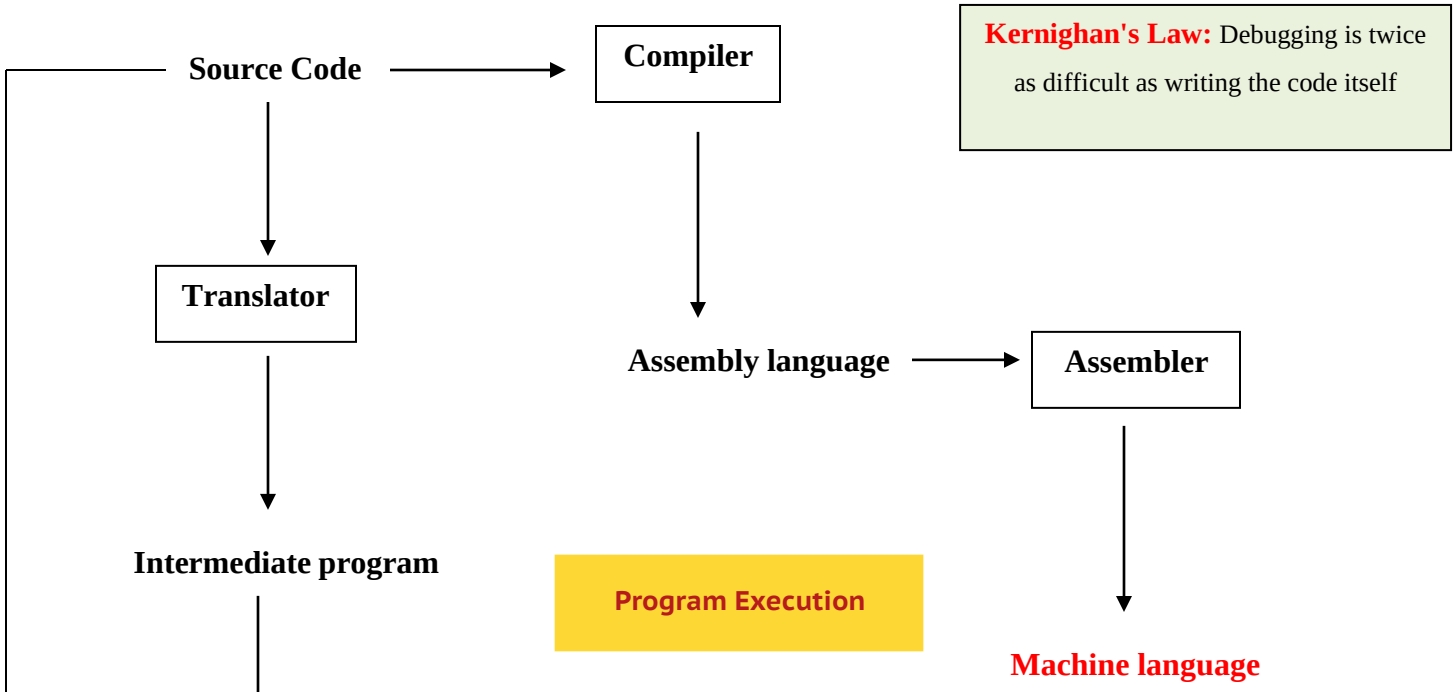
(Word processors, IDEs, Spreadsheets)

George Box's Law

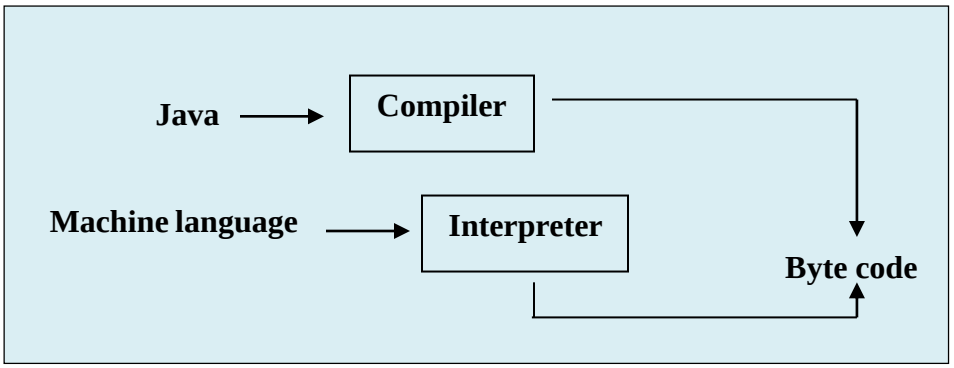
All systems models have flaws, but as long as they don't have too many flaws, they can be useful

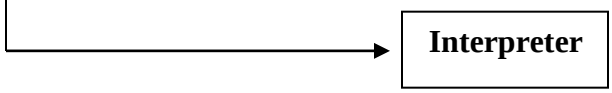
The Spotify Model:

Rather than being organized on technologies, teams are organized around features



Kernighan's Law: Debugging is twice as difficult as writing the code itself





Data abstraction	Control abstraction
The concealment of data details	The concealment of implementation details

The Law of Conservation of Complexity (Tesler's Law):

Each application has a certain level of inherent complexity that cannot be eliminated or concealed

Premature Optimization

We start optimizing before we even realise we need to

Boolean

Data type that stores two types of values (0 and 1).

- 0 represents **false value** and 1 represents **true value**

Dunbar's number

150

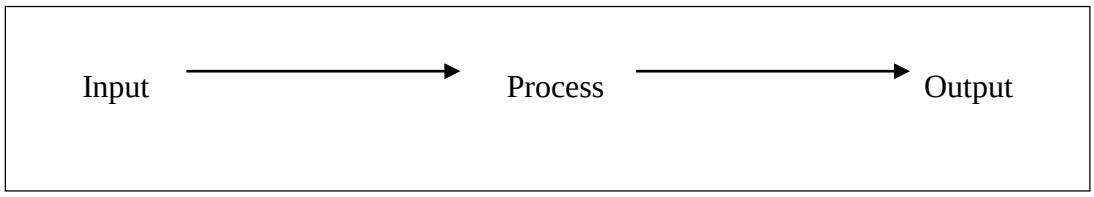
The maximum number of people we can keep track of and include in our ongoing social network

```
#include <stdio.h>
#include <stdbool.h>

int  main()  {
  bool x = false;
  if(x){
    printf("x is true.");
  }
  else{
    printf("x is false.");
  }
}
```

If you don't include this **header file**, the program will not compile

Output on the screen:

```
x is false
```



Control structure

- **Sequence control structure**
- **Selection control structure**

Sequence control structure	Selection control structure
• Go to the Contacts • Dial the Burger number • Order the Burger • Hang up	Open your wallet IF you have enough money, **THEN** • Go to the Contacts • Dial the Burger number. • Order the Burger • Hang up Or **ELSE** Forget the whole thing

- **Source Code →** Read by People
- **Object Code →** Read by the Computer

Assembly Level Language

The instruction

mov al, 061h

implies:

Move the hexadecimal value 61 (**97 decimal**) into the processor register named "al"

Amara's Law

↓

We tend to overestimate the technology's short-term impact and underestimate its long-term impact

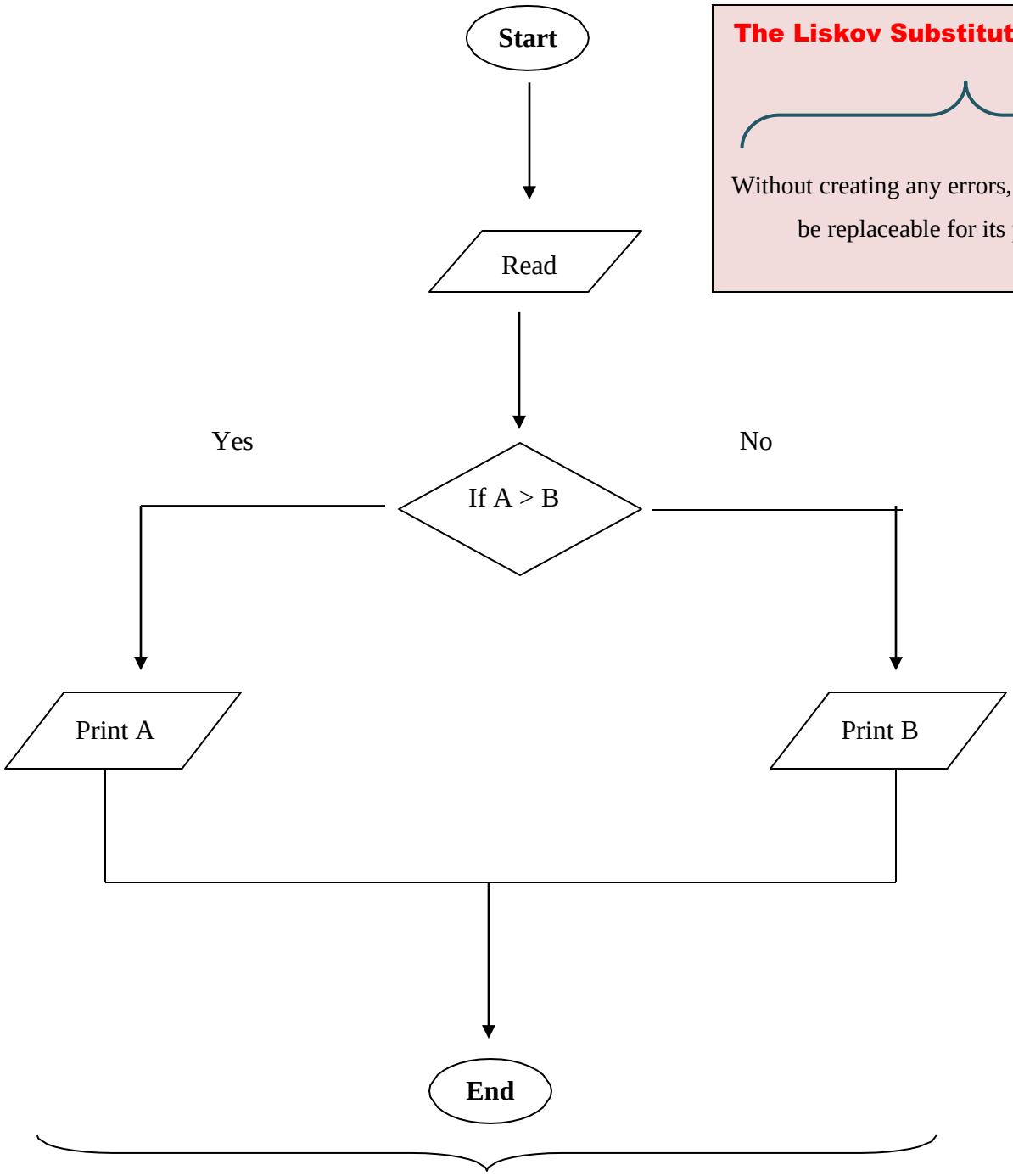
Sarnoff's Law: The utility of a broadcast network rises in direct proportion to the number of viewers

Postel's Law

(Robustness Principle)

Protocol developers should be conservative in what they do and liberal in what they accept from others

Flow Chart to find largest of two numbers:



The Liskov Substitution Principle:

Without creating any errors, a child class must be replaceable for its parent class

Limitations of Flowchart

- If the program logic is quite complicated, then flowchart becomes complex and clumsy.
- In the event that modifications are required the flowchart may require re-drawing totally.As the flowchart symbols can't be typed, reproduction of flowchart turns into an issue.

```
String x = "32";
int y = Integer.parseInt(s1); // y now holds 32
```

```
String x = "32";
double y = Double.parseDouble(s1); // y now holds 32
```

Parkinson's Law:

The amount of work grows in proportion to the amount of time available to complete it

a <<= b → a × 2^b

```
#include<stdio.h>
main()
{
int a, b, c;
a=2;
b=2;
c = a <<= b;
printf("The value of c = %d", c);
}
```

Beckstrom's Laws of Cyber Security:

- Everything connected to the Internet is vulnerable to attack.
- Everything is connected to the Internet.
- Everything else is dependent on the first two laws.

V = B – C – SI – L

- **V** = The utility of a network
- **B** = The benefit a network offers
- **C** = The expense of providing the network
- **SI** = Security investment that a enterprise organization spends in order to minimize losses
- **L** = Actual losses as a consequence of poor security

Output on the screen:

The value of c = 8

a >>= b → a × 2^{-b}

Gall's Law: A complicated system that works is developed from a simpler system that worked

```
#include<stdio.h>
main()
{
int a, b, c;
a=2;
b=2;
c = a <<= b;
printf("The value of c = %%", c);
}
```

The Broken Windows Theory: Poor quality code can give the impression that efforts to improve it are being neglected or devalued, leading to even worse code

Output on the screen:

The value of c = %

90–9–1 Principle

(1% Rule)

Ninety percent of people just consume content, while nine percent alter or modify it, and one percent adds it

Java Simplifications of C++

- no pointers — **just references**
- no functions — declare **static** methods
- no global variables — **public static** variables
- no destructors — **garbage collection** and **finalize**
- no linking — dynamic class loading
- no header files — define **interface**
- no operator overloading — only method overloading
- no member initialization lists — call **super** constructor
- no preprocessor — **static final constants** and automatic inlining
- no multiple inheritance — **implement multiple interfaces**
- no structures, unions, Enums — **typically not needed**

Law of Leaky Abstraction

Details of the abstracted lower level concept are leaking up through the higher level concept

Heap:	Dynamically allocated global memory
Stack:	Local memory for function calls

C++ does not support garbage collection

Java does not support multiple inheritance

Kerckhoff's principle:

In cryptography, a system should be secure even if everything about it is widely known except for a little piece of information called the key

Conway's Law:

Any piece of software is a representation of the organizational structure that created it

Hoare's Law of Large Programs:

Every huge problem contains a little problem that is attempting to escape

Loops

- Unbounded (**indefinite**) loop – while and **do-while** statement
- Bounded (**definite**) loop – for statement

Hofstadter's Law: A project will always take longer than expected

- **Internal Sort:** The data to be sorted is stored in the computer's main memory
- **External Sort:** The data to be sorted is stored in some external device
- **In Place Sort:** The amount of extra space required to sort the data is constant with the input size

Ninety–ninety rule:

The first 90% of the code consumes 10% of the time to complete

The remaining 10% of the code consumes the remaining 90% of the time to complete

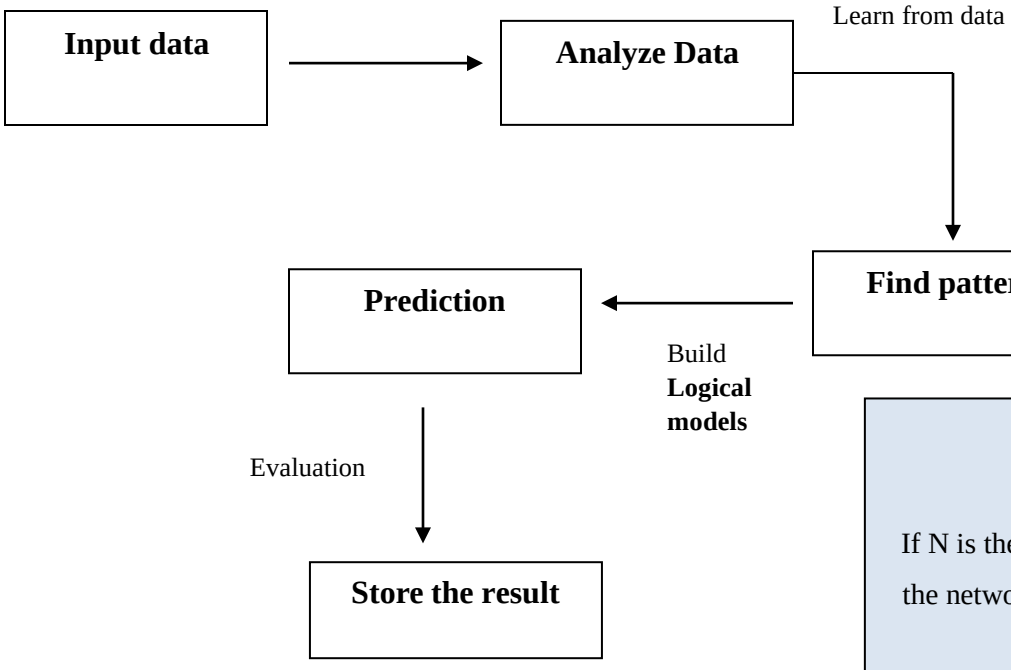
Problem solving methodology

- Problem definition
- Problem analysis
- Design the problem
- Coding
- Program testing and Debugging
- Documentation
- Program maintenance

Hick-Hyman Law:

The higher the number of options presented to users, the longer it will take them to make a selection

How does Machine learning Work?



Metcalfe's Law:

If N is the number of network users, then the network's utility is proportional to N^2

7 Steps of Machine Learning:

- Gather the Data
- Prepare that data
- Choose a model
- Train the model
- Evaluation
- Hyperparameter Tuning
- Prediction

Linus's Law:

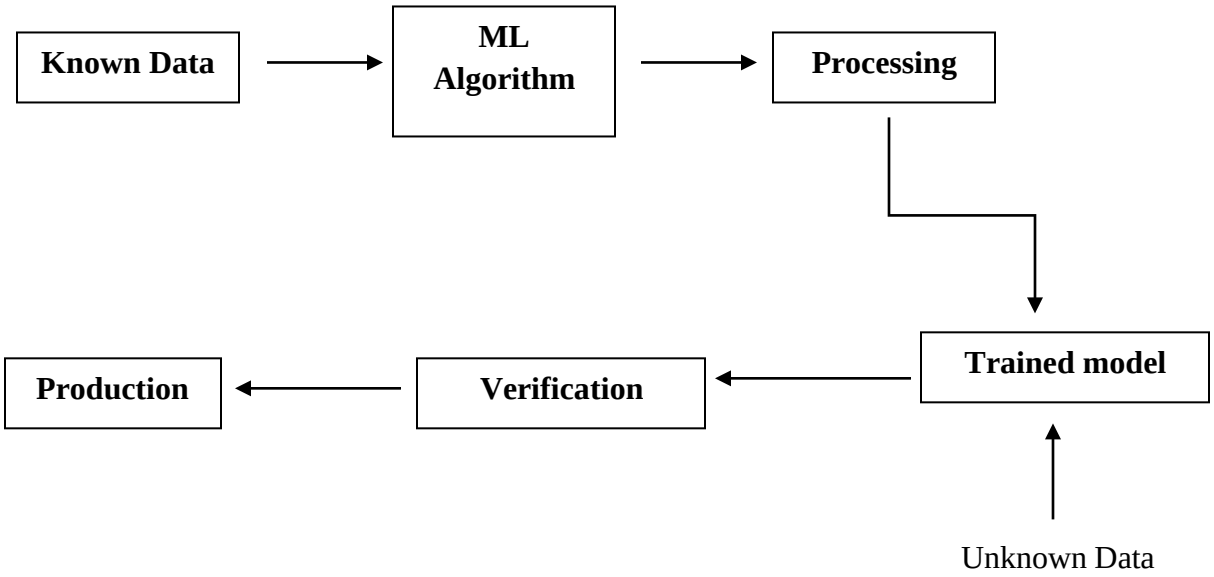
Almost any problem will be promptly identified and the solution will be evident to someone if you have a large enough beta-tester and co-developer team

Brooks' Law

Adding workforce to a late software project only delays it

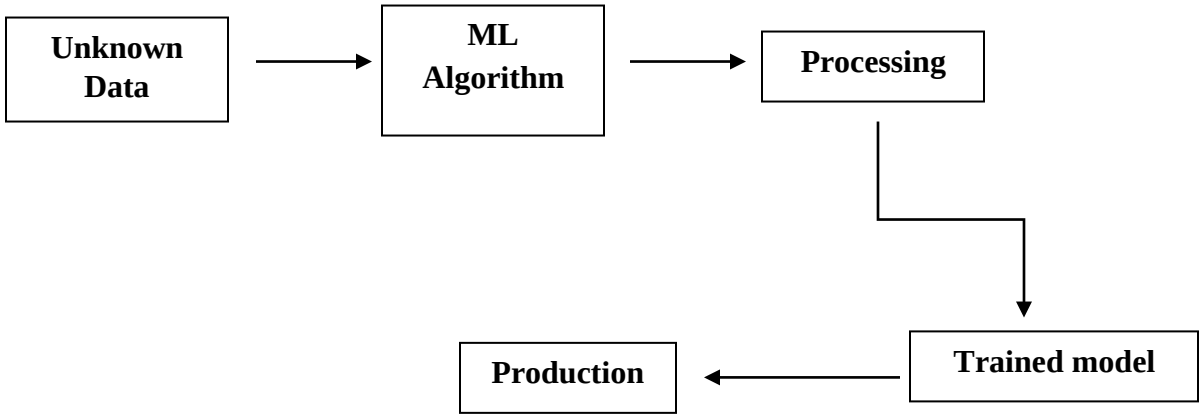
3 Main techniques of Machine Learning:

- **Supervised Learning**



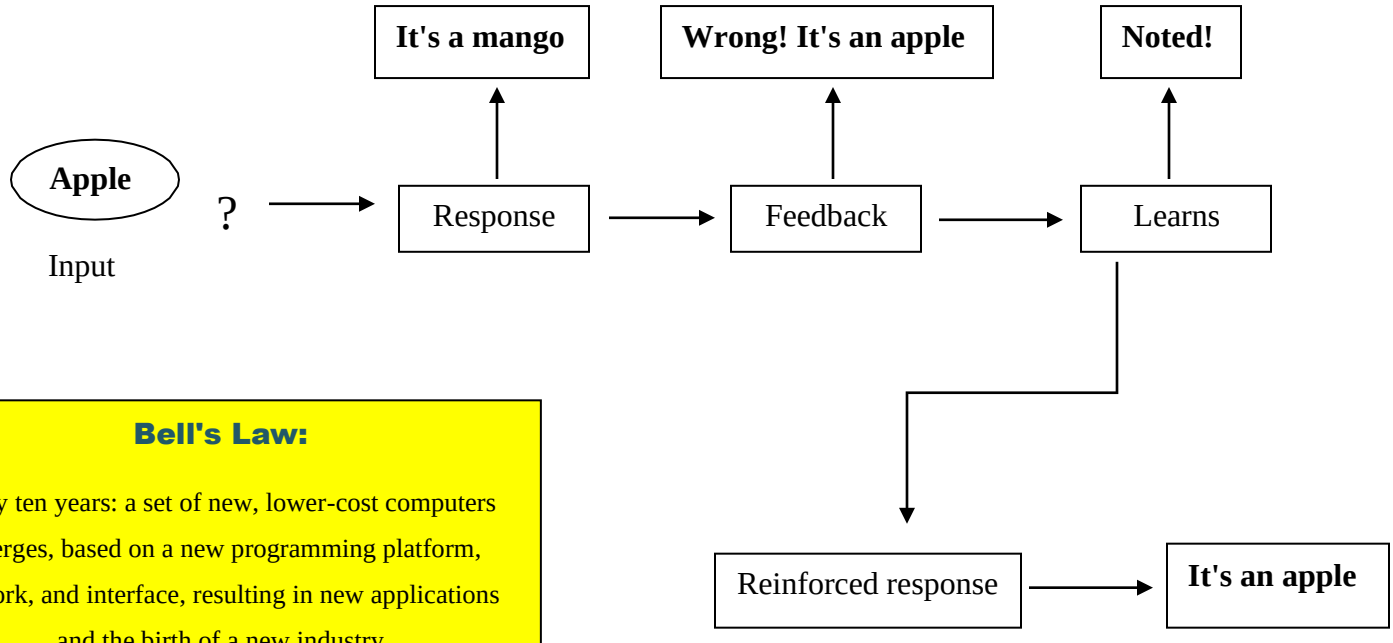
- **Unsupervised Learning**

Gilder's Law: Every twelve months, the total bandwidth of communication systems triples



Wirth's Law: Software slows down more quickly than hardware speeds up

Reinforcement Learning (Learn from mistakes):



Bell's Law:

Every ten years: a set of new, lower-cost computers emerges, based on a new programming platform, network, and interface, resulting in new applications and the birth of a new industry

Hypponen's Law: Every time a device is described as "smart," it is vulnerable

Fitt's Law:

The time it takes for a pointer to move from its present position to a target is proportional to the target's distance and size

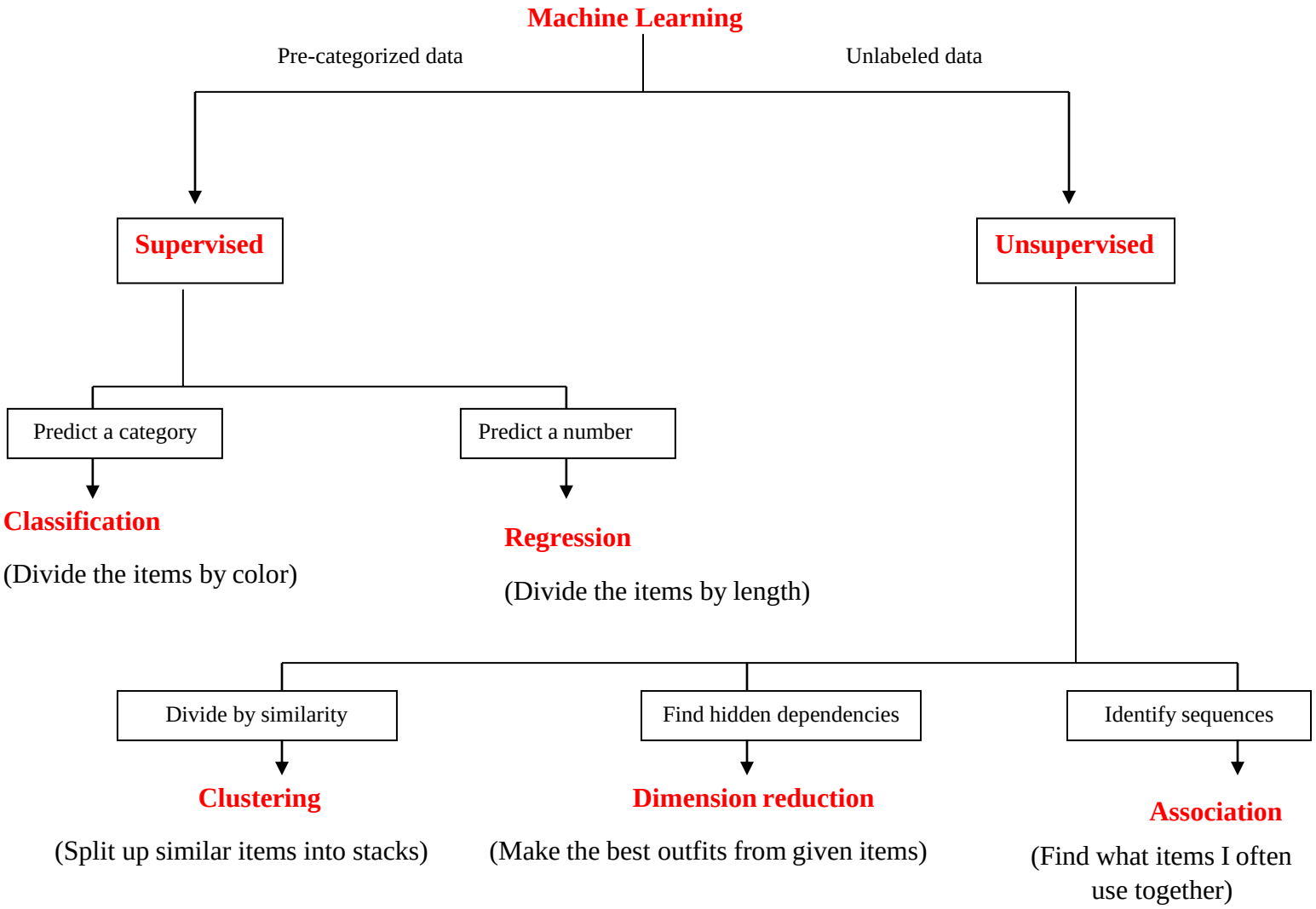
Cunningham's Law

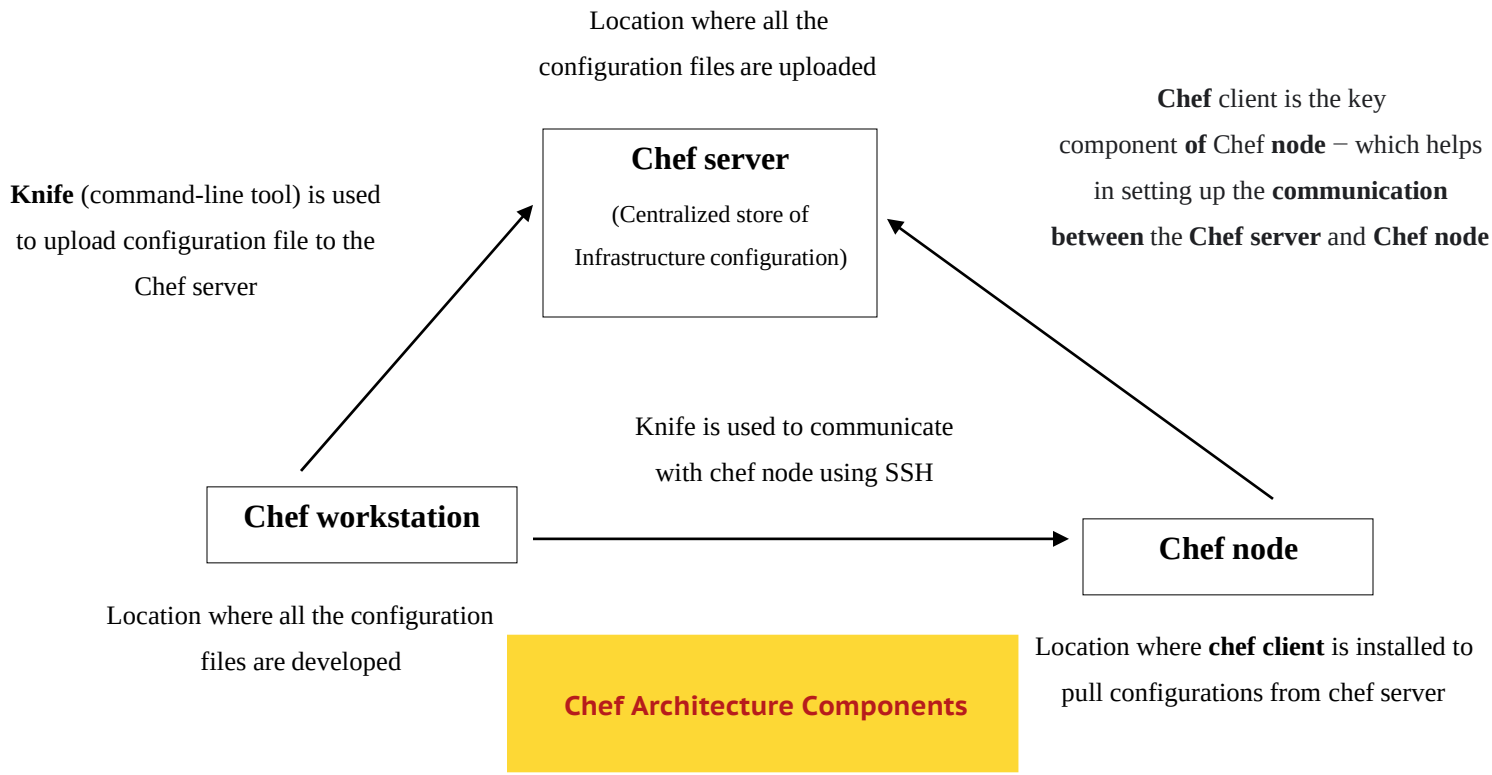
The best approach to acquire the right answer on the Internet is to publish the wrong answer, not to ask a question

Kryder's Law:

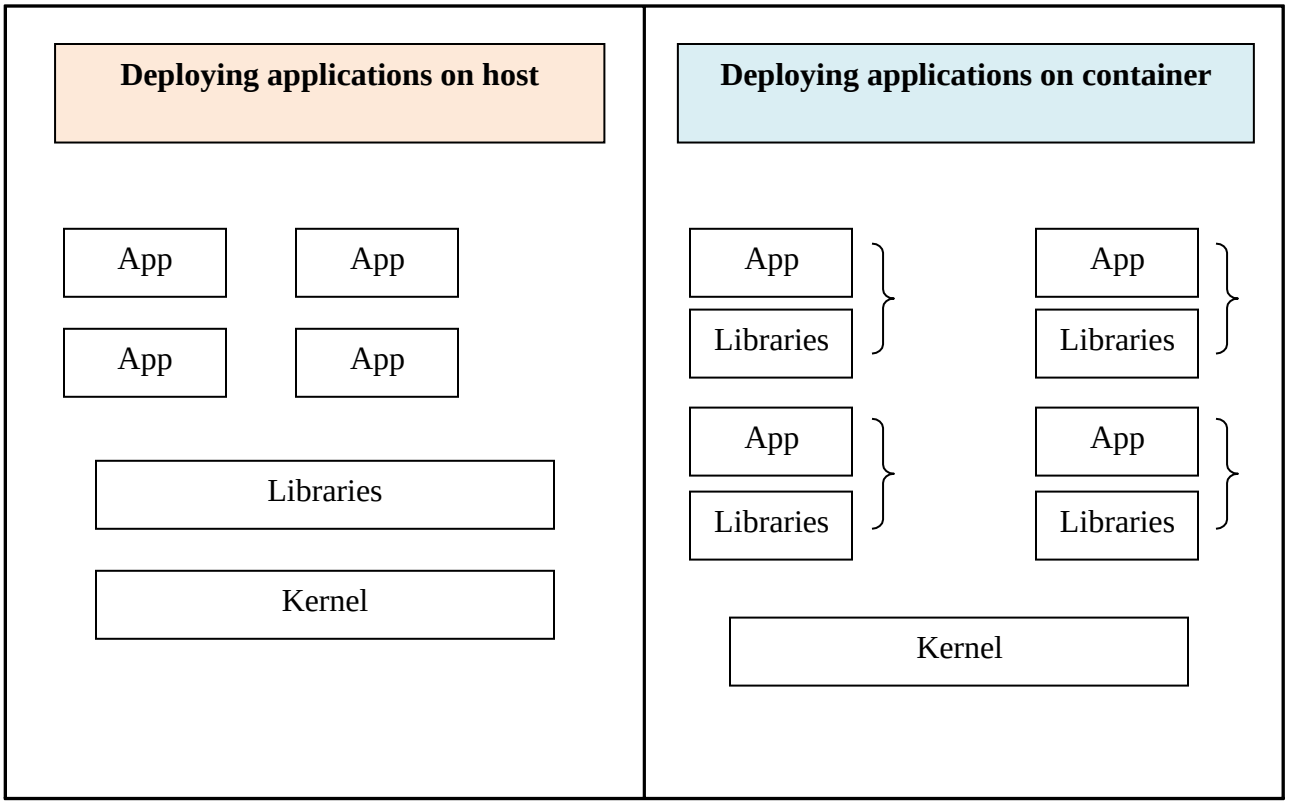
Every thirteen months, the amount of data a disc can hold doubles

Storage will become less expensive as the amount of data a disc can hold increases





Chef is a powerful automation tool that transforms infrastructure into code



- **Deploying applications on host:** All the applications will share the same libraries present in the operating system.
- **Deploying applications on container:** All the applications have the necessary libraries isolated from the rest of the operating system and cannot be intervened by any other application. If one application crashes, other application will keep running flawlessly and won't experience any technical issues. This isolation also decreases security risks: If one application is hacked or breached by malware (including viruses, ransomware and spyware), any resulting negative outcomes won't spread to the other applications.

Docker allows you to create containers

Kubernetes assists you to manage containers

Benefits of infrastructure as code:

Reed's Law:

The utility of large networks grows exponentially as the network grows in size

- Speed
- Consistency
- Accountability
- Increased Efficiency
- Lower Cost

Murphy's Law

↓

Something will go wrong if it can go wrong

Traditional Application Development Process:

Development → Testing → Quality Assurance → Testing → Production

Security applied

DevOps Development Process:

Development → Testing → Quality Assurance → Testing → Production

Security applied

Linux

Developer	Community Linus Torvalds
Written in	C, Assembly language
OS family	Unix-like
Working state	Current
Source model	Open source
Initial release	September 17, 1991; 29 years ago
Marketing target	Cloud computing, embedded devices, mainframe computers, mobile devices, personal computers, servers, supercomputers
Available in	Multilingual
Platforms	Alpha, ARC, ARM, C6x, AMD64, H8/300, Hexagon, Itanium, m68k, Microblaze, MIPS, ND S32, Nios II, OpenRISC, PA-RISC, PowerPC, RISC-V, s390, SuperH, SPARC, Unicore32, x86, XBurst, Xtensa
Kernel type	Monolithic
Userland	GNU
Default user interface	Unix shell
License	GPLv2 and others (the name "Linux" is a trademark)
Official website	www.linuxfoundation.org

If Microsoft ever does applications for Linux it means I've won.

Linus Torvalds

Linus Torvalds

Creator of Linux

What is Linux and why is it so popular?

Whether you know it or not you are already using Linux (the best-known and most-used open source operating system) every day. From supercomputers to smartphones, the Linux operating system is everywhere. As an operating system, **Linux** is a family of open source Unix-like software based on the Linux kernel - that sits underneath all of the other software on a computer, receiving requests from those programs and relaying these requests to the computer's hardware. With regard to careers, it is becoming increasingly valuable to have Linux skills rather than just knowing how to use Windows. In general, Linux is harder to manage than **Windows**, but offers more flexibility and configuration options.

Every desktop computer uses an operating system. The most popular operating systems in use today are: Windows, Mac OS, and LINUX. Linux is the best-known notoriously reliable and highly secure open source portable operating system -- very much like UNIX -- that has become very popular over the last several years -- created as a task done for pleasure by Linus Torvalds -- computer science student at the **University of Helsinki** in Finland -- in the early 1990s and later developed by more than a thousand people around the world.

Linux is fast, free and easy to use, that sits underneath all the other software on a computer − runs your computer -- handling all interactions between you and the hardware i.e., whether you're typing a letter, calculating a money budget, or managing your food recipes on your computer, the Linux operating system (similar to other Operating Systems, such as Windows XP, Windows 7, Windows 8, and Mac OS X) provides the essential air that your computer breathes.

Linux is the most important technology advancement of the twenty-first century and Licensed under the General Public License (GPL) that Linux uses ensures that the software will always be open to anyone and whose source code is open and available for any user to check, which makes it easier to find and repair vulnerabilities and it power the laptops, development machines and

servers at Google, Facebook, Twitter, NASA, and New York Stock Exchange, just to name a few. Linux has many more features to amaze its users such as: Live CD/USB, Graphical user interface (X Window System) etc.

Build-to-order: A manufacturing methodology in which a product is manufactured only when a confirmed customer order has been received

Build-to-plan: A manufacturing methodology in which a product is manufactured without regard to consumer orders

Why LINUX?

Although **Microsoft Windows** (which is the most likely the victim of viruses and malware) has made great improvements in reliability in recent years, it considered less reliable than Linux. Linux is notoriously reliable and secure and it is free from constant battling viruses and malware (which may affect your desktops, laptops, and servers by corrupting files, causing slow downs, crashes, costly repairs and taking over basic functions of your operating system) – and it keep yourself free from licensing fees i.e., zero cost of entry ... as in free. You can install Linux on as many reliable computer ecosystems on the planet as you like without paying a cent for software or server licensing. While **Microsoft Windows** usually costs between $99.00 and $199.00 USD for each licensed copy and fear of losing data.

CD-ROM

(Compact disc read-only memory)

↓

High-capacity optically read-only memory in the form of a compact disc

Below are some examples of where Linux is being used today:

- Android phones and tablets
- Servers
- TV, Cameras, DVD players, etc.
- Amazon
- Google
- U.S. Postal service
- New York Stock Exchange

Checkpoint: The status of a data transfer is recorded in order to allow the data transfer to be restarted if it is ever interrupted.

Clipboard

(Pasteboard)

↓

A temporary storage area provided by the system to hold data

Linux Operating System has primarily three components:

Collaborative management: A working partnership between Internet commerce partners and Internet service providers to ensure that commercial transactions are completed successfully

- **Kernel**

Kernel is the core part of Linux Operating System and interacts directly with hardware. It is responsible for all major activities of the Linux operating system.

- **System Library**

System libraries are special programs using which application programs accesses Kernel's features.

- **System Utility**

System Utility programs are responsible to do specialized tasks.

Closedown

The process of turning off a device

Important features of Linux Operating System:

- Portable
- Open Source
- Multi-User
- Multiprogramming
- Hierarchical File System
- Security

The number of bits used to represent one character of data

Clocking: A method of managing the number of **data bits** transferred over a telecommunication line in a particular amount of time

Now Linux (successfully being used by several millions of users worldwide) has grown passed the stage where it was almost exclusively an academic system, useful only to a handful of people with a technical background. It provides more than the operating system: there is an entire infrastructure supporting the chain of effort of creating an operating system, of making and testing programs for it, of bringing everything to the users, of supplying maintenance, updates and support and customizations, runs on different platforms including the Intel and Alpha platform. Today, Linux is ready to accept the challenge of a fast-changing world to do various

types of operations, call application programs etc. Since the hiring focus is shifting more and more toward DevOps type skills, a Linux skill set will be the types of things that will make you very deployable.

The **command-line interface** is one of the nearly all well built trademarks of Linux. There exists an ocean of Linux commands, permitting you to do nearly everything you can be under the impression of doing on your Linux operating system. Although, this to the end of time creates a problem: by all of so copious commands accessible to manage, you don't comprehend where and at which point to fly learning them, especially when you are learner. If you are facing this problem, and are peering for a painless method to begin your command line journey in Linux, you've come to the right place, we will launch you to a hold of well liked and helpful Linux commands.

Breakpoint: A point in a software program at which the code stops execution

Description:

Display system date and time.

Command:

```
date
```

Persistence: The process of continuing of an effect after the source of the effect has been removed

Description:

Display calendar.

Command:

```
cal
```

Description:

Display date, time and calendar.

Command:

```
date & cal
```

Description:

Display August month 2016 year calendar.

Read operation:

Any action that retrieves data without changing it in any way

Command:

```
cal 8 2016
```

Description:

Used to clear the terminal window.

Command:

```
clear
```

Description:

Exit from the terminal window.

Database mirroring

The idea of developing and keeping numerous copies of a single database for backup, restoration, and performance improvement purposes

Command:

```
exit
```

Cross-compiler: A compiler that runs on one platform but produces executable code for another platform

Description:

Display free and used system memory.

Command:

```
free
```

Description:

Display free and used system memory in bytes.

Command:

```
free  -b
```

Description:

Display free and used system memory in kilobytes.

Command:

```
free  -k
```

Description:

Display free and used system memory in megabytes.

Command:

Access control list: A set of rules that determines who has access to a system resource and who does not

```
free  -m
```

Description:

Change user password.

> **Collation:** A collection of rules that specify how data in a database is ordered and compared

---

Command:

```
passwd
```

Description:

Power-off the machine.

Command:

```
shutdown
```

Description:

Power-off the machine immediately.

> **Grouped Lock Request:** A single operation that simultaneously requests locks on multiple tables or rows

---

Command:

```
shutdown  -h now
```

Description:

Power-off the machine after 10 minutes.

> **Eviction:** A methodology of removing a piece of data from a cache

Command:

```
shutdown  -h +10
```

Description:

Print current working directory.

Command:

```
echo $PWD
```

Description:

Print previous working directory.

Command:

```
echo $OLDPWD
```

Description:

Executes the 11th command in command history.

Command:

```
!11
```

Cache invalidation

The process of replacing or eliminating entries from a cache in a computer system

Fog Computing: Data, computing, storage, networking and applications are all located somewhere in between the data source and the cloud in a decentralized computing infrastructure

Description:

Reveals your command history.

Command:

```
history
```

Description:

Power off or reboot the Operating system.

Command:

```
sudo reboot
```

Description:

Display the IP address of the host.

Command:

```
ip address
```

Description:

List the size of files and directories.

Command:

```
ls  -s
```

Write operation: Any database operation that alters the data contained in the database

Database replication: The act of creating a distributed database by moving data from one server's database to another server's database so that all users have access to the same amount of data

Read-only Transaction

A transaction that makes no changes to the data

Description:

View mounted file systems.

Command:

```
mount
```

Description:

Display the information of disk usage of files and directories.

Command:

```
du
```

Description:

Tells you how long the system has been running.

Command:

```
uptime
```

Description:

Set current date as 02 Nov 1988.

Command:

Dynamic Link Library (DLL):

A collection of small programs and data that can be used by multiple applications at the same time to finish a given task

```
date --set 1998-11-02
```

Description:

Set current time as 12:11:02 IST.

Command:

```
date --set 12:11:02
```

Description:

View Specific Disk Partition in Linux.

Ephemerality: A piece of data or scenario lasting for a very short time

---

Command:

```
fdisk -l /dev/sda
```

Description:

Lists all files and directories in the present working directory.

Command:

```
ls
```

Description:

Report the process information.

Deadlock: A circumstance in which 2 computer applications that share the same resource effectively block each other from using it, causing both applications to stop working

Command:

```
ps
```

Description:

Display disk usage.

Command:

```
df
```

Description:

Display disk usage in gigabytes, megabytes, or kilobytes.

Command:

```
df  -H
```

Description:

Delete every file and every directory.

Command:

```
rm  -r *
```

```
[manju@localhost ~]$ echo "Hello World"
Hello World
[manju@localhost ~]$ echo -n "Hello World"
Hello World[manju@localhost ~]$
```

Description:

Provides a quick overview of the currently running processes.

Command:

```
top
```

Description:

The system performs an immediate reboot.

Command:

```
reboot
```

Description:

Terminate processes without having to log out or reboot.

Command:

```
kill
```

Description:

Change the current working directory.

Command:

```
cd
```

```
[manju@localhost ~]$ ((b=2+2)); echo $b
4
```

Description:

Create a new session on the system.

Command:

```
login
```

Description:

List open files.

Command:

```
lsof
```

Description:

List USB devices.

Command:

```
lsusb
```

Description:

Check the status of the network services.

Command:

```
[manju@localhost ~]$ read number;
If you enter a number: 26
[manju@localhost ~]$ echo "Your age is $number years."
Your age is 26 years.
```

```
service network status
```

```
[manju@localhost ~]$ touch 1.txt
[manju@localhost ~]$ ls 1.txt>STDOUT
[manju@localhost ~]$ cat STDOUT
1.txt
```

Description:

Start the network service.

Command:

```
service network start
```

Description:

Stop the network service.

Command:

```
service network stop
```

Description:

Restart the network service.

Command:

```
service network restart
```

Description:

Report information about the users currently on the machine and their processes.

Command:

```
w
```

```
for i in $( ls /var/ ); do echo $i; done
```

List every file and directory inside the **/var/** directory

Description:

Display the current directory.

Command:

```
pwd
```

Description:

Displays CPU architecture information (such as number of CPUs, threads, cores, sockets, and more).

Command:

```
lscpu
```

Description:

Displays the number of processing units available to the current process.

Command:

```
nproc
```

```
[manju@localhost ~]$ echo -e "\nHello \t World \t Program\n"

Hello   World        Program

[manju@localhost ~]$
```

Description:

The system performs an immediate reboot.

Command:

```
init 6
```

Description:

Power-off the machine.

Command:

```
init 0
```

Description:

List files by date.

Command:

```
ls -lrt
```

Description:

Report information about storage devices such as hard disks, flash drives etc.

Command:

```
lsblk
```

```
[manju@localhost ~]$ a[0]="Alan"; a[1]="Albert"; echo ${a[1]}
Albert
```

Description:

Show exit status of previous command.

Command:

```
echo $?
```

Description:

Lists a few useful info commands.

Command:

```
info
```

Description:

Prints current year's calendar.

Command:

```
cal -y
```

Description:

Check the status of all the services.

Command:

```
[manju@localhost ~]$ a[0]="Alan"; a[1]="Albert"; echo $a
Alan
```

```
service --status-all
```

Description:

Display time in hh:mm:ss.

Command:

```
date +%T
```

Description:

Tells when the user last logged on and off and from where.

Command:

```
last  -1 username
```

Description:

Sort files and directories by extension name.

Command:

```
ls  -X
```

Description:

Display the manual for the pwd command.

```
[manju@localhost ~]$ x=(h f e); echo $x; echo ${x[@]}; echo ${x[*]}
h
h f e
h f e
```

Command:

```
man pwd
```

Description:

Displays information about running processes in the form of a tree.

Command:

```
pstree
```

Description:

Resets your terminal.

Command:

```
reset
```

Description:

Displays What date is it this Friday.

Command:

```
date -d fri
```

```
[manju@localhost ~]$ a="Albert Einstein."; echo ${a/Einstein/Alan}
Albert Alan.
```

```
[manju@localhost ~]$ echo ${s-xyz}; echo ${s:-xyz}
xyz
xyz
```

```
[manju@localhost ~]$ echo ${a+26}
26
```

Description:

Displays the size of each individual file.

Command:

```
du  -a
```

Description:

Display information about the Advanced configuration and power Interface.

Command:

```
acpi
```

Description:

Takes you two folders back.

Command:

```
cd ../..
```

Description:

Takes you to the previous directory.

Command:

```
cd -
```

Description:

Displays a list of shell built-in commands.

Command:

```
help
```

Servlet

A little computer application that extends the functionality of a larger piece of server software

Description:

Lists your last logins.

Command:

```
last yourusername
```

ISDN

(Integrated Services Digital Network)

A technique to send more data over current telephone lines

Description:

Create a new directory called myfiles.

Command:

```
mkdir myfiles
```

Throughput: The maximum amount of data that can be sent over a given connection

Description:

Remove the directory myfiles.

Command:

FDDI

(Fiber Distributed Data Interface)

A reference for transferring data at a rate of roughly 100,000,000 bits per second through optical fiber connections

```
rmdir myfiles
```

Description:

Disable password for a specific user "root1".

Command:

```
passwd -d root1
```

Description:

Switch to user "root1".

Command:

```
sudo su root1
```

Description:

Exit from the terminal window.

Command:

```
logout
```

Description:

Creates a user "root1".

DHCP **(Dynamic Host Configuration Protocol)** A protocol that allows a machine to receive an IP address from a local network server

Packet Switching: The method of transferring data through the Internet

Intranet	**Internet**
A more secure private network ↓ Only the users of the organization can access the information	A less secure public network ↓ Anyone can access the information

Command:

```
useradd "root1"
```

Description:

Assign password to user "root1".

Command:

```
passwd "root1"
```

Description:

Repeats the last command.

Command:

```
!!
```

Description:

Display Who you are logged in as.

Command:

```
whoami
```

```
[manju@localhost ~]$ x=(a b c d e f g h i j k l m n o p q r s t u v w x y z); y=x[10];echo ${!y}
k
```

```
[manju@localhost ~]$ expr 3 \* 12
36
[manju@localhost ~]$ expr 3 \* 1
3
```

Description:

Display the login name of the current user.

Command:

```
logname
```

Description:

Report the name of the kernel.

Command:

```
uname
```

Description:

Print the kernel version.

Command:

```
uname  -v
```

Description:

Print the operating system.

Command:

```
uname  -o
```

```
[manju@localhost ~]$ num=6+2*20; echo $num
6+2*20
[manju@localhost ~]$ declare -i num; num=6+2*20; echo $num
46
```

Description:

Report the machine hardware name.

Command:

```
uname  -m
```

Description:

Print version information and exit.

Command:

```
uname  --version
```

Description:

Print the kernel release.

Command:

```
uname  -r
```

Description:

Report the network node hostname.

Command:

```
uname  -n
```

Description:

Display all port connections (both TCP and UDP).

Command:

```
netstat  -a
```

Description:

Display only TCP (Transmission Control Protocol) port connections.

Command:

```
netstat  -at
```

Description:

Display only UDP (User Datagram Protocol) port connections.

Command:

```
netstat  -au
```

Description:

Display all active listening ports.

Command:

```
netstat  -I
```

Description:

Display all active listening TCP ports.

Command:

```
netstat  -It
```

Description:

Display all active listening UDP ports.

Command:

```
netstat  -lu
```

Description:

Reveal all the information about the current user (user id, username, group id, group name etc.).

Command:

```
id
```

Description:

Reveal all the information about the user "root1" (user id, username, group id, group name etc.).

Command:

```
id  root1
```

Python code:

```
x = 'a bowman'
print('{0:>{1}s}'.format(x, len(x)+4))
print('{0:_^{1}s}'.format(x, len(x)+4))
```

Output

```
 a bowman
__a bowman__
```

Description:

Print the machine's architecture.

Command:

```
arch
```

Description:

Display the list of available fonts.

Command:

```
fc-list
```

```
# prints the default encoding in the sys module
import sys
print(sys.getdefaultencoding())
# Output: utf-8
```

Description:

Create two directories (myfiles, files).

Command:

```
mkdir myfiles files
```

Description:

install apache (**CentOS**).

Command:

```
yum install httpd
```

```
# divmod(7,2) computes quotient and remainder of 7/2
print(divmod(7, 2))
# Output: (3, 1)
# quotient = 3
# remainder = 1
```

Description:

install apache (**Ubuntu**).

Command:

```
apt install httpd
```

Description:

upgrade apache (**CentOS**).

Command:

```
yum update httpd
```

```
x = 12
if x == 28:
  print('x is 28')
elif x == 36:
  print('x is 36')
elif x > 15:
  print('x is greater than 15')
elif x%2 == 0:
  print('x is a multiple of 2')
else:
  print('x...?')

# Output:
    x is a multiple of 2
```

Description:

upgrade apache (**Ubuntu**).

Command:

```
apt update httpd
```

Description:

uninstall apache (**CentOS**).

```
for x in range(3):
 x = x*2
 print(x)

# Output:
              0
              2
              4
```

Command:

```
yum remove httpd
```

Description:

uninstall apache (**Ubuntu**).

Command:

```
apt remove httpd
```

```
x = "https://en.wikipedia.org/wiki/Isaac_Newton"

print(x.split('/'))

# Output: ['https:', '', 'en.wikipedia.org', 'wiki', 'Isaac_Newton']
```

Description:

Display usage summary for the command (date).

Command:

```
date --help
```

```
print("1988".isnumeric())
# Output: True
```

Description:

List active connections to/from system.

Command:

```
ss -tup
```

```
print('''{name} laid the foundations of {subject} and put forward the {law} in 1687.'''.format(subject='Calculus', law='law of gravitation', name='Sir Isaac Newton'))
# Output:
  Sir Isaac Newton laid the foundations of Calculus and put forward the law of
                                gravitation in 1687.
```

Description:

List internet services on a system.

Command:

```
ss -tupl
```

Description:

Display all active UNIX listening ports.

```
name = 'Albert'
if 'b' in name:
    print("Found 'b' in name")
    if name == 'Albert':
       print("Albert Einstein")
    else:
       print("Some other name")
else:
    print("A name without 'b'")

# Output:
       Found 'b' in name
        Albert Einstein
```

Command:

```
netstat  -lx
```

Description:

Display all the active interfaces details.

Command:

```
ifconfig
```

```
def myfunc(x, y=14):
  print(x, y)
myfunc(11) # Output: 11 14
myfunc(11, 12) # Output: 11 12
```

Description:

Display information of all network interfaces.

Command:

```
ifconfig  -a
```

```
x = 56
print(eval('3 + 4*2 + x'))
# Output: 67
print(exec('for i in range(3): print(i)'))
# Output:
                    0
                    1
                    2
                  None
```

Description:

Compare the contents of two files (1.txt, 2.txt).

Command:

```
diff 1.txt 2.txt
```

Description:

Tells you how many lines, words, and characters there are in a file (1.txt).

Command:

```
wc 1.txt
```

```
# prints the widely used UNIX timestamp
import time
print(int(time.time()))
# Output: 1658163944
```

Description:

Compresses file (1.txt), so that it take up much less space.

Command:

```
gzip 1.txt
```

Description:

Uncompresses file (1.txt) compressed by gzip.

Command:

```
gunzip 1.txt
```

```
# prints the path of os module
import os
print(os)
# Output:
<module 'os' from 'C:\\Users\\Manju\\Anaconda3\\lib\\os.py'>
```

Description:

Examine the contents of the file (1.txt).

Command:

```
cat 1.txt
```

```
import json
b = {'x': None,
     'y': 'Alan',
     'z': 'Albert'}
print(json.dumps(b))
# Output: {"x": null, "y": "Alan", "z": "Albert"}
```

Description:

Display calendar.

Command:

```
ncal
```

```
import logging
x = logging.getLogger('alan')
y = logging.getLogger('mary')
print(x is y) # Output: False
z = logging.getLogger('alan')
print(x is z) # Output: True
```

Description:

Removes the file (1.txt).

Command:

```
rm 1.txt
```

```
a = 'x y'
print(a.split())
# Output: ['x', 'y']
print(a.split(' '))
# Output: ['x', '', 'y']
```

Description:

Rename a file named 1.txt to 0.txt.

Command:

```
mv 1.txt 0.txt
```

Description:

Replace the contents of 0.txt with that of 1.txt.

Command:

```
cp 1.txt 0.txt
```

```
import importlib
math = importlib.import_module('math')
print(math.sqrt(4))
# Output: 2.0
```

Description:

Create an empty file (**test.txt**).

Command:

```
touch test.txt
```

```
import sys, imp
x = sys.modules.setdefault('math', imp.new_module('math'))
print(x.sqrt(16))
# Output: 4.0
```

Description:

Print the last 10 lines of a file (1.txt).

Command:

```
tail 1.txt
```

```
def divide(divisor, dividend):
   return dividend // divisor
print(divide(14, 986))
# Output: 70
print(divide(dividend=986, divisor=14))
# Output: 70
```

Description:

Print N number of lines from the file (1.txt).

Command:

```
tail  -n N 1.txt
```

```
from pprint import pprint
pprint('hi')
# Output: hi
```

Description:

Prints the number of words in a file (1.txt).

Command:

```
wc -w 1.txt
```

```
x = ['a', 'am', 'bell', 'david', 'johnson', 'language']
print([i.upper() for i in x if len(i) > 3])
# Output:
                                True
            ['BELL', 'DAVID', 'JOHNSON', 'LANGUAGE']
```

Description:

Prints the number of characters from a file (1.txt).

Command:

```
wc -m 1.txt
```

```
print({21, 22, 23} == {23, 22, 21})
# Output: True
```

Description:

Prints the length of the longest line in a file (1.txt).

Command:

```
wc -L 1.txt
```

```
print(u'john'.translate({ord('j'):None,ord('n'):u'cv'}))

# Output: ohcv

print(u'albert'.translate({ord('a'):None,ord('r'):u'cv'}))

# Output: lbecvt
```

Description:

Print information about usb ports, graphics cards, network adapters etc.

Command:

```
lspci
```

```
import ast
print(ast.literal_eval('54')) # prints 54
print(ast.literal_eval('[54,96]')) # prints [54, 96]
```

Description:

View contents of a file (1.txt).

Command:

```
less 1.txt
```

```
for i in range(5):

  print('{:>+9,.2f}'.format(i))


# Output:
              +0.00
              +1.00
              +2.00
              +3.00
              +4.00
```

Description:

Display calendar (last month, current month, and next month).

Command:

```
cal -3
```

Description:

Compare the contents of three files (1.txt, 2.txt, 3.txt) line by line.

Command:

```
diff3 1.txt 2.txt 3.txt
```

```
# prints the byte order of current architecture

import sys

print(sys.byteorder)

# Output: little
```

Description:

Compare two files (1.txt, 2.txt) line-by-line.

Command:

```
comm 1.txt 2.txt
```

Description:

Perform byte-by-byte comparison of two files (1.txt, 2.txt).

Command:

```
cmp 1.txt 2.txt
```

```
class M16: pass
c = M16()
c.b = 54
print(c.b)
# Output: 54
```

Description:

Prints the CRC checksum and byte count for the file "myfiles.txt".

Command:

```
cksum myfiles.txt
```

Description:

Append contents of files (1.txt, 2.txt) into one file (0.txt).

Command:

```
cat 1.txt 2.txt > 0.txt
```

Description:

```
x =[2,4,-6,-9,10,11,12]
if all(i>0 for i in x):
 print('All are positive')
else:
 print('Some are negative')
 # Output: Some are negative
```

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt).

Command:

```
sed  r 1.txt 2.txt 3.txt > 0.txt
```

Description:

```
x =[2,4,6,9,10,11,12]
if any(i<0 for i in x):
  print('Some are negative')
else:
  print('None are negative')
# Output: None are negative
```

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt).

Command:

```
sed  h 1.txt 2.txt 3.txt > 0.txt
```

Description:

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt).

Command:

```
sed  -n  p 1.txt 2.txt 3.txt > 0.txt
```

Shortcuts:

```
| ctrl+c           | Halts the current command           |
| ctrl+z           | Stops the current command           |
|                  |                                     |
| ctrl+d           | Logout the current session          |
| ctrl+w           | Erases one word in the current line |
|                  |                                     |
| ctrl+u           | Erases the whole line               |
| ctrl+r           | Type to bring up a recent command   |
```

Description:

Writes contents of a file (0.txt) to output, and prepends each line with line number.

Command:

```
nl 0.txt
```

Description:

Create a empty file (test1.txt) inside a directory (test).

Command:

```
mkdir test
cd test
pwd
touch test1.txt
```

Description:

Gather information about hardware components such as CPU, disks, memory, USB controllers etc.

Command:

```
sudo lshw
```

Description:

Gather information about file system partitions.

Command:

```
sudo fdisk -l
```

Description:

Displays the line (good morning) in which the string (good) is found in the file (1.txt).

Command:

```
grep good 1.txt
```

Description:

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt) using for loop.

Command:

```
for i in {1..3}; do cat "$i.txt" >> 0.txt; done
```

Description:

Search for files (test.txt, test1.txt, test2.txt, test.php, test.html) in a directory as well as its sub-directories.

Command:

```
find test*
```

Description:

Displays status related to a file (1.txt).

Command:

```
stat 1.txt
```

```
###
|     Command       |  Description        |
|:-----------------------|-----------------:      |
| vi                | Open vi editor      |
| i                 | Go to Insert mode   |
|                   |                     |
| a =20; b =64;     |                     |
| print (a + b);    |                     |
| Hit Escape to return to Normal mode.  |
| :w hello.py       | Save text           |
| :q                | Quit                |
| python hello.py   |Print the output:84 |
```

Description:

Download the file (file.txt) from url "http: //website.com/files/file.txt".

Command:

```
wget http://website.com/files/file.txt
```

Description:

Display host's numeric ID in hexadecimal format.

Command:

```
hostid
```

Description:

Display file type of the file (myfiles.txt).

Command:

```
file myfiles.txt
```

Description:

Create a file (myfiles.txt) containing a text (Hello World).

Command:

```
echo 'Hello World' > myfiles.txt
```

Description:

Create a file (myfiles.txt) containing a text (Hello World).

Command:

```
printf 'Hello World' > myfiles.txt
```

Description:

Display IP address of the hostname.

Command:

```
hostname  -i
```

Description:

Add a new line of text to an existing file (1.txt).

Command:

```
echo "Hello world!" >> 1.txt
echo "this is 2nd line text" >> 1.txt
echo "last line!" >> 1.txt
```

Description:

Displays a single line description about a command (cal).

Command:

```
whatis cal
```

```
###
|     Command      |  Description       |
|:---------------------|-----------------:      |
| vi                | Open vi editor     |
| i                 | Go to Insert mode  |
| Type some text.   |                    |
| Hit Escape to return to Normal mode.  |
| :w test.txt       | Save text          |
| :q                | Quit               |
| :q!               |Quit without saving |


###
|     Command      |  Description          |
|:---------------------|-----------------:         |
| vi                | Open vi editor        |
| i                 | Go to Insert mode     |
| $name = "Paul";   |                       |
| print "$name";    |                       |
| Hit Escape to return to Normal mode.     |
| :w hello.pl       | Save text             |
| :q                | Quit                  |
| perl hello.pl     |Print the output: Paul |


###
|     Command                 |  Description           |
|:-----------------------------------|-----------------:          |
| vi                          | Open vi editor         |
| i                           | Go to Insert mode      |
| echo "What is your name?"   |                        |
| read PERSON                 |                        |
| echo "Hello, $PERSON"       |                        |
| Hit Escape to return to Normal mode.                 |
| :w hello.sh                 | Save text              |
| :q                          | Quit                   |
| sh hello.sh                 | Output:                |
|                             | What is your name?     |
|                             | If you enter: Zara Ali |
|                             | Hello, Zara Ali        |
```

Description:

Check the network connectivity between host (your connection) and server (Google server).

Command:

The first ever Linux kernel just occupied only 65 KB.

```
ping google.com
```

90% of the public cloud workload is run on Linux distros.

Description:

Find the location of source/binary file of a command (cal).

Command:

```
whereis cal
```

```
[manju@localhost ~]$ test 15 -gt 12 && echo "True" || echo "False"
True
```

Because 15 is greater than 12, this command prints the text "True"

```
[manju@localhost ~]$ test 5 -lt 12 && echo "True" || echo "False"
True
```

Because 5 is lesser than 12, this command prints the text "True"

```
[manju@localhost ~]$ echo "William Hawking" | sed 's/William/Stephen/'
Stephen Hawking
```

Replaces the string 'William' with 'Stephen'

```
[manju@localhost ~]$ test 12 == 12 && echo "True" || echo "False"
True

[manju@localhost ~]$ test 5 == 12 && echo "True" || echo "False"
False

[manju@localhost ~]$ echo 'Hello World' > file.txt
[manju@localhost ~]$ cat file.txt
Hello World
```

```
[manju@localhost ~]$ awk '{print $0}' file.txt
Hello World
[manju@localhost ~]$ awk '{print $1}' file.txt
Hello
[manju@localhost ~]$ awk '{print $2}' file.txt
World
```

All of the 500 fastest supercomputers run Linux

echo $$

Display the process id of the current process

echo $!

Display the process id of the background process that recently went away

```
[manju@localhost ~]$ grep "Hello" file.txt
Hello World
[manju@localhost ~]$ grep "Hello" file.txt myfile.txt
file.txt: Hello World
myfile.txt: Hello Linux
```

Display all the lines with the word "Hello" in the file.txt and myfile.txt

```
find -type f
# Find all the files

find -type d
# Find all the directories

find . -name file.txt
# Find file.txt in the current directory
```

```
import os
os.system('ls')
```

```
import subprocess
subprocess.call ('ls')
```

Execution of the linux command "ls" using the python program to list all the files and directories in the current directory

Command:

```
last reboot
```

Description:

Show system reboot history

Command:

```
dmesg
```

Description:

Displays the messages from the kernel ring buffer (a data structure that records messages related to the operation of the kernel**)**

Command:

```
[manju@localhost ~]$ for a in Albert Alan John; do echo $a; done
Albert
Alan
John
```

```
cat /proc/cpuinfo
```

Description:

Display CPU information

```
[manju@localhost ~]$ echo "scale=2;22/7" | bc
3.14
[manju@localhost ~]$ echo "scale=3;22/7" | bc
3.142
```

```
i=1; while [[ $i -le 5 ]] ; do echo "$i";(( i = i+1 )); done
```

Output:

1
2
3
4
5

Command:

```
cat /proc/meminfo
```

Description:

Display memory information

```
[manju@localhost ~]$ let b=14/2; echo "14 / 2 =" $b
14 / 2 = 7
```

Command:

```
lspci -tv
```

Description:

Display PCI (Peripheral Component Interconnect**) devices**

```
[manju@localhost ~]$ let a=5*2; echo "5*2 =" $a
5*2 = 10
```

Command:

```
lsusb -tv
```

Description:

Display USB devices

Ephemeral storage: Any storage medium lasting for a very short time

Command:

```
[manju@localhost ~]$ let c=8-6; echo "8 - 6 =" $c
8 - 6 = 2
```

```
free -h
```

Description:

Display free and used memory (-h for human readable, -m for MB, -g for GB**)**

Command:

```
[manju@localhost ~]$ let a=15+20; echo "15 + 20 =" $a
15 + 20 = 35
```

```
mpstat 1
```

Description:

Display processor related statistics

Command:

```
[manju@localhost ~]$ touch 1.txt 2.txt
[manju@localhost ~]$ ls 1.txt 2.txt > STDERR
[manju@localhost ~]$ cat STDERR
1.txt
2.txt
```

```
vmstat 1
```

Description:

```
[manju@localhost ~]$ echo 14 + 15 = $((14 + 15))
14 + 15 = 29
[manju@localhost ~]$ echo 3 ^ 2 = $[ 3 ** 2 ]
3 ^ 2 = 9
```

Display virtual memory statistics

Command:

```
[manju@localhost ~]$ func() { echo "hello, world" ; }; func
hello, world
```

```
iostat 1
```

Description:

Display Input / Output statistics

Command:

```
watch df -h
```

Description:

Execute "df -h" command, showing periodic updates

Command:

```
ps -ef
```

```
[manju@localhost ~]$ cat 1.txt
Alan
Albert
Hawking
John
[manju@localhost ~]$ file='1.txt'; while read line; do echo $line; done < $file
Alan
Albert
Hawking
John
```

Description:

Display all the currently running processes on the system

```
[manju@localhost ~]$ filename=12.txt; if [ -f "$filename" ]; then echo "File exists" ; else echo "File does not exist"; fi
File does not exist
```

Command:

```
ip a
```

Description:

Display all network interfaces and IP address

Command:

```
[manju@localhost ~]$ ls -lrt | grep ^- | awk 'END{print $NF}'
1.txt
```

Display the Last Updated File

```
dig wikipedia.org
```

Description:

Display DNS information for domain (wikipedia.org)

```
[manju@localhost ~]$ a="Albert Einstein was a German-born theoretical physicist"; echo $a| cut -d ' ' -f 1-3
Albert Einstein was
```

Command:

```
host wikipedia.org
```

Description:

```
[manju@localhost ~]$ for i in $(seq 1 2 5); do echo "Welcome $i times"; done
Welcome 1 times
Welcome 3 times
Welcome 5 times
```

Display the IP address details of the specified domain (wikipedia.org**)**

Command:

```
netstat -nutlp
```

Description:

Display listening Transmission Control Protocol (TCP**) and the User Datagram Protocol (**UDP**) ports and corresponding programs**

Command:

```
rpm -qa
```

Description:

List all installed packages

Command:

```
yum list installed
```

Description:

List all installed packages (CentOS)

Command:

```
for (( ; ; )); do echo "infinite loop [ hit CTRL+C to stop]"; done
```

```
yum info httpd
```

Description:

Display description and summary information about package "httpd" **(CentOS)**

Command:

```
du -ah
```

Description:

Display disk usage for all files and directories in human readable format

Command:

```
[manju@localhost ~]$ echo "$(pwd)"; echo "`pwd`"
/home/manju
/home/manju
```

```
du -sh
```

Description:

Display total disk usage off the current directory

Command:

```
[manju@localhost ~]$ echo {1,2}.txt
1.txt 2.txt
[manju@localhost ~]$ echo {1..5}
1 2 3 4 5
```

```
cd /etc
```

Description:

Change to the /etc directory

Command:

```
ps -A
```

Description:

List the status of all the processes along with process id and PID

Command:

```
#include <stdio.h>
int main()
{
  printf("Hello world\n");
  return 0;
}



gcc Hello.c
```

Hello.c

Description:

Compile the C program saved in Hello.c file

Command:

```
#include <iostream>
int main()
{
std::cout << "Hello world!";
  return 0;
}
```

Hello.cpp

```
g++ Hello.cpp
```

Description:

Compile the C++ program saved in Hello.cpp file

Command:

```
tty
```

Description:

Displays the file name of the terminal connected to standard input

Command:

```
public class MyClass {
public static void main(String [] args) {
System.out.println("Hello, World!");
}
}

javac MyClass.java
```

MyClass.java

Description:

Compile the Java program saved in MyClass.java file using javac compiler

Command:

```
od -b myfiles.txt
```

Description:

Displays the contents of myfiles.txt **file in octal format**

Command:

```
od -c myfiles.txt
```

Description:

Displays the contents of myfiles.txt **file in character format**

Record of change: Recording of modifications made to the system

Command:

```
od -An -c myfiles.txt
```

Description:

Displays the contents of myfiles.txt **file in character format but with no offset information**

Command:

```
csplit myfiles.txt 13 62 101
```

Description:

If the file **myfiles.txt** has 123 lines, the **csplit command** would create four files: the xx00 file would contain lines 1−12, the xx01 file would contain lines 13−61, the xx02 file would contain lines 62−100, the xx03 file would contain lines 101−123

Command:

Error guessing: A software testing methodology that involves guessing the type of error that might occur in the code

```
md5sum myfiles.txt
```

Description:

Prints a 32-character (128-bit) checksum of myfiles.txt **file using the MD5 algorithm**

Command:

```
more myfiles.txt
```

Description:

Displays the content of myfiles.txt **file**

Command:

```
sha1sum myfiles.txt
```

Description:

Prints SHA1 (160-bit) checksum of myfiles.txt **file**

SHA 1 → Secure Hash Algorithm 1

Command:

```
shred myfiles.txt
```

Description:

Overwrites the myfiles.txt **file repeatedly − in order to make it harder for even very expensive hardware probing to recover the data**

Command:

```
cat myfile.txt

01. Einstein
02. Newton
03. Maxwell
04. Tesla
05. Edison

tac myfile.txt

05. Edison
04. Tesla
03. Maxwell
02. Newton
01. Einstein
```

Description:

Print the lines of myfile.txt **in reverse (from last line to first)**

Command:

```
uniq myfiles.txt
```

Description:

Delete repeated lines in the file (myfiles.txt).

Command:

```
chkconfig --list
```

Description:

Displays a list of system services and whether they are started (on**) or stopped (**off**) in run levels 0−6**

Command:

```
halt -p
```

Description:

Power-off the system

Command:

```
xdg-open myfiles.txt
```

Description:

Open a file (myfiles.txt).

Command:

```
lastlog
```

Description:

Prints the details of the last login (login-name, port and last login time**)**

Command:

```
lastlog -t 1
```

```
[manju@localhost ~]$ x=eat.read.write.txt; echo ${x%.*}
eat.read.write
[manju@localhost ~]$ x=eat.read.write.txt; echo ${x%%.*}
eat
[manju@localhost ~]$ x=eat.read.write.txt; echo ${x#*.}
read.write.txt
[manju@localhost ~]$ x=eat.read.write.txt; echo ${x##*.}
txt
```

Description:

Displays the login information (1 day ago**)**

Command:

```
lastlog -u manju
```

```
[manju@localhost ~]$ pgrep bash
4504 # Display process ID of bash

cat /proc/4504/environ

Display the environment variables associated with the process ID 4504
```

Description:

Display lastlog information for a particular user (manju**)**

Command:

```
cat /etc/passwd
```

```
more /etc/passwd
```

```
less /etc/passwd
```

```
getent passwd
```

```
# Print all the files that were accessed within the last 8 days
find . -type f -atime -8 -print

# Print all the files that are having access time exactly 8-days old
find . -type f -atime 8 -print

# Print all the files that have an access time older than 8 days
find . -type f -atime +8 -print

# Print all the files that have an access time older than 8 minutes
find . -type f -amin +8 -print

find . -type f -newer 1.txt -print
```

Print all the files that have a modification time greater than that of the modification time of a given 1.txt file

Description:

List all users on Linux

Command:

```
tail -5 /etc/passwd

head -5 /etc/passwd
```

Description:

List last 5 users on Linux

List first 5 users on Linux

Command:

```
wall "The system will be shutdown in 10 minutes."
```

Description:

The message (The system will be shutdown in 10 minutes.**) will be broadcasted to all users that are currently logged in**

Command:

```
chage -l manju
```

```
find . -type f -size +5k
# Find all files having size > 5 kilobytes
find . -type f -size -5k
# Find all files having size < 5 kilobytes
find . -type f -size 5k
# Find all files having size = 5 kilobytes
```

Description:

List the password and its related details for a user (manju**)**

Command:

```
chage -M 10 manju
```

Description:

Set Password Expiry Date for an user (manju**)**

Command:

```
chage -E "2020-07-30" manju
```

Description:

Set the Account Expiry Date for an User (manju**)**

Command:

```
chage -I 10 manju
```

Description:

Force the user (manju**) account to be locked after 10 inactivity days**

Command:

```
ftp 192.168.42.77
```

Description:

Connect to an FTP server at remote server IP address "192.168.42.77**"**

Command:

ODBC

(Open Database Connectivity)

A common API for interacting with database management systems

```
arp -a
```

Description:

Lists all the peers connected at various interfaces along with their MAC Addresses **and IP addresses**

Command:

Java Database Connectivity (JDBC):

A Java API that allows you to connect to a database

```
dnsdomainname
```

Description:

Recordset: The collection of records and fields that are returned when a query is executed

Display the system's DNS domain name

Command:

```
domainname
```

Description:

Display the name of the domain your machine belongs to

Command:

```
echo 'Hello World!' | base64

Output: SGVsbG8gV29ybGQhCg==
```

Description:

Encode text (Hello World!**) to base64**

Command:

```
echo 'SGVsbG8gV29ybGQhCg==' | base64 -d

Output: Hello World!
```

Description:

Decode (SGVsbG8gV29ybGQhCg==**) to text (**Hello World!**)**

Command:

```
fc-cache -f -v
```

Description:

Build font information cache files

Command:

```
cat 1.txt

Einstein
Newton
Albert

fmt 1.txt

Einstein Newton Albert
```

```
[manju@localhost ~]$ cal
      May 2022
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31

[manju@localhost ~]$ if grep -q 'cal' ~/.bash_history; then echo "You seem to
have typed 'cal' before."; fi

You seem to have typed 'cal' before.
```

Description:

Formats text in a single line

```
cat phy.txt

Albert Einstein was a German-born theoretical physicist, widely acknowledged to
be one of the greatest physicists of all time. Einstein is known for developing
the theory of relativity, but he also made important contributions to the
development of the theory of quantum mechanics.


fmt -w 1  phy.txt

 Albert
 Einstein
 was
 a
 German-born
 theoretical
 physicist,
 widely
 acknowledged
 to
 be
 one
 of
 the
 greatest
 physicists
 of
 all
 time.
 Einstein
 is
 known
 for
 developing
```

```
the
theory
of
relativity,
but
he
also
made
important
contributions
to
the
development
of
the
theory
of
quantum
mechanics.
```

```
[manju@localhost ~]$ echo Albert
Albert

[manju@localhost ~]$ !e
echo Albert
Albert

[manju@localhost ~]$ !echo
echo Albert
Albert

[manju@localhost ~]$ !echo Einstein
echo Albert Einstein
Albert Einstein
```

```
cat phy.txt

Albert Einstein was a German-born theoretical physicist, widely acknowledged to
be one of the greatest physicists of all time. Einstein is known for developing
the theory of relativity, but he also made important contributions to the
development of the theory of quantum mechanics.


fold -w 20  phy.txt

Albert Einstein was
```

```
 a German-born theor
etical physicist, wi
dely acknowledged to
be one of the great
est physicists of al
l time. Einstein is
known for developing
the theory of relat
ivity, but he also m
ade important contri
butions to the devel
opment of the theory
of quantum mechanic
s.
```

```
find . -type f -size +10k
# Display files having size > 10 kilobytes

find . -type f -size -10k
# Display files having size < 10 kilobytes

find . -type f -size 10k
# Display files having size = 10 kilobytes
```

Command:

```
traceroute google.com
```

Description:

Prints the route that a packet takes to reach the Google (172.217.26.206**) host from the local machine**

Command:

```
cat 1.txt
```

```
Einstein
Newton
Albert

gzip 1.txt

zcat 1.txt.gz

Einstein
Newton
Albert
```

```
[manju@localhost ~]$ echo {2,4}
2 4
[manju@localhost ~]$ echo {2,4}{2,4}
22 24 42 44
[manju@localhost ~]$ echo {2,4}{2,4}{2,4}
222 224 242 244 422 424 442 444
```

Description:

View the contents of zipped file

Command:

```
zdiff 1.txt.gz 2.txt.gz
```

Description:

Compare the contents of two zipped files (1.txt.gz, 2.txt.gz**)**

```
 du -cks *
# Display the space usage of files in the current directory
```

Command:

```
ss | less
```

Description:

List all connections

Command:

```
ss -aA tcp
```

```
echo "Albert" > 1.txt ; cp 1.txt 2.txt ; cp 1.txt 3.txt;
# 2.txt and 3.txt are copy of 1.txt

[manju@localhost ~]$ x="Albert Einstein"; echo ${x/Albert/Elsa}
Elsa Einstein
```

Description:

Filter out TCP (Transmission Control Protocol**) connections**

Command:

```
ss -aA udp
```

```
[manju@localhost ~]$ x=abcdefghijklmnopqrstuvwxyz; echo ${x:4}
efghijklmnopqrstuvwxyz

[manju@localhost ~]$ x=abcdefghijklmnopqrstuvwxyz; echo ${x:4:8}
efghijkl

[manju@localhost ~]$ x=abcdefghijklmnopqrstuvwxyz; echo ${x:(-1)}
z
[manju@localhost ~]$ x=abcdefghijklmnopqrstuvwxyz; echo ${x:(-2):2}
yz
```

Description:

Filter out UDP (User Datagram Protocol**) connections**

Command:

```
ss -lnt
```

Description:

Display only listening sockets

Command:

```
ss -ltp
```

```
[manju@localhost ~]$ du -ak /home/manju | sort -nrk 1 | head -n 4
8664    /home/manju
8088    /home/manju/.cache
7100    /home/manju/.cache/tracker
5120    /home/manju/.cache/tracker/meta.db-wal

# Find the largest size files from a given directory
```

Description:

Print process name and PID

Command:

```
ss -s
```

```
netstat -tnp
```

List opened port and services

Description:

Print summary statistics

Command:

```
ss -tl6
```

Description:

Display only IPv6 connections

Command:

```
ss -tl -f inet
```

```
ls -l | awk '{print $1 " : " $8}'
```

Print the permission and filename of each file in the current directory

Description:

Display only IPv4 socket connections

Command:

```
ss -t4 state established
```

Description:

Display all IPv4 TCP sockets that are in connected state

Command:

```
pmap 3244
```

Description:

View the memory map of a process with Process ID (3244)

Command:

```
apropos -r 'remove file'
```

Description:

Find command that removes file

Command:

```
apropos editor
```

Description:

Display information about the editing programs that are available on a system

```
[manju@localhost ~]$ echo "<a href=\"$img\" ><img src=\"thumbs/$img\" title=\"$img\" /></a>" >> 1.html
[manju@localhost ~]$ cat 1.html
<a href="" ><img src="thumbs/" title="" /></a>
```

Command:

```
apropos pstree
```

Description:

Provide information about the pstree command (which displays the names of the processes currently on the system in the form of a tree diagram**)**

```
chmod 744 file.txt
# Changes the permission of the file.txt and assigns the permission: rwxr--r--

head -10 myfile1.txt
# Display the first 10 lines of the file myfile1.txt

tail -10 myfile1.txt
# Display the last 10 lines of the file myfile1.txt

ls; who
# Execute ls command first and then who command

[manju@localhost ~]$ du -s /home/manju
33840 /home/manju
```

The **apropos command** is useful when you know what you want to do, but you have no idea what command you should be using to do it. If you were wondering **how to locate files**, for example, the commands

- apropos find
- apropos locate

would have a lot of suggestions to offer.

Display the total disk space used by the user "manju"

```
basename /etc/passwd

Output: passwd

basename /usr/local/apache2/conf/httpd.conf
```

- = **:**Used for assigning value to the variable
- = = **:**Used for string comparison

```
Output: httpd.conf
```

```
echo a b c d e f| xargs

Output: a b c d e f


echo a b c d e f| xargs -n 3

Output:

a b c
d e f
```

display only 3 items per line

Command:

```
env
```

Description:

Print out a list of all environment variables

Command:

```
printenv HOME
```

Description:

Print HOME variable value

```
cat score.txt

Albert-30
John-50
William-80
Stephen-20
Justin-40

cut -d- -f2 score.txt

30
50
80
20
40

cut -d- -f1 score.txt

Albert
John
William
Stephen
Justin
```

```
[manju@localhost ~]$ printf "2 + 2 = %d" 4
2 + 2 = 4

[manju@localhost ~]$ printf "William %f" 2
William 2.000000
```

```
[manju@localhost ~]$ printf "Hi %s, I am %s" Alan Albert
Hi Alan, I am Albert
```

```
cat 1.txt

Hello World


cat 2.txt

Computer Program


paste 1.txt  2.txt

Hello World    Computer Program
```

```
[manju@localhost ~]$ a="Einstein"; echo "Albert $a";
Albert Einstein
[manju@localhost ~]$ a="Einstein"; echo 'Albert $a'
Albert $a
```

```
cat 1.txt

Hello World


cat 2.txt

Computer Program


join 1.txt  2.txt

Hello World    Computer Program
```

```
[manju@localhost ~]$ echo $(cal) $(date) >> 12.txt
[manju@localhost ~]$ cat 12.txt
May 2022 Su Mo Tu We Th Fr Sa 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
18 19 20 21 22 23 24 25 26 27 28 29 30 31 Fri May 27 00:14:13 PDT 2022
```

Command:

```
rev 1.txt
```

Description:

Reverse lines of a file (1.txt)

```
cat 3.txt

22
33
11
77
55

sort 3.txt

11
22
33
55
77
```

sorts numeric values in 3.txt file and displays sorted output

```
[manju@localhost ~]$ path="/user/documents/1.pdf"; echo ${path%.pdf}
/user/documents/1
[manju@localhost ~]$ path="/user/documents/1.pdf"; echo ${path%.pdf}.txt
/user/documents/1.txt
[manju@localhost ~]$ path="/user/documents/1.pdf"; echo ${path%/*}
/user/documents
```

```
[manju@localhost ~]$ path="/user/documents/1.pdf";echo ${path##*.}
pdf
[manju@localhost ~]$ path="/user/documents/1.pdf"; echo ${path##*/}
1.pdf
[manju@localhost ~]$ path="/user/documents/1.pdf"; echo ${path#*/}
user/documents/1.pdf
```

```
[manju@localhost ~]$ if ping -c 1 google.com; then echo " It looks that your internet connection is operational"; fi

PING google.com (172.217.163.174) 56(84) bytes of data.
64 bytes from maa05s05-in-f14.1e100.net (172.217.163.174): icmp_seq=1 ttl=128 time=11.8 ms

--- google.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 11.815/11.815/11.815/0.000 ms
It looks that your internet connection is operational
```

```
cat 1.txt

Hello World


cat 1.txt | tr "[a-z]" "[A-Z]"

HELLO WORLD
```

convert from lower case to upper case

```
[manju@localhost ~]$ name="Albert Einstein"; echo ${name:7:9}
Einstein
```

```
cat 5.txt

zz
zz
yy
yy
yy
xx

uniq 5.txt

zz
yy
xx
```

removes duplicate lines and displays unique lines

```
cat 6.txt

Einstein
Newton
Tesla
```

```
awk -F ':' '{print $1}' /etc/shadow|uniq -u

# Print the login names of all users on a system
```

```
nl 6.txt

1 Einstein
2 Newton
3 Tesla
```

numbers the lines in a file (6.txt)

Command:

```
ls -l *.txt
```

Description:

Lists the files with .txt **extension**

> The thing with Linux is that the developers themselves are actually customers too: that has always been an important part of Linux.
>
> **Linus Torvalds**

Linux	Unix
Free to use (**open source**)	Licensed Operating System (**closed source**)
Linux is just the kernel	Unix is a complete package of Operating System
Bash (**Bourne Again SHell**) is default shell for Linux	Bourne Shell is default shell for Unix
Portable and is booted from a USB Stick	Unportable
Source code is accessible to the general public	Source code is not accessible to anyone
Uses Graphical User Interface with an optional Command Line Interface	Uses Command Line Interface

Command:

```
echo $SHELL
```

Description:

Print the Default shell of user

Command:

```
echo $0
```

Description:

Display the name of the currently running process **($0 is the name of the running process**). If you use it inside of a shell then it will return the name of the shell. If you use it inside of a script, it will return the name of the script

Command:

```
echo *
```

Description:

Print all files and folders – similar to ls command

```
sed -n '10p' file.txt

# Print the 10th line in the file.txt
```

```
[manju@localhost ~]$ grep -c "Hello" file.txt
1
```

Count the number of lines containing the word **"Hello"** in the file.txt

Command:

```
ps -p $$

Output:

   PID  TTY          TIME   CMD
  3352 pts/0    00:00:00 bash
```

```
sed '50i\HELLO' file.txt
# Inserts a line "HELLO" at every 50th line of a file.txt
```

Description:

Print the process ID of the current shell ($$ is the process ID of the current shell)

Command:

```
cat /etc/shells
```

Description:

List shells

Command:

```
echo m*
```

Description:

Display the files in the current folder that start with the letter "m".

Command:

```
last
```

Description:

List last logins of users and what happened such as "shutdown**" or "**crash**" etc.**

Command:

Command:

```
echo ~
```

Description:

Print your home folder path

```
bzip2 -k phy.txt
```

Description:

Compresses but does not deletes the original file

phy.txt → phy.txt.bz2

Command:

```
bzip2 -d phy.txt.bz2
```

Description:

Decompresses the compressed file (phy.txt.bz2**)**

phy.txt.bz2 → phy.txt

Command:

```
bzcat phy.txt.bz2
```

Description:

Display the contents of compressed file (phy.txt.bz2**)**

Command:

```
bunzip2 phy.txt.bz2
```

Description:

Decompresses the compressed file (phy.txt.bz2**)**

Command:

```
crontab -l
```

Description:

Display current logged-in user's crontab entries

```
cat /dev/null > phy.txt
```

```
cp /dev/null phy.txt
```

```
echo "" > phy.txt
```

```
echo > phy.txt
```

Description:

Empty the content of a file (phy.txt**)**

Command:

> **Independent software vendor:** An enterprise organization that develops and distributes software for one or more computer hardware platforms

```
nohup ping google.com &
```

Description:

Ping google.com **and send the process to the background**

Command:

- **Recode:** Modifications to implementation features
- **Redesign:** Modifications to design features
- **Respecify:** Modifications to requirements features

```
nohup ping google.com > log.txt &
```

Description:

Save the ping logs to log.txt

```
pgrep -a ping

Output:

3858 ping google.com
4200 ping google.com
4236 ping google.com

kill 3858

pgrep -a ping

Output:

4200 ping google.com
4236 ping google.com
```

Command:

```
ls -la /home
```

Description:

Display the contents of /home

Command:

```
sudo shutdown 2
```

Description:

Power-off the machine after 2 minutes

Command:

```
shutdown -c
```

Description:

Cancel the shutdown process

Command:

```
pr 36.txt
```

Description:

Display the contents of the file (36.txt**) one page after the other**

Command:

```
stty -a
```

Description:

Display all current terminal settings

Data in Use

↓

Data that is currently being processed by a computer

Command:

```
ls -1
```

Description:

List files one per line

Heterogeneous Network: A network consisting of systems with different architectures

Command:

Homogeneous Network: A network consisting of systems with the same architecture

```
yes John
```

Description:

Outputs a string (John**) repeatedly until killed**

Command:

```
vdir
```

Description:

List files and directories in the current directory (one per line**) with details**

Command:

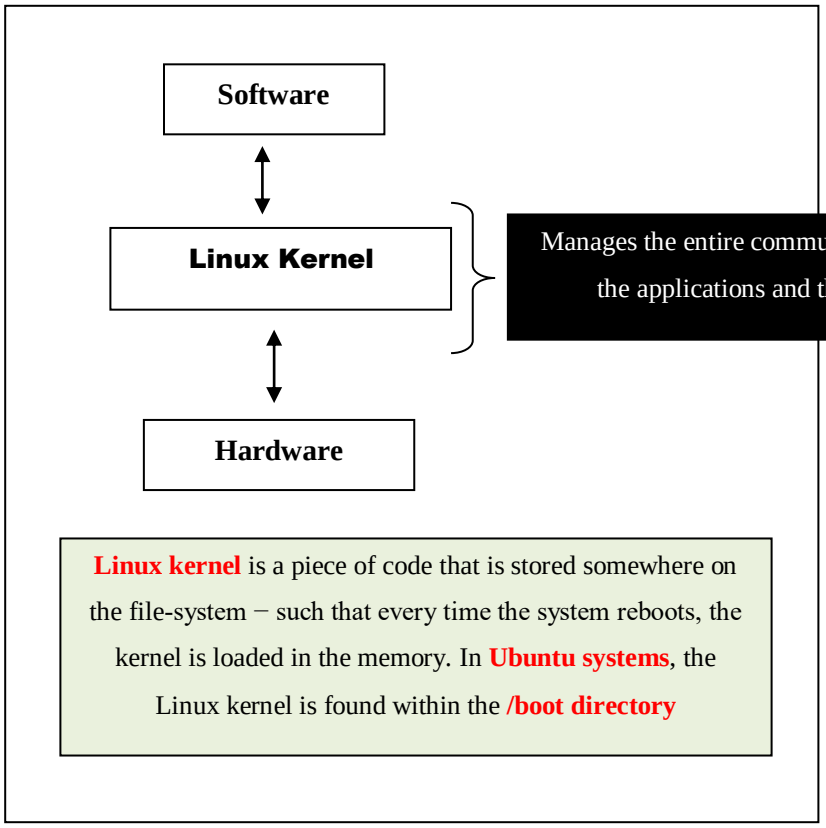
```
who -b
```

Description:

Print when the system was booted

```
# Open phy.txt with nano
```

```
nano phy.txt


# Open phy.txt with vim

vim phy.txt
```



Manages the entire communication between the applications and the hardware

Linux kernel is a piece of code that is stored somewhere on the file-system − such that every time the system reboots, the kernel is loaded in the memory. In **Ubuntu systems**, the Linux kernel is found within the **/boot directory**

Memory

The internal storage location where data and information is stored on a computer

Network

A group of computers connected to each other in order to send and receive data

Protocol

A set of rules that are followed by two devices while interacting with each other

- **Transmission Control Protocol** (TCP)
- **Internet Protocol** (IP)

Communication protocols used to connect host computers on the Internet

Port (USB)

A **hardware connecting component** that enables two computers to allow data sharing physically

```
w --ip-addr
```

```
# Displays information regarding the users currently on the machine, login time, IDLE time,
TTY and CPU time

Output:

11:12:10 up  1:29,  2 users,  load average: 0.02, 0.04, 0.10
USER     TTY      FROM  LOGIN@   IDLE   JCPU   PCPU WHAT
manju    :0       :0    02:43   ?xdm?   3:30   0.65s gdm-session-worker [pa
manju    pts/0    :0    11:01    2.00s  0.10s  0.01s w --ip-addr

w –short
# Omits CPU time and login information
Output:

11:11:46 up  1:28,  2 users,  load average: 0.02, 0.04, 0.11
USER     TTY      FROM              IDLE WHAT
manju    :0       :0               ?xdm?  gdm-session-worker [pam/gdm-password]
manju    pts/0    :0                2.00s w --short
```

Command:

```
findmnt
```

Description:

Display a list of currently mounted file systems

Command:

```
ip addr show
```

Description:

List IP addresses and network interfaces

Command:

```
netstat -pnltu
```

Description:

List active (listening) ports

Durability:

Regardless of whether the database fails or not, each database transaction will be preserved by backups or other methods

Command:

```
journalctl
```

Description:

Display systemd, kernel and journal logs

Command:

Isolation: If several transactions are submitted at the same time, the database will prevent them from interfering

```
sudo systemctl status network
```

Description:

Display the status of network service

Command:

```
sudo systemctl start network
```

Description:

Start the network service

Command:

```
sudo systemctl stop network
```

Description:

Stop the network service

Command:

Go program:

```
package main
import "fmt"

func main() {
    lang:= []string{"Java", "Go", "Python"}
    for x, y:= range lang {
       fmt.Println(x, y)
    }
}
```

Output:

```
0 Java
1 Go
2 Python
```

```
sestatus -b
```

Description:

Display the current state of Booleans

Command:

```
getenforce
```

Description:

Reports whether SELinux **is enforcing, permissive or disabled**

Security-Enhanced Linux (SELinux) is a security architecture for Linux systems that allows administrators to have more control over who can access the system

```
setenforce 0
```

```
getenforce
```

```
Output:

Permissive
```

```
setenforce 1
```

```
getenforce
```

```
Output:

Enforcing
```

- **Enforcing** - SELinux security policy is enforced.
- **Permissive** - SELinux prints warnings instead of enforcing.
- **Disabled** - No SELinux policy is loaded.

Intent-based networking (IBN): An emerging technological approach that improves network operations and uptime by combining intelligence, analytics and automation

Command:

```
sestatus
```

Description:

Display the current status of the SELinux that is running on your system

Command:

```
ps -aef
```

Industrial internet of things (IIoT):

In the industrial sector, smart sensors, instruments and autonomous robots are being used to improve production and industrial processes

Description:

Display full listing of processes on your system

Command:

```
sar
```

Data Deduplication:

Duplicate blocks within a dataset are identified and removed, lowering the amount of traffic that would flow over wide-area network connections

Description:

Display System Activity Report

Command:

Fail-safe

In the case of a failure, a system that automatically switches to a safer operational mode

```
ulimit
```

Description:

Report the resource limit of the current user

```
Output:

Unlimited        The current user can consume all the resources the current system supports
```

2 types of resource limitation:

- **Hard resource limit:** The physical limit that the user can reach.
- **Soft resource limit:** The limit that is manageable by the user (**its value can go up to the hard limit**)

Command:

Consistency checker:

A software tool for checking the accuracy and reliability of demands in design specifications

```
ulimit -a
```

Description:

Report all the resource limits for the current user

Command:

Triggers: The Structured Query Language commands that are automatically executed whenever there is a modification in the database

```
ulimit -s
```

Description:

Check the maximum stack size of the current user

Command:

Audit log: A record that contains information about the users who accessed the system and the operations that users performed during a specific time period

```
ulimit -e
```

Description:

Check out the max scheduling priority of the current user

Autonomous Database: A cloud-based solution that employs machine learning to automate many of the everyday database management activities

Crash recovery

↓

Database recovery from a failure

Command:

```
ulimit -u
```

Description:

Display the maximum number of user processes

Command:

```
ulimit -v
```

Description:

Check out the size of virtual memory

Change Data Capture (CDC):

A method of keeping track of data changes in a database

Command:

```
ulimit -n
```

Description:

Check out how many file descriptors a process can have

Command:

```
man limits.conf
```

```
[manju@localhost ~]$ echo "alanBalbertBjohnBmary" | xargs -d B
alan albert john mary

[manju@localhost ~]$ echo "alanBalbertBjohnBmary" | xargs -d B -n 2
alan albert
john mary

[manju@localhost ~]$ echo "ALAN TURING" | tr 'A-Z' 'a-z'
alan turing
```

Description:

Display the in-depth information on the limits.conf **configuration file**

Command:

```
sar -V
```

Description:

Display the sar version

Command:

```
sar -u 2 5
```

```
[manju@localhost ~]$ echo 987654 | tr '0-9' '9876543210'
012345 # Encrypted
[manju@localhost ~]$ echo 012345 | tr '9876543210' '0-9'
987654 # Decrypted
```

Description:

Report CPU details total 5 times with the interval of 2 seconds

Command:

```
sar -n DEV 1 3 | egrep -v lo
```

Description:

Report about network interface, network speed, IPV4, TCPV4, ICMPV4 network traffic and errors

Command:

```
sar -v 1 3
```

```
[manju@localhost ~]$ echo "Albert 1905 Papers" | tr -d '0-9'
Albert  Papers

[manju@localhost ~]$ echo "Albert 1905 Papers" | tr -d -c '0-9 \n'
 1905
```

Description:

Report details about the process, kernel thread, i-node, and the file tables

Command:

```
sar -S 1 3
```

```
[manju@localhost ~]$ date "+%d %B %Y"
29 July 2022
[manju@localhost ~]$ date --date "Mar 03 1988" +%A
Thursday
```

Description:

Report statistics about swapping

Command:

```
sar -b 1 3
```

Description:

Report details about I/O operations like transaction per second, read per second, write per second

Command:

```
sudo systemctl status firewalld
```

Description:

Display the status of the firewalld

Fragmentation

The methodology of breaking down an **IP datagram** into smaller bits to match the needs of a specific physical network

Command:

```
sudo systemctl start firewalld
```

Description:

Start the firewalld **service**

```
[manju@localhost ~]$ n=alan.turing.ai.pdf; echo ${n#*.}
turing.ai.pdf
```

```
[manju@localhost ~]$ n=alan.turing.ai.pdf; echo ${n##*.}
pdf
```

firewalld is a firewall management tool for Linux operating systems

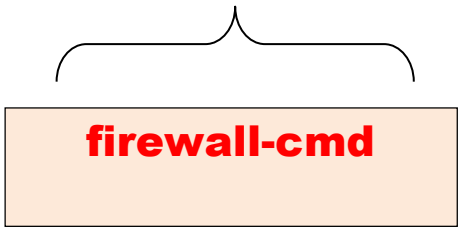
```
ls ??[rgi]*

# List all files in the 3rd place an r, g or i
```

Command:

```
firewall-config
```

Description:

Start the graphical firewall configuration tool



Command:

```
firewall-cmd --list-all-zones
```

Description:

List all zones

Interpreter	Compiler
Translates the program line by line into machine code	Scans the whole program and converts it into machine code

Command:

```
firewall-cmd --get-default-zone
```

Description:

Check the currently set default zone

Command:

```
firewall-cmd --list-services
```

Description:

Display currently allowed service on your system

Command:

```
firewall-cmd --list-ports
```

Description:

List the ports that are open on your system

Command:

```
firewall-cmd --zone=work --list-services
```

Description:

List services that are allowed for the public zone

Command:

```
mtr --report google.com
```

Description:

Provides information about the route that Internet traffic takes between the local system and a remote host (google.com**)**

Command:

```
sudo yum install samba
```

Description:

install Samba (CentOS**)**

Samba is client/server technology that implements network resource sharing across operating systems. With Samba, files and printers can be shared across Windows, Mac and Linux/UNIX clients.

Iterative incremental software development: The methodology of constructing a software application in tiny steps by iterating over the various software phases a number of times

Software entropy: The extent to which a software system is disorganized

Command:

```
firewall-cmd --permanent --zone=public --add-service=samba
```

Description:

Add Samba service to firewalld

Command:

```
zip q.zip q.txt
```

Description:

Create a zip file (q.zip**)**

Design of experiments:

An approach for designing experiments in order to obtain data suitable for Statistical analysis

Command:

```
unzip q.zip
```

Description:

Unzip a zip file (q.zip**)**

```
zipcloak q.zip

-------------------------------------------------------------------------------

# zipcloak prompts you for a password, and then ask you to confirm it:

   Enter password:
   Verify password:

...if the passwords match, it encrypts q.zip file

-------------------------------------------------------------------------------


unzip q.zip


#  When you try to unzip the q.zip file, it prompts you for the password before
allowing you to extract the file (q.txt) it contains
```

Command:

```
zgrep -l "Einstein" *
```

Description:

Display the names of the files with the word (Einstein**) present in it**

Command:

```
zipsplit -n 1048576 q.zip
```

Description:

Split q.zip **file to create a sequence of zipfiles** (q1.zip, q2.zip…..) − each no larger than **1048576 bytes** (one megabyte)

```
You could concatenate (q1.zip, q2.zip…..) into a new file, w.zip, with the
command:

                                cat q*.zip > w.zip
```

Command:

```
mtr google.com
```

```
[manju@localhost ~]$ echo alan turing | sed 's/\w\+/[&]/g'
[alan] [turing]
[manju@localhost ~]$ echo Albert Einstein 1905 Papers | sed 's/Einstein \([0-9]\)/\1/'
Albert 1905 Papers
```

Description:

Test the route and connection quality of traffic to the destination host **google.com**

Command:

```
[manju@localhost ~]$ echo alanalanalanalan | sed 's/alan/ALAN/2g'
alanALANALANALAN
[manju@localhost ~]$ echo alanalanalanalan | sed 's/alan/ALAN/3g'
alanalanALANALAN
[manju@localhost ~]$ echo alanalanalanalan | sed 's/alan/ALAN/4g'
alanalanalanALAN
```

```
route
```

Description:

Display IP routing table of a Linux system

Command:

```
[manju@localhost ~]$ echo einstein ALBERT | sed 's/\([a-z]\+\) \([A-Z]\+\)/\2 \1/'
ALBERT einstein
```

```
nmcli dev status
```

```
[manju@localhost ~]$ echo user: $(whoami);
user: manju
[manju@localhost ~]$ echo OS: $(uname);
OS: Linux
```

Description:

View all your network devices

Command:

Ergonomics: The science of fine-tuning product design to make it more user-friendly

```
nmcli con show
```

Description:

Check network connections on your system

Command:

Elegant solution: A solution that achieves the highest desired result with the least amount of effort

```
ss -ta
```

Description:

List all TCP ports (**sockets**) that are open on a server

Command:

Amelioration pattern:

A design pattern that explains how to get from a poor to a good solution

```
ss -to
```

Description:

Display all active TCP connections together with their timers

Command:

```
type -a alias
```

Description:

Check Bash Aliases in Linux

```
# Difference between %Y and %y is %Y will print 4 digits while %y will print the last 2 digits of the year.

echo "We are in the year = $(date +%Y)"
echo "We are in the year = $(date +%y)"
```

```
# Difference between %B and %b is, %B will print full month name while %b will print abbreviated month name.

echo "We are in the month = $(date +%b)"
```

```
echo "We are in the month = $(date +%B)"

# Difference between %A and %a is, %A will print full Weekday name while %a will print abbreviated weekday name.

echo "Current Day of the week = $(date +%A)"
echo "Current Day of the week = $(date +%a)"
```

```
echo "Date using %D = $(date +%D)"
echo "Date using %F = $(date +%F)"
```

```
Date using %D = 08/15/21
Date using %F = 2021-08-15
```

```
echo "current time in 24 hour format = $(date +%T)"
```

```
current time in 24 hour format = 01:27:46
```

```
echo "current time in 12 hour format = $(date +%r)"
```

```
current time in 12 hour format = 01:27:47 AM
```

```
# Print yesterday's date and time.
echo "Yesterday = $(date -d "Yesterday")"


# Print Tomorrow date and time.
```

```
echo "tomorrow = $(date -d "tomorrow")"

# Find what is the date and time before 10 days from now.
echo "Before 10 days = $(date -d "tomorrow -10 days")"

# Find last month and next month
echo "Last month = $(date -d "last month" "%B")"
echo "Next month = $(date -d "next month" "%B")"

# Find last year and next year
echo "Last Year = $(date -d "last year" "+%Y")"
echo "Next Year = $(date -d "next year" "+%Y")"
```

Command:

```
ls -lai /
```

Description:

Get the number of inodes of files in a directory (**root directory**)

Command:

```
sudo du --inode /
```

Description:

Get the total number of inodes in the root directory

Command:

```
ss -o state established '( sport = :http or sport = :https )'
```

Description:

Get the list of all clients connected to HTTP (Port 80) or HTTPS (Port 443)

Command:

```
ss -tn src :80 or src :443
```

Description:

List the numerical port numbers

Command:

```
sudo yum install putty
```

Description:

Install **PuTTy** on CentOS

Command:

```
sudo watch netstat -tulpn
```

Check the Linux version

- `cat /etc/*release`
- `cat /etc/os-release`

Description:

Watch TCP and UDP Open Ports in Real-Time

Command:

```
sudo watch ss -tulpn
```

Description:

Watch TCP and UDP Open Ports in Real-Time

Command:

- **Front End** → The representation part, what user sees
- **Back End** → Controlling of all the requests, made by users

```
timeout 5s ping google.com
```

Description:

Timeout a ping command after 5 seconds

Capability Maturity Model:

An approach for improving and refining a company's software development process

Command:

```
yum install curl
```

Description:

Install curl on CentOS

Synchronize-and-stabilize

A strategy to design and handle large-scale software systems on a broad scale

Command:

```
ss -ua
```

Description:

List all UDP Connections

Shotgun debugging: The methodology of detecting and eliminating errors from a computer software, hardware, or software problem by attempting multiple different solutions at the same time in the hopes of finding one that works

Command:

```
ss -lu
```

Description:

List all Listening UDP Connections

Command:

```
ss -p
```

```
[manju@localhost ~]$ ls ab*
abc.cpp  abc.txt  ab.pdf  ab.txt

[manju@localhost ~]$ ls ab?(.)pdf
ab.pdf

[manju@localhost ~]$ ls ab?(.pdf|.cpp)
ab.pdf
```

Description:

Display the Process IDs related to socket connections

Command:

```
ss -4
```

Description:

Display IPv4 and IPv6 Socket Connections

Command:

```
wc -l /home/fi*
# List the files in '/home' directory that start with 'fi' and then display the number of lines in each file
```

```
ss -6
```

Description:

Display IPv6 connections

> **"The only way to learn a new programming language is by writing programs in it."**
>
> **– Dennis Ritchie**

Command:

```
ss -at '( dport = :22 or sport = :22 )'
```

Description:

Filter Connections by Port Number

```
[manju@localhost ~]$ ls -l | grep '^-'| awk 'BEGIN {sum=0} {sum = sum + $5} END {print sum}'
26591723
```

Print the sum of bytes **(size of the file)** of all files in a directory

```
grep -w Hello file.txt
# Display the lines containing the word "Hello" in file.txt

sed 'p' < file.txt
# Duplicate each line in file.txt
```

Command:

```
sdiff phy.txt score.txt
```

Description:

Show Difference between Two Files (**phy.txt and score.txt**)

Command:

```
history -c
```

Description:

Delete or clear all the entries from bash history

Command:

```
ping -c 5 www.google.com
```

Description:

The ping test will stop after sending 5 packets

```
# count number of lines in each .txt file

ls *.txt | xargs wc -l

# count number of words in each .txt file

ls *.txt | xargs wc -w

# count number of characters in each .txt file

ls *.txt | xargs wc -c

# count lines, words and characters in each .txt file

ls *.txt | xargs wc
```

Command:

```
lslogins -u
```

Description:

Displays user accounts

Command:

```
systemctl list-units --type=service
```

Description:

List all loaded services on your system (whether active; running, exited or failed)

Command:

```
systemctl --type=service
```

Description:

List all loaded services on your system (whether active; running, exited or failed)

Command:

```
systemctl list-units --type=service --state=active
```

Description:

List all loaded but active services

Command:

```
systemctl --type=service --state=active
```

Description:

List all loaded but active services

Command:

```
systemctl list-units --type=service --state=running
```

Description:

List all running services (i.e., all loaded and actively running services)

Command:

```
[manju@localhost ~]$ ls ab@(.txt|.cpp)
ab.cpp  ab.txt
[manju@localhost ~]$ ls ab+(.txt|.pdf)
ab.pdf  ab.txt
[manju@localhost ~]$ ls ab*(.txt|.cpp)
ab.cpp  ab.txt
[manju@localhost ~]$ ls ab!(.pdf|.txt)
abc.cpp  ab.cpp  abc.txt
```

```
systemctl --type=service --state=running
```

Description:

List all running services (i.e., all loaded and actively running services)

```
#scan a single port

nc -v -w 2 z 192.168.56.1 22

# scan multiple ports

nc -v -w 2 z 192.168.56.1 22 80

# scan range of ports

nc -v -w 2 z 192.168.56.1 20-25
```

```
[manju@localhost ~]$ tel="002-006-369"; echo ${tel#*-}
006-369

[manju@localhost ~]$ tel="002-006-369"; echo ${tel##*-}
369

[manju@localhost ~]$ tel="002-006-369"; echo ${tel#*-*-}
369

[manju@localhost ~]$ tel="002-006-369"; echo ${tel%-*}
002-006

[manju@localhost ~]$ tel="002-006-369"; echo ${tel%%-*}
002
```

Command:

```
cat /etc/resolv.conf
```

Description:

Find out your DNS Server IP address

Command:

```
[manju@localhost ~]$ a="Albert"; echo ${#a}
6
```

```
less /etc/resolv.conf
```

Description:

Find out your DNS Server IP address

Command:

```
[manju@localhost ~]$ a=Albert; echo ${a?Alan}
Albert
```

```
findmnt -l
```

Description:

Display a list of currently mounted file systems in a tree-like format

Command:

```
[manju@localhost ~]$ a=Albert; b=1905; echo $(($a + $b))
1905
[manju@localhost ~]$ a="Albert"; b=1905; echo $(($a + $b))
1905
[manju@localhost ~]$ a="Albert"; b=1905; echo $a + $b
Albert + 1905
```

```
uptime -p
```

Description:

Check Linux Server Uptime

Command:

```
uptime –s
```

Description:

Check Linux Server Starting Time

Command:

```
uptime –h
```

Description:

Display uptime's version information

Command:

```
grep -o -i Justin score.txt | wc –l
```

Description:

Count the number of times "**Justin**" appears in the file (**score.txt**)

Command:

```
crontab -r
```

Description:

Delete all crontab jobs

```
ADD=$(( 1 + 2 ))
echo $ADD
                                    3

MUL=$(( $ADD * 5 ))
echo $MUL
                                   15

SUB=$(( $MUL - 5 ))
echo $SUB
                                   10

DIV=$(( $SUB / 2 ))
echo $DIV
                                    5
MOD=$(( $DIV % 2 ))
echo $MOD

                                    1
```

Command:

```
expr length "This is myw3schools.com"
```

Description:

Find the length of a string (**This is myw3schools.com**)

```
echo '3+5' | bc
                                        8

awk 'BEGIN { a = 6; b = 2; print "(a + b) = ", (a + b) }'

                                    (a + b) =  8
```

Command:

```
factor 10
```

Description:

Decompose an integer **(10)** into prime factors

Command:

```
ps -e
```

Description:

Display every active process on a Linux system

Command:

```
ps -x
```

```
[manju@localhost ~]$ echo alan | sed 's/a/A/' | sed 's/n/N/'
AlaN
[manju@localhost ~]$ echo alan | sed 's/a/A/;s/n/N/'
AlaN
[manju@localhost ~]$ echo alan | sed -e 's/a/A/' -e 's/n/N/'
AlaN
```

Description:

Display User Running Processes

Command:

```
ps -fU manju
```

```
[manju@localhost ~]$ echo -e "a\nl\nb\ne\nr\nt" | grep a -A 1
a
l

ifconfig | cut -c-10 | tr -d ' ' | tr -s '\n'
```

Print the list of network interfaces available on a system

Description:

Display a user's processes by user name (**manju**)

Command:

```
ps -fu 1000
```

Description:

Display a user's processes by real user ID (**RUID**)

Command:

```
ps -U root -u root
```

Description:

Display every process running with root user privileges (real and effective ID)

```
echo -e "The following users are logged on the system:\n\n $(who)"


              manju    :0           Aug 15 03:31 (:0)

              manju    pts/1        Aug 15 03:32 (:0)
```

Command:

```
sh <(curl https://nixos.org/nix/install) --daemon
```

Description:

Install Nix Package Manager in Linux

Software psychology: The scientific study of behavior and mental processes aimed at identifying and describing human limits while dealing with computing machines

Command:

```
locale
```

Description:

View System Locale in Linux

Command:

```
locale -a
```

System Modeling: A methodology for describing, displaying, evaluating and changing a system's architecture

Description:

Display a list of all available locales

```
cat score.txt

                                    Justin-40


cat score.txt | tr [:lower:] [:upper:]

                                    JUSTIN-40


cat score.txt | tr [a-z] [A-Z] >output.txt
cat output.txt

                                    JUSTIN-40
```

```
cat domainnames.txt

                                www. google. com
                                  www. fb. com
                                www. mactech. com

cat domainnames.txt | tr -d ''

                                  www.google.com
                                    www.fb.com
                                  www.mactech.com
```

Remove the spaces in the domain names

```
cat domainnames.txt

                                www.google.....com
                                    www.fb.com
                                  www.mactech.Com

cat domainnames.txt | tr -s ''
                                  www.google.com
                                    www.fb.com
                                  www.mactech.Com

echo "My UID is $UID"

                                   My UID is 0


echo "My UID is $UID" | tr " "  "\n"

                                        My
                                       UID
                                       is
                                        0

echo "myw3schools.com  =>Linux-Books,Src,Tutorials" | tr " " ":"


                    myw3schools.com:=>Linux-Books,Src,Tutorials
```

A space into a " : " character

Command:

```
!sud
```

Description:

Re-execute previously used command

Command:

```
!sudo
```

```
ps -eo comm,pcpu --sort -pcpu | head

# List the top 10 CPU consuming processes


[manju@localhost ~]$ cat /proc/meminfo | head -1
MemTotal:       999936 kB

[manju@localhost ~]$ cat /proc/cpuinfo | sed -n 5p
model name    : Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz
                 ↳ The processor name
```

Description:

Re-execute previously used command

Command:

```
cut -d: -f1 < /etc/passwd | sort | xargs
```

Description:

Generate a compact list of all Linux user accounts on the system

Command:

```
[manju@localhost ~]$ who -H
NAME     LINE         TIME             COMMENT
manju    :0           2022-06-08 09:34 (:0)
manju    pts/2        2022-06-08 09:48 (:0)

[manju@localhost ~]$ who
manju    :0           2022-06-08 09:34 (:0)
manju    pts/2        2022-06-08 09:48 (:0)
```

```
zcat phy.txt.gz myfiles.txt.gz
```

Description:

View multiple compressed files (**phy.txt.gz and myfiles.txt.gz**)

Command:

```
find . -type f -name "*.php"
```

Description:

Find all php files in a directory

```
[manju@localhost ~]$ x=0; until [ $x -gt 2 ]; do echo $x; x=`expr $x + 1`; done
0
1
2
```

Command:

```
find . -type f -perm 0777 -print
```

Description:

Find all the files whose permissions are 777

Command:

```
find / -type f ! -perm 777
```

Description:

Find all the files without permission 777

Command:

```
find / -perm /g=s
```

Description:

Find all SGID set files

Command:

```
find / -perm /u=r
```

Description:

Find all Read-Only files

Command:

Computer Supported Collaborative Work (CSCW):

Computing tools and advanced technology that assist a team of individuals involved in the project from multiple locations

```
find / -perm /a=x
```

Description:

Find all Executable files

```
[manju@localhost ~]$ ps auxw | grep nginx   (# Check if Nginx is running or not)
manju       7859  0.0  0.0 112648   960 pts/0    R+   05:32   0:00 grep --color=auto nginx
[manju@localhost ~]$ ps auxw | grep apache  (# Check if Apache is running or not)
manju       7864  0.0  0.0 112648   956 pts/0    S+   05:33   0:00 grep --color=auto apache
```

Command:

Upsert: When no existing entry is found, a database operation that either changes an existing entry or inserts a new entry is performed

```
find . -type f -name "phy.txt" -exec rm -f {} \;
```

Description:

Find and remove **phy.txt** File

Command:

Application Transformation: The process of bringing enterprise existing applications to modern technological stacks in order to reform outdated applications to satisfy customers' demands

```
find . -type f -name "*.txt" -exec rm -f {} \;
```

Description:

To find and remove multiple **.txt** files

Command:

```
ps -eo pcpu,pid,user,args | sort -r | head -10

# Check CPU utilization
```

```
find . -type f -name "*.mp3" -exec rm -f {} \;
```

Description:

Application Refactoring: The act of modifying software's source code without changing its functionality in order to enhance understandability, richness, manageability, and expandability − among other nonfunctional aspects of the code

To find and remove multiple **.mp3** files

Command:

```
[manju@localhost ~]$ x=alan; echo alan turing | sed "s/$x/ALAN/";
ALAN turing
```

```
find /tmp -type d -empty
```

Description:

Find all Empty Directories

Command:

```
find /tmp -type f -name ".*"
```

```
[manju@localhost ~]$ x="alan"; echo ${x}
Alan
[manju@localhost ~]$ x=5496; echo ${#x}
4 # length of the value of x
```

Description:

File all Hidden Files

```
[manju@localhost ~]$ echo -e "\e[1;31m Alan \e[0m"
 Alan
[manju@localhost ~]$ echo -e "\e[1;42m Alan \e[0m"
 Alan
```

Command:

```
find / -mtime 50
```

Description:

Find Last 50 Days Modified Files

Command:

```
find / -atime 50
```

```
find . ! -name "*.txt" -print
# List all the files whose names do not end in .txt

find . -type f -name "*.txt" -print0 | xargs -0 rm -f
# List of all the .txt files and remove them
```

Description:

Find Last 50 Days Accessed Files

Command:

```
find / -cmin -60
```

Description:

Find Changed Files in Last 1 Hour

Command:

```
find / -mmin -60
```

Description:

Find Modified Files in Last 1 Hour

Command:

```
find / -amin -60
```

Description:

Find Accessed Files in Last 1 Hour

Command:

Command:

```
type cat
```

Description:

Identifies whether the "cat" command is a shell built-in command, subroutine, alias, or keyword.

```
find / -size 50M
```

Description:

Find all 50MB files

Command:

```
find / -type f -size +100M -exec rm -f {} \;
```

```
[manju@localhost ~]$ echo "$BASH_VERSION"
4.2.46(1)-release
```

Description:

Find and Delete 100MB Files

Command:

```
find / -type f -name *.mp3 -size +10M -exec rm {} \;
```

Description:

Find all .mp3 files with more than 10MB and delete them

```
[manju@localhost ~]$ for x in `seq 5 -2 1`; do echo -en "${x} ..."; done; echo "Albert"
5 ...3 ...1 ...Albert
```

Command:

```
cat /proc/sys/fs/file-max
```

Description:

Find Linux Open File Limit

Command:

```
[manju@localhost ~]$ echo *
1.txt 1.zip ab.txt AB.txt bc.txt BC.txt cd.txt CD.txt Desktop Documents
Downloads io.txt IO.txt log Music Pictures Public Templates test.txt Videos
[manju@localhost ~]$ echo '*'
*
```

```
ulimit -Hn
```

Description:

Check Hard Limit in Linux

Command:

```
ulimit -Sn
```

Description:

Check Soft Limits in Linux

Command:

```
timedatectl  status
```

```
[manju@localhost ~]$ mkdir -p /home/users/{1,2,3}.txt/documents
[manju@localhost ~]$ ls -ld /home/users/*/documents
drwxr-xr-x. 2 root root 6 May 29 19:27 /home/users/1.txt/documents
drwxr-xr-x. 2 root root 6 May 29 19:27 /home/users/2.txt/documents
drwxr-xr-x. 2 root root 6 May 29 19:27 /home/users/3.txt/documents
```

Description:

Display the current time and date on your system

Command:

```
[manju@localhost ~]$ touch ab.txt bc.txt cd.txt io.txt
[manju@localhost ~]$ ls [coibd]*
bc.txt  cd.txt  io.txt
```

```
timedatectl list-timezones
```

Description:

View all available timezones

Command:

```
timedatectl list-timezones | egrep -o "Asia/B.*"
timedatectl list-timezones | egrep -o "Europe/L.*"
timedatectl list-timezones |  egrep  -o "America/N.*"
```

Description:

```
[manju@localhost ~]$ ps -eaf | grep -w apache
manju   4033    3913   0 01:15 pts/1    00:00:00 grep --color=auto -w apache
```

List all of the **Apache** web server processes

Find the local timezone according to your location

Command:

```
timedatectl set-timezone "Asia/Kolkata"
```

```
[manju@localhost ~]$ echo Alan
Alan
[manju@localhost ~]$ echo !echo
echo echo Alan
echo Alan
[manju@localhost ~]$ echo \!echo
!echo
```

Description:

Set your local timezone in Linux

Command:

```
[manju@localhost ~]$ echo Albert Einstein was a Jewish\; German-born theoretical physicist.
Albert Einstein was a Jewish; German-born theoretical physicist.
```

```
swapon --summary
```

Description:

```
[manju@localhost ~]$ echo Albert, Alan \& John
Albert, Alan & John
```

View a summary of swap space usage by device

Command:

```
cat /proc/swaps
```

```
[manju@localhost ~]$ ls -ld /etc/rc{0,1,2,3,4,5,6}.d
lrwxrwxrwx. 1 root root 10 May 18 20:54 /etc/rc0.d -> rc.d/rc0.d
lrwxrwxrwx. 1 root root 10 May 18 20:54 /etc/rc1.d -> rc.d/rc1.d
lrwxrwxrwx. 1 root root 10 May 18 20:54 /etc/rc2.d -> rc.d/rc2.d
lrwxrwxrwx. 1 root root 10 May 18 20:54 /etc/rc3.d -> rc.d/rc3.d
lrwxrwxrwx. 1 root root 10 May 18 20:54 /etc/rc4.d -> rc.d/rc4.d
lrwxrwxrwx. 1 root root 10 May 18 20:54 /etc/rc5.d -> rc.d/rc5.d
lrwxrwxrwx. 1 root root 10 May 18 20:54 /etc/rc6.d -> rc.d/rc6.d
```

Listing of /etc/rc*.d

Description:

Check swap usage information

```
[manju@localhost ~]$ echo "Albert Einstein, \"was a German-born theoretical physicist.\""
Albert Einstein, "was a German-born theoretical physicist."
```

```
# start recording of Linux terminal

script history_log.txt

                    Script started, file is history_log.txt

exit

                     Script done, file is history_log.txt
```

Command:

```
[manju@localhost ~]$ echo 'Albert "Einstein"'
Albert "Einstein"
[manju@localhost ~]$ echo "Albert 'Einstein'"
Albert 'Einstein'
```

```
dir -shl
```

Description:

> **"Don't write better error messages, write code that doesn't need them."**
>
> **– Jason C. McDonald**

List files and their allocated sizes in blocks

Command:

```
less /proc/sys/dev/cdrom/info
```

Description:

Display information about CD-ROM

```
while true; do date >> date.txt ; sleep 5 ; done &

cat date.txt

                        Mon Aug 16 03:05:36 PDT 2021
                        Mon Aug 16 03:05:41 PDT 2021
                        Mon Aug 16 03:05:46 PDT 2021
                        Mon Aug 16 03:05:51 PDT 2021
```

```
chmod u+x 256.txt
# Makes the file "256.txt" executable for the user (owner) only

chmod a+x 256.txt
# Makes the file "256.txt" executable for all (everyone)
```

```
a=$(pwd)
echo "Current working directory is : $a"
```

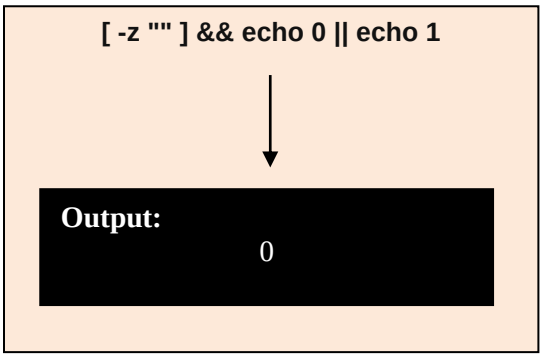
```
/home/manju
```

Command:

```
echo *.jpeg
```

Description:

Print all **.jpeg** files



Command:

```
echo 'linux' | fold -w1
```

Description:

Break down a word (**linux**) into individual

```
l
i
n
u
x
```

Command:

```
find . -user root
```

Description:

Output the files with respect of the user **(root)** owned files in the current directory

Command:

```
[manju@localhost ~]$ array=("Albert" "is" "a" "Physicist"); echo ${array[@]}
Albert is a Physicist
```

```
strace pwd
```

Description:

Trace a command **(pwd)** execution

Command:

```
top -u manju
```

Description:

Display specific User **(manju)** process details

```
echo ${variable:a:b}
```

- a → Start position
- b → Length

```
[manju@localhost ~]$ variable="My name is Albert and I am a Physicist.";
[manju@localhost ~]$ echo ${variable:11:6}
Albert
```

```
ls -d ?[ab]*
# List files where second letter is a or b
```

```
egrep "^ma|^jo" /etc/passwd|cut -d: -f1
# List the usernames which starts with ma or jo

[manju@localhost ~]$ array=("Albert" "is" "a" "Physicist"); echo ${array[0]}
Albert
[manju@localhost ~]$ array=("Albert" "is" "a" "Physicist"); echo ${array[1]}
is
```

Command:

```
uname -or
```

Workflow management

A **methodology** that defines workflows for quick re-design and re-implementation of processes as business organization demands change

Description:

Find Out Linux Kernel Version

Datasheet

A data presentation format consisting of columns and rows

Command:

```
uname -a
```

Description:

Print linux system information

Process Model: A paradigm for a collection of more or less ordered steps that must be completed in order to achieve a goal

Command:

```
cat /proc/version
```

Description:

Display some of your system information including the Linux kernel version

Command:

Database proxy: A database connector responsible for managing connections between database users and database servers

```
cat /etc/centos-release
```

Description:

Find Out Linux Distribution Name and Release Version

Command:

Functional Dependency

The value of Y is determined by the value of X

Y is functionally dependent on X

```
fuser .
```

Coding by Convention

(Convention over configuration):

A programming architecture approach that asserts that configuration should only be needed if we're doing anything new − thereby decreasing or removing the requirement for extra software configuration files

Description:

Displays the PIDs of processes currently accessing your current working directory

Command:

Dirty read:

Suppose transaction A modifies a row. Before transaction A commits the modification, transaction B reads the updated row

```
fuser -v -m .bashrc
```

Description:

Determine which processes are accessing your **~.bashrc file**

Command:

```
sudo fuser --list-signals
```

Description:

Displays all the possible signals that can be used with the fuser tool

Command:

```
sudo fuser -k -HUP /boot
```

Description:

Sends the HUP signal to all processes that have your **/boot directory** open

Command:

```
ls -al
```

Description:

List all the files with the file permissions, the number of links to that file, the owner of the file, the group of the file, the file size in bytes, the file's last modified datetime and the file name

Command:

```
echo "shutdown -h now" | at -m 23:55
```

Description:

Shutdown the system at 23:55 today

Command:

```
echo "updatedb" | at -m 23.55
```

Description:

Run **updatedb** at 23:55 today

Creates and updates the database of file names used by locate

 # Everyone can now read the file

chmod a+r myfiles.txt

 # Everyone can now read and write the file

chmod a+rw myfiles.txt

 # Others (not the owner, not in the same group of the file) cannot read, write or execute the file

chmod o-rwx myfiles.txt

Command:

```
echo $(ls -al)
```

Description:

Execute command "ls -al" and print the result to the standard output

Command:

```
top -b -o +%MEM | head -n 22
```

Description:

Display the top 15 processes sorted by memory use in descending order

Command:

```
top -b -o +%MEM | head -n 22 > report.txt
```

Description:

Redirect the output to a file (**report.txt**) for later inspection

Command:

```
ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head
```

Description:

Check Top Processes sorted by RAM or CPU Usage in Linux

Command:

```
find . -type f \( -name "*.sh" -o -name "*.txt" \)
```

Description:

Find all files in the current directory with **.sh** and **.txt file** extensions

Command:

```
find . -type f \( -name "*.sh" -o -name "*.txt" -o -name "*.c" \)
```

Description:

Find all files in the current directory with **.sh, .c** and **.txt file** extensions

Name of the shell	Program name	Symbol
Bourne Shell	sh	$
Korn Shell	ksh	$
C Shell	csh	%

- **Operating system:** [DOS, UNIX, LINUX, Windows, Novel NetWare]
- **Language processors:** [Interpreter, Compiler, Assembler, Editor]
- **System Utilities:** [Loader and Linker]

Editors for document files	Editors for non-document files
Notepad	Turbo C
WordPad	Turbo C++
MS-WORD	Borland C/C++
	vi editor
	vim editor
	pico
	Emacs

LAN	WAN
Small computer network that covers a building or a campus	Wider computer network that covers a city or the entire globe
High bandwidth	Low bandwidth
Lower delays as they cover smaller distances	Greater delays as they cover far distances
Security is very high	Security is high

Generally, programmers commit **three types of errors**. They are:

- Syntax errors
- Logical errors
- Run-time errors

```
[manju@localhost ~]$ for (( i=0; i<=2; i=i+1 )); do echo "Welcome $i times"; done
Welcome 0 times
Welcome 1 times
Welcome 2 times
```

```
[manju@localhost ~]$ array=("Albert" "is" "a" "Physicist"); echo ${!array[@]}
0 1 2 3
```

Print all array indexes

Syntax errors:

```
[manju@localhost ~]$ a=1; b=2; echo $((a+b))
 3
```

The syntax of assignment statement in C is:

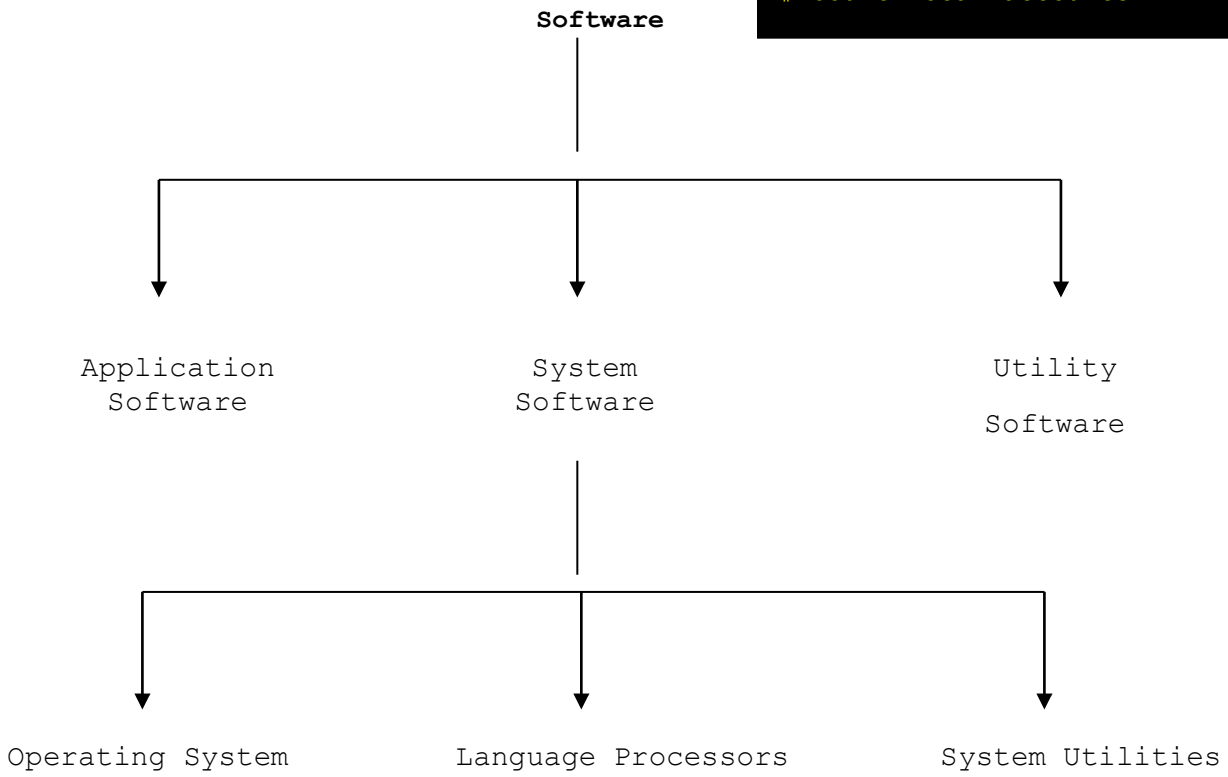
c = a+b;

If the above statement is typed without the semicolon at the end, then there will be a ***syntax error*** because of a missing semicolon.

Logical errors: A Mistake in a program's source code that result in incorrect or unexpected behavior. It is a type of runtime error that may simply produce the wrong output or may cause a program to crash while running.

Run-time errors → Device errors, improper sequencing of constructs, errors in system software, Keypunch errors, incorrect data input.

```
cat /etc/passwd|wc -l
# Count local accounts
```




```
[manju@localhost ~]$ wc -l /etc/passwd|cut -d" " -f1
43
```

Count local accounts

Recursion is the technique that defines a function in terms of itself. That is, a function which performs a particular task is repeatedly calling itself. The best example of recursively defined function is computing the **factorial of a given number**.

```
[manju@localhost ~]$ Url="www.wikipedia.org"; echo ${Url%.*}
www.wikipedia
[manju@localhost ~]$ Url="www.wikipedia.org"; echo ${Url%%.*}
www
[manju@localhost ~]$ Url="www.wikipedia.org"; echo ${Url#*.}
wikipedia.org
[manju@localhost ~]$ Url="www.wikipedia.org"; echo ${Url##*.}
org
```

```
#include<iostream>
using namespace std;

int factorial(int n);

int main()
{
    int n;

    cout << "Enter a positive integer: ";
    cin >> n;

    cout << "Factorial of " << n << " = " << factorial(n);

    return 0;
}

int factorial(int n)
{
    if(n > 1)
        return n * factorial(n - 1);
    else
        return 1;
}
```

```
[manju@localhost ~]$ echo "5 * 2.77" | bc
13.85
[manju@localhost ~]$ x=16; echo "$x * 1.5" | bc
24.0
[manju@localhost ~]$ echo "sqrt(16)" | bc
4
[manju@localhost ~]$ echo "4^2" | bc
16
```

Suppose the user enters the number 6.

main() function calls **factorial(int n)** with 6 as an passed argument

Since: 6 > 1

6 × **factorial** (6−1) = 6 × **factorial** 5 = 720

720 is returned to **main()** function by **factorial(int n)** to print

```
Factorial of 6 = 720
```

on the console screen using **printf** statement.

Description:

List the files in the bin directory.

Command:

```
ls /bin
```

Description:

List the files in the bin directory and the etc directory.

Command:

```
ls /bin /etc
```

Description:

Moves the file test.txt to the folder newrepo.

Command:

```
mv test.txt ./newrepo
```

Description:

Deletes all the lines in the test.txt containing tue word.

Command:

```
sed -i "/tue/d" test.txt
```

Git Commands

Description:

Display information about previous commits.

Command:

```
git log
```

Description:

Display information about previous commits (detailed).

Command:

```
git log --summary
```

Description:

Display information about previous commits (briefly).

Command:

```
git log --oneline
```

Description:

Obtain the repository "Git-Commands" from the URL "https://github.com/manjunath5496/Git-Commands.git**".**

Command:

```
git clone https://github.com/manjunath5496/Git-Commands.git
```

Description:

Display most commonly used git commands.

Command:

```
git help
```

Description:

Display git version.

Command:

```
git version
```

Description:

Set the basic configurations on github (your name and email).

Command:

```
git config --global user.name "myw3schools"
git config --global user.email myw3schools@gmail.com
```

Description:

Check status.

Command:

```
git status
```

Description:

List all branches (local and remote).

Command:

```
git branch -a
```

Description:

Display Git configurations.

Command:

```
git config --list
```

Description:

Add an empty file "test.txt" to an existing repo "colors".

Command:

```
touch test.txt
git init
git add test.txt
git commit -m "first commit"
git remote add origin git@github.com:myw3schools/colors.git
git push -u origin main
```

```
[manju@localhost ~]$ echo ${new:-einstein}
Einstein

[manju@localhost ~]$ echo $BASHPID
11066
```

The process ID of bash

```
[manju@localhost ~]$ find . -type f -mmin -10
./.cache/abrt/lastnotification
./.bash_history

[manju@localhost ~]$ find . -type f -mmin -10 | wc -l
2
```

Files that are modified in last 10 minutes

Number of files that are modified in last 10 minutes

- The term computer "**bug**" was, in fact, inspired by a real insect.
- Computer Programming was instrumental in helping end **World War II**
- The first programming language was called **FORTRAN**

Introduction

You might have an algorithm for getting from office to home, for making a chunk of code that calculates the terms of the Fibonacci sequence, or for finding what you're looking for in a retail store. Algorithms are the building blocks of computer programs or sequence of unambiguous instructions (the term 'unambiguous' indicates that there is no room for subjective interpretation) that tells how the problem could be addressed and solved − which is definitely overblown in their importance like road maps for accomplishing a given, well-defined automated reasoning task − which always have a clear stopping point.

Long division and column addition are examples that everyone is familiar with − even a simple function for adding two numbers is implementation of a particular algorithm. Online grammar checking uses algorithms. Financial computations use algorithms. Robotic field uses algorithms for controlling their robot using algorithms. An **encryption algorithm** transforms data according to specified actions to protect it. A search engine like Google uses search engine algorithms (such as, takes search strings of keywords as input, searches its associated database for relevant web pages, and returns results). In fact, it is difficult to think of a task performed by your computer that does not use computer rules that are a lot like a recipes (**called algorithms**).

The use of computer algorithms (step-by-step techniques used for Problem-solving) plays an essential role in space search programs. Scientists have to use enormous calculations, and they are managed by high-end supercomputers, which are enriched with detailed sets of instructions that computers follow to arrive at an answer. Algorithms have applications in many different disciplines from **science** to **math** to **physics** and, of course, computing − and provide us the most ideal option of accomplishing a task. Here is some importance of algorithms in computer programming.

- **To improve the effectiveness of a computer program:** An algorithm (procedure or formula forsolving a problem, based on conducting a sequence of specified actions) can be used to improve the speed at which a program executes a problem and has the potential of reducing the time that a program takes to solve a problem.
- **Proper usage of resources:** The right selection of an algorithm will ensure that a program consumes the least amount of memory. Apart from memory, the algorithm can determine the amount of processing power that is needed by a program.

The algorithm for a child's morning routine could be the following:

Step 1: Wake up and turn off alarm

Step 2: Get dressed

Step 3: Brush teeth

Step 4: Eat breakfast

Step 5: Go to school

Smet's Rule: A methodology for generating **joint belief functions** from conditional belief functions

Pattern mining: The process of identifying specific patterns within the data

The algorithm to add two numbers entered by user would look something like this:

Step 1: Start

Step 2: Declare variables num1, num2 and sum

Step 3: Read values num1 and num2

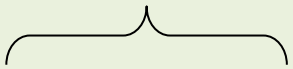
Step 4: Add num1 and num2 and assign the result to sum

sum ← num1 + num2

Step 5: Display sum

Step 6: Stop

Unique Name Assumption



Different names will always imply different elements in the model

Two of these algorithms accomplish exactly the same goal, but each algorithm does it in completely different way to achieve the required output or to accomplish our task. In computer programming, there are often many different ways – algorithms (any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values as output) − to accomplish any given task. Each algorithm has credits and demerits in different situations. If you have a million integer values between **-2147483648** and **+2147483647** and you need to sort them, the bin sort is the accurate algorithm to use. If you have a million book titles, the quick sort algorithm might be the best choice. By knowing the toughness and weaknesses of the different algorithms, you pick the best one to accomplish a specific task or to solve a specific problem.

One of the most important aspects of an algorithm is how fast it can manipulate data in various ways, such as inserting a new data item, searching for a particular item or sorting an item. It is often easy to come up with a list of rules to follow in order to solve a problem, but if the algorithm is too slow, it's back to the

drawing board. Efficiency of an algorithm depends on its design and implementation. Since every procedure or formula for solving a problem based on conducting a sequence of specified actions − uses computer resources to run − execution time and internal memory usage are important considerations to analyze an algorithm.

Why Study Algorithms?

Algorithms are the heart of computer science (usually means a procedure or basically instance of logic written in software that solves a recurrent problem of finding an item with specific properties among collection of items or transforming data according to specified actions to protect it), and the subject has countless practical applications as well as intellectual depth that is widely used throughout all areas of information technology including solving a mathematical problem (as of finding the greatest common divisor) in a finite number of steps that often involves repetition of an operation. The word algorithm − a mathematical concept whose roots date back to 600 AD with invention of the decimal system − derives from the name of the ninth century Persian mathematician and geographer, **Mohammed ibn-Musa al-Khwarizmi**, who was part of the royal court in Baghdad and who lived from about 780 to 850. On the other hand, it turns out algorithms (widely recognized as the foundation of modern computer coding) have a long and distinguished history stretching back as far as the Babylonians.

Although there is some available body of facts or information about early multiplication algorithms in Egypt (around **1700-2000** BC) the oldest algorithm is widely recognized to be valid or correct to have been found on a set of **Babylonian clay tablets** that date to around **1600 - 1800 BC**. Their exact significance only came to be revealed or exposed around 1972 when an American computer scientist, mathematician, and professor emeritus at Stanford University **Donald E. Knuth** published the first English translations of various Babylonian **cuneiform** mathematicaltablets.

Here are some short extracts from his 1972 manuscript that explain these early algorithms:-

> **"The calculations described in Babylonian tablets are not merely the solutions to specific individualproblems; they are actually general procedures for solving a whole class of problems." - Pages 672 to 673 of "**Ancient Babylonian Algorithms**".**

The wedge-shaped marks on clay tablets also seem to have been an early form of instruction manual:-

"Note also the stereotyped ending, 'This is the procedure,' which is commonly found at the end of each section on a table. Thus the Babylonian procedures are genuine algorithms, and we can commend the Babylonians for developing a nice way to explain an algorithm by example as the algorithm itself was being defined...." - Pages 672 to 673 of "Ancient Babylonian Algorithms".

The use of computers, however, has raised the use of algorithms in daily transactions (like accessing an automated teller machine (ATM), booking an air or train or buying something online) to unprecedented levels of real-world problems with solutions requiring advanced algorithms abounds. From **Google search** to morning routines, algorithms are ubiquitous in our everyday life − and their use is only likely to grow to break down tasks into chunks that can be solved through specific implementations. Many of the problems, though they may not seem realistic, need the set of well-defined algorithmic knowledge that comes up every day in the real world. By developing a good understanding of a series of logical steps in an algorithmic language, you will be able to choose the right one for a problem and apply it properly. Different algorithms play different roles in programming – and algorithms are used by computer programs **where a program –**

- Get input data.
- Process it using the complex logics.
- Stop when it finds an answer or some conditions are met.
- Produce the desired output.

To give you a better picture, here is the most common type of algorithms:

- Searching Algorithms
- Sorting Algorithms
- Path finding Algorithms
- Tree and graph based algorithms
- Approximate Algorithms
- Compression Algorithms
- Random Algorithms
- Pattern Matching
- Sequence Finding and a lot more

When a software agent is unsure how its activities will impact the environment, or when the environment is changing unexpectedly, it is faced with strategic uncertainty.

You only need to define your problem then select the right algorithm to use. The word **algorithm** may not appear closely connected to kids, but the truth is that − for kids − understanding the process of building a step by step method of solving a problem helps them build a strong foundation in logical thinking and problem solving. Here are some problems you can ask your kid to discuss algorithmic solutions with you:

- How do we know if a number is odd or even?
- How do we calculate all of the factors of a number?
- How can we tell if a number is prime?
- Given a list of ten numbers in random order, how can we put them order?

Rescaling:

↓

Multiplying or dividing a variable by some constant

Algorithms has shown it can yield results in all industries — from predicting insurance sales opportunities and generating the millions of search inquiries every day to automating medicine research, optimizing transportation routes, and much more. While algorithms help companies like **Master Card** and **Visa** to keep their users' information, such as card number, password, and bank statement safely − **algorithms aren't perfect**. They fail and some fail spectacularly. Over the past few years, there have been some serious fails with algorithms, which are the formulas or sets of rules used in digital decision-making processes. Now people are questioning whether we're putting too much trust in the algorithms. When algorithms go bad: Online failures show humans are still needed. Disturbing events at **Facebook**, **Instagram** and **Amazon** reveal the importance of context.

- **Dependent events:** Events that influence the probability of occurrence of other events
- **Independent events:** Events that does not influence the probability of occurrence of other events

Knowledge Query and Manipulation Language (KQML):

A language and interface used for transmitting knowledge and information between **virtual personal assistants** and information sources.

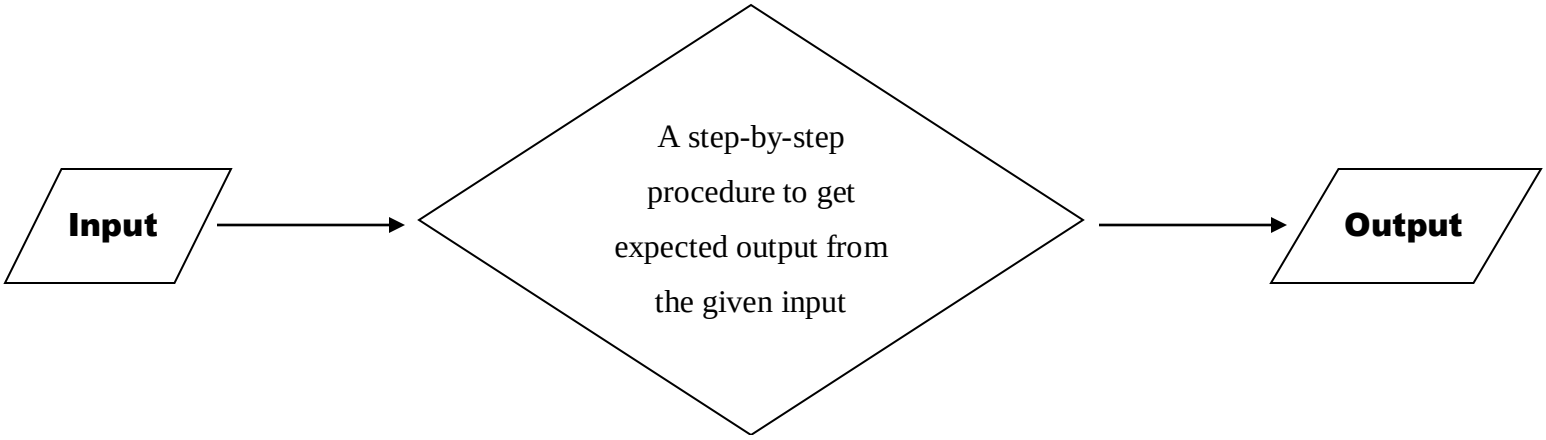
→ that continuously and autonomously conducts numerous operations on behalf of an enterprise organisation

Saddle Point:

↓

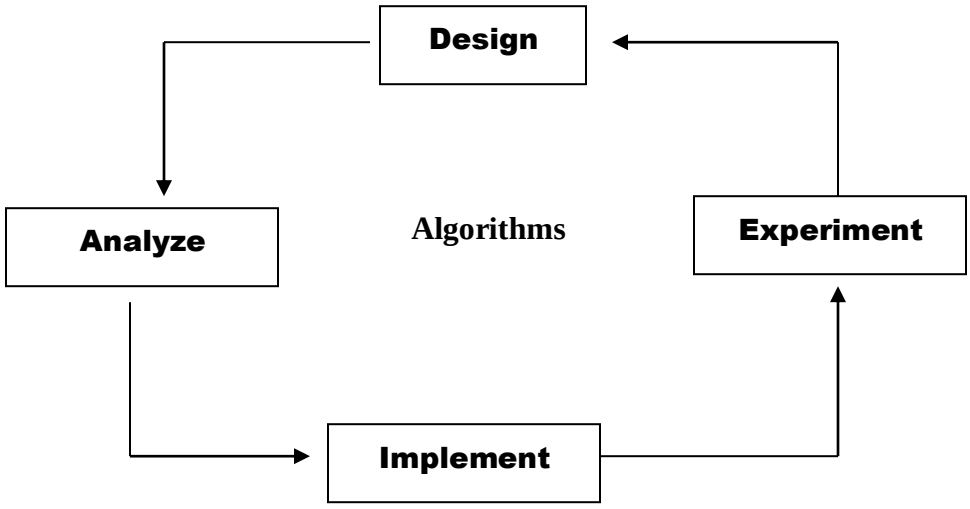
In game theory, a scenario in which the minimum of one player's maximum losses is the same as the maximum of another player's minimum losses

Scripting language	Programming language
Platform-specific	Platform-agnostic (**cross-platform**)
Interpreted	Compiled
Faster at runtime	Slower at runtime
More code-intensive	Less code-intensive
Creates standalone applications	Creates applications as part of a stack

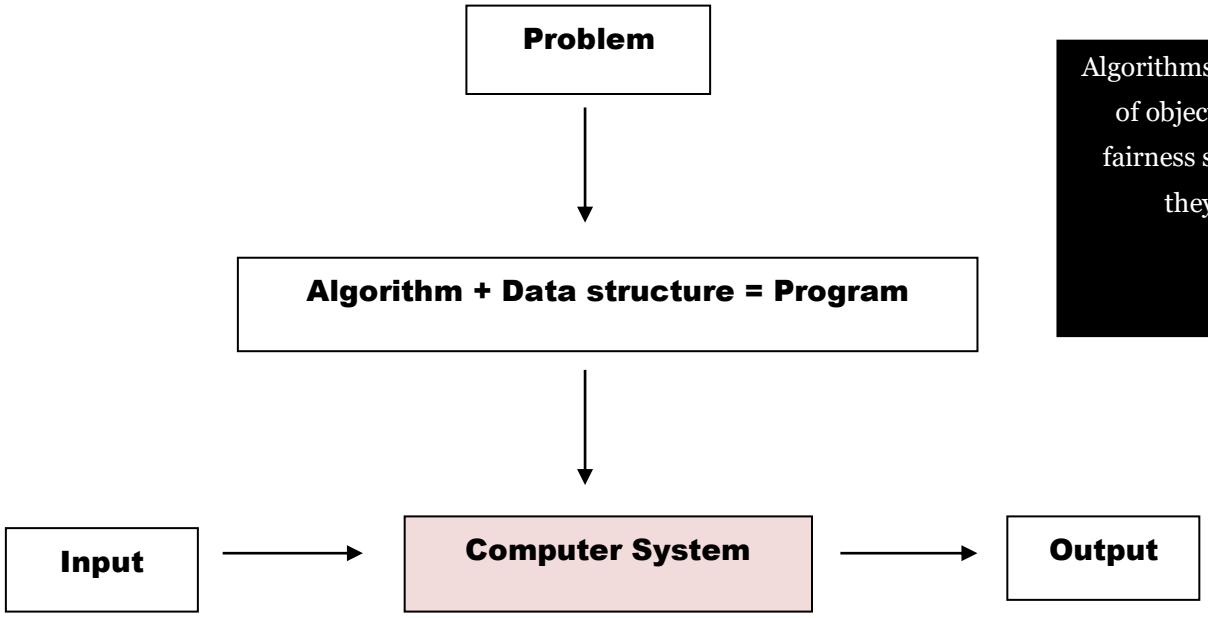


Algorithm

Priori Analysis	Posterior Analysis
checking the algorithm before its implementation	checking the algorithm after its implementation



Algorithms Design Cycle



> Algorithms are not arbiters of objective truth and fairness simply because they're math.
>
> – Zoe Quinn

Characteristics of an Algorithm

Set of rules to get the desired output from the given input

- Well-defined input
- Desired output
- Finiteness (**not end up in an infinite loops**)
- Effectiveness (**executed in finite time**)
- Definiteness (**precisely defined**)

Data structure

The way in which various **program data elements** are organized and stored into the memory so that the data can be used efficiently

```
[manju@localhost ~]$ grep -r 'Hello world' ./*
./myfile.txt:Hello world
```

Search for the string "Hello world" in files of a directory recursively

Linear Data Structure	Non-Linear Data Structure
Data elements are arranged in a linear sequence	Data elements are not arranged in a sequence
Implementation is easy	Implementation is difficult
Inefficient memory utilization	Efficient memory utilization

```
[manju@localhost ~]$ find ./* -type f -exec grep -H 'Hello world' {} \;
./myfile.txt:Hello world
```

Performance analysis of an algorithm depends on 2 factors:

- **Space Complexity:** The amount of memory space required by an algorithm to complete its task
- **Time Complexity:** The amount of time required by an algorithm complete its task

The number of operations an algorithm performs to complete its task (**considering that each operation takes the same amount of time**)

Linear Data Structure	Non-linear Data Structure
Data items are arranged in sequential order (**one after the other**)	Data items are arranged in non-sequential order (**hierarchical manner**)
Memory is not utilized in an efficient way	Memory is utilized in an efficient way
Time complexity increase with the data	Time complexity remains the same

The algorithm to find the largest number among 3 numbers:

```
Step 1: Start

Step 2: Declare variables a, b and c.

Step 3: Read variables a, b and c.

Step 4: If a > b

          If a > c
```

```
            Display a is the largest number.
         Else
            Display c is the largest number.
      Else
         If b > c
            Display b is the largest number.
         Else
            Display c is the greatest number.
Step 5: Stop
```

The time taken by the computer to run code = number of instructions × time to execute each instruction

Factors that affect run time of an algorithm:

- The hardware platform used
- Representation of abstract data types
- Efficiency of compiler
- Implementer programming skill
- Complexity of underlying algorithm
- Size of the input

- **Worst Case Complexity:** The maximum time taken by an algorithm to complete its task.
- **Best Case Complexity:** The minimum time taken by an algorithm to complete its task.
- **Average Case Complexity:** The average time taken by an algorithm to complete its task.

Run time

- Polynomial time
- Superpolynomial time

Polynomial time → run time that does not increase faster than n^k, which includes:

- constant time (n^0)
- logarithmic time (log n)
- linear time (n^1)
- quadratic time (n^2) and other higher degree polynomials (like n^3)

n → input size

Superpolynomial time → run time that does increase faster than n^k, which includes:

- exponential time (2^n)
- factorial time (n!) and anything else faster.

An algorithm is said to take **constant time** (n^0), if

- The run time of an algorithm → **constant** (doesn't increase) − no matter how large the input size increases

An algorithm is said to take **logarithmic time** (log n), if

- The run time of an algorithm increases in direct proportion to the logarithm of the input size

An algorithm is said to take **linear time** (n^1), if

- The run time of an algorithm increases in direct proportion to the input size

↓

Whenever input size doubles, the running time increases twofold

An algorithm is said to take **quadratic time** (n^2), if

- The run time of an algorithm increases in direct proportion to the input size squared

↓

Whenever input size doubles, the running time increases fourfold

An algorithm is said to take **cubic time** (n^3), if

- The run time of an algorithm increases in direct proportion to the cube of the input size

↓

Whenever input size doubles, the running time increases eightfold

An algorithm is said to take **factorial time** (!n), if

- The run time of an algorithm increases in direct proportion to the factorial of the input size

↓

Whenever input size increases by 1, the running time increases by a factor of input size

The better the time complexity of an algorithm is, the faster the algorithm will complete its task

An algorithm is said to take **linearithmic time** (n log n), if

- The run time of an algorithm increases in direct proportion to the input size times the logarithm of the input size

→ The total amount of memory space used by the algorithm to execute and produce the result

Space Complexity = Auxiliary space + Memory space used by input values

→ The extra memory space used by the algorithm during its execution

For any program, **memory space** is required for the following purposes:

- To store compiled version of instructions (**Instruction Space**)
- To store information of partially executed functions at the time of function call (**Environmental Stack**)
- To store all the variables and constants (**Data Space**)

Constant space complexity	takes the same amount of memory space regardless of the input size (n)
Logarithmic space complexity	takes memory space proportional to log n
Linear space complexity	takes memory space directly proportional to n
Linearithmic space complexity	takes memory space directly proportional to n log n
Quadratic space complexity	takes memory space directly proportional to n^2
Cubic space complexity	takes memory space directly proportional to n^3
Exponential space complexity	takes memory space directly proportional to 2^n
Factorial space complexity	takes memory space directly proportional to !n

```
#include<stdio.h>
int main()
{
  int x = 4, y = 6, z;
  z = x + y;
  printf("%d", z);
   return 0;
}
```

- In the above program, 3 integer variables are used, hence they will take up **4** bytes each, so the total space occupied by the above-given program is **4** × 3 = 12 bytes.

- In this program, we have three integer variables. Therefore, this program always takes 12 bytes of memory space to complete its execution. And because this memory space requirement is fixed for the above program, hence space complexity is said to be **constant space complexity** or O(1) space complexity.

```
public int sumArray(int[] array) {
    int size = 0;
    int sum = 0;

    for (int iterator = 0; iterator < size; iterator++) {
        sum += array[iterator];
    }

    return sum;
}
```

In the above program:

- **array** – the function's only argument – the space taken by the array is equal to 4n bytes (where n is the length of the array)
- **size** – a 4-byte integer
- **sum** – a 4-byte integer
- **iterator** – a 4-byte integer

The total memory space needed for this program to execute is 4n + 4 + 4 + 4 = 4n + 12 bytes − which will increasing linearly with the increase in the input value n, hence it is called as **linear space complexity** or O(n) space complexity.

To search an element in a given array, it can be done in two ways:

- Linear search
- Binary search

KLOC (thousands of lines of code):

A conventional metric for determining the size of a software program, as well as how long or how many people it will take to create it

Linear Search:

Linear search is a very basic and simple search algorithm. In this type of search, a sequential search is made over all elements one by one. Every element is checked and if a match is found then that particular element is returned, otherwise the search continues till the end of the data collection.

For Example:

To search the element 17 it will go step by step in a sequence order:

8	10	12	15	17	20	25
17						

Verifiability: The fact that anything can be proven to be accurate or right

Match not found

8	10	12	15	17	20	25
	17					

Code walkthrough:

A methodology for detecting algorithmic and logical problems in programming code

Match not found

8	10	12	15	17	20	25
		17				

Match not found

8	10	12	15	17	20	25
			17			

Match not found

8	10	12	15	17	20	25
				17		

Match found

Element 17 is returned.

Linear search (whose running time increases linearly with the number of elements in the array. For example if number of elements is doubled then, on average, the search would take twice as long) is rarely used practically because other search algorithms such as the binary search algorithm and hash tables allow significantly faster searching comparison to linear search.

Binary Search:

Binary Search is applied on the sorted array or list. In binary search, we first compare the value with the elements in the middle position of the array. If the value is matched, then we return the value. If the value is less than the middle element, then it must lie in the lower half of the array and if it's greater than the element then it must lie in the upper half of the array. We repeat this procedure on the lower (or upper) half of the array. Binary Search is useful when there are large numbers of elements in an array.

We shall learn the process of **binary search** with a pictorial example. The following is our sorted array and let us assume that we need to search the location of value **31** using binary search.

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9
(low)									(high)

First, we shall determine mid of the array by using this formula −

$$\text{mid} = \text{low} + \frac{(\text{high} - \text{low})}{2}$$

Here it is, $0 + \frac{(9-0)}{2} = 4$ (integer value of 4.5). So, 4 is the mid of the array.

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9
(low)				(mid)					(high)

Now we compare the value stored at location 4, with the value being searched, i.e. 31. We find that the value at location 4 is 27, which is not a match. As the value is greater than 27 and we have a sorted array, so we also know that the target value must be in the upper portion of the array. We change our low to mid + 1 and find the new mid value again.

low = mid + 1 = 4 +1= 5

$$\text{mid} = \text{low} + \frac{(\text{high} - \text{low})}{2} = 5 + \frac{(9-5)}{2} = 7$$

Our new mid is 7 now. We compare the value stored at location 7 with our target value 31.

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

The value stored at location **7 is not a match**; rather it is more than what we are looking for. So, the value must be in the lower part from this location.

We change our high to mid − 1 and find the new mid value again.

high = mid − 1 = 7−1= 6

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9
				(low)		(high)			

$$\text{mid} = \text{low} + \frac{(\text{high}-\text{low})}{2} = 4 + \frac{(6-4)}{2} = 5$$

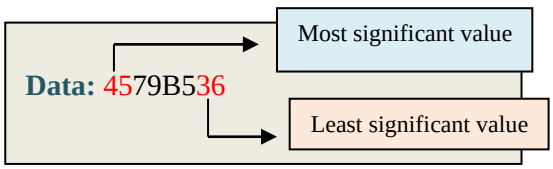
Our new mid is 5 now.

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9
				(low)	(mid)	(high)			

We compare the value stored at location 5 with our target value. We find that it is a match.

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

We conclude that the target value 31 is stored at location 5.



Big-endian	**Little-endian**
The sequence's most significant value is placed first at the lowest storage address	The sequence's least significant value is placed first at the lowest storage address

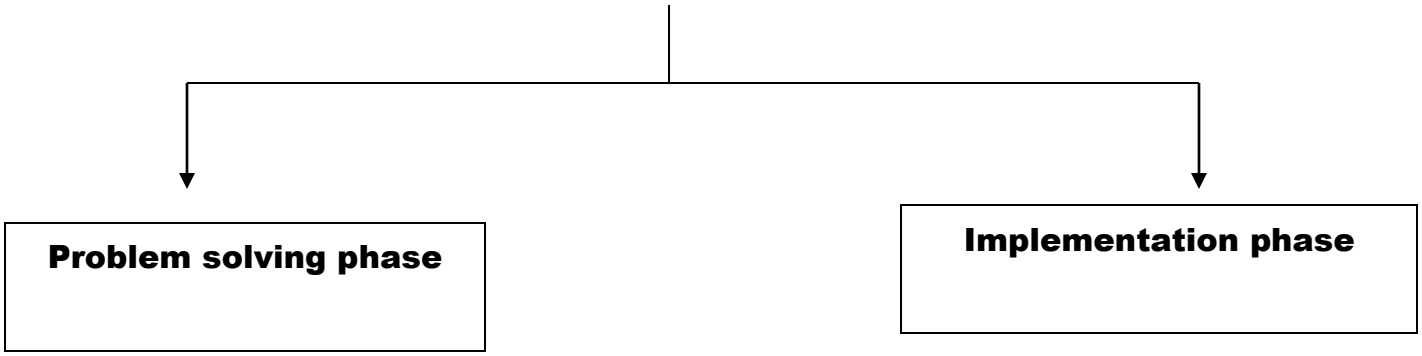
value	45	79	B5	36
Address	0	1	2	3

Increasing memory address

value	36	B5	79	46
Address	0	1	2	3

Increasing memory address

2 phases of Computer Programming Task



Pseudocode	Algorithm
A method of developing an algorithm	A finite sequence of well-defined, computer-implementable instructions, typically help to simplify and understand the problem
Easy to understand, interpret and easier ease of construction	Quite hard to understand and complex ease of construction

Pseudocode to calculate the area of a circle	Algorithm to calculate the area of a circle

AreaofCircle()
{
BEGIN
Read: Number radius, Area;
Input r;
Area = 3.14 * r * r;
Output Area;
END
}
	1. Start.
	2. Read the radius value r as the input given by the user.
	3. Calculate the area as Area: 3.14 * r * r.
	4. Display the Area.
	5. End.

Dempster's Rule of Combination

Combining evidence from multiple sources to arrive at a **degree of belief** that takes into account all available evidence

expressed by a mathematical entity called the belief function

Object-oriented programming aficionados think that everything is an object….this [isn't] so. There are things that are objects. Things that have state and change their state are objects. And then there are things that are notobjects. A binary search is not an object. **It isan algorithm.**

(Alexander Stepanov)

Dempster-Shafer Theory: A belief function and plausible reasoning-based mathematical theory of evidence that determines the probability of an event occurring by combining separate pieces of evidence

Extension of a Concept: The group of things in an application domain to which a concept refers

Anyone, from the most clueless amateur to the best cryptographer, can create an **algorithm** that he himself can't break.

(Bruce Schneier)

DENDRAL: A rule-based expert system that identifies compounds based on spectrum and nuclear magnetic resonance data

Demodulation

The rule that 0+ (0 + b) = b can be rewritten as 0 + b = b, effectively eliminating the need for the extra terms

Mathematics is as much an aspect of culture as it is a collection of algorithms.

(Carl Benjamin Boyer)

Random World Model: A reasoning approach based on the assumption that all basic events are equally probable

Jittering

↓

To avoid overplotting in statistical graphs, random noise is introduced to data

Knowledge compilation: A group of strategies for dealing with the persistent nature of a variety of artificial intelligence challenges

[**The Euclidean algorithm is**] the granddaddy of all algorithms, because it is the oldest nontrivial algorithm that has survived to the present day.

(Donald Knuth)

→ Mimic the cognitive process of a human expert on a specific topic

Knowledge Engineering:

The use of tools and techniques to develop AI expert systems

You cannot invent an algorithm that is as good at recommending books as a good bookseller, and that's the secret weapon of the bookstore -is that no algorithm will ever understand readers the way that other readers can understand readers.

(John Green)

Meta-Knowledge: Knowledge about a preselected knowledge

Meta-Reasoning: Reasoning about the process of reasoning

Meta-rule: A rule that explains how to apply other rules

The emphasis on mathematical methods seems to be shifted more towards **combinatorics** and **set theory** - and away from the algorithm of differential equations which dominates mathematical physics.

(John von Neumann)

Subjective probability: A kind of probability obtained from a person's personal assessment of the probability of a given result

Phenomenology

The study of phenomena

Data dominates. If you've chosen the right data structures and organized things well, the **algorithms** will almost always be self-evident. Data structures, not algorithms, are central to programming.

(Rob Pike)

Retrospective study

Uses previously collected data for purposes other than research

More data beats clever algorithms, but better data beats more data.

(Peter Norvig)

Real Time Operation System

respond to data at a rate that is similar to the rate at which it arrives

The Google algorithm was a significant development. I've had thank-you emails from people whose lives have been saved by information on a medical website or who have found the love of their life on a dating website.

(Tim Berners-Lee)

Many programming languages share the same structure.

Computers use binary code to store data.

```
find /* -type f -atime -20

# Find all files which have been accessed less than 20 days ago


find /* -type f -atime +20

# Find all files which have been accessed more than 20 days ago
```

Synchronous

One request at a time

Asynchronous

Multiple requests at a time

The first programmer in the entire world was a woman. Her name was Ada Lovelace and she was born in 1815. It is believed by some people that Lovelace published the first algorithm intended to be carried out by a machine.

```
find ./mydir -type f -name "*.py" -ls -delete

# Remove all "*.py" files from mydir recursively


find -size 100c

# Find files equal to 100 characters


more +3 file.txt

# Display the contents of file.txt starting from the line 3


mv * ../

# Move all files in the current folder to the upper directory
```

Action statement

A command given to the computer that instructs the computer to perform actions such as opening a file and saving a file

Concurrency

(Dealing with lots of things at once)

More than one task is processed at the same time on a single-core processor – but no two tasks are executed at the same time

Parallelism

(Doing lots of things at once)

Two tasks are executed at the same time on a multi-core processor

```
whereis -m locate
# Find help files for the command "locate"


grep -E 'He|wo' file.txt
# Display the line containing the He or wo in file.txt


[manju@localhost ~]$ date
Fri May 13 01:15:03 PDT 2022
[manju@localhost ~]$ cd !$
cd date
bash: cd: date: No such file or directory
```

Software reuse

Existing + new modules —combined→ Software program

Modularization

Software program —broken down→ modules

```
df -haT
# List all files systems and their types in a readable manner
```

Hardcode

A part of a computer program which cannot be changed in any way except by changing the entire program itself

Hardcoding

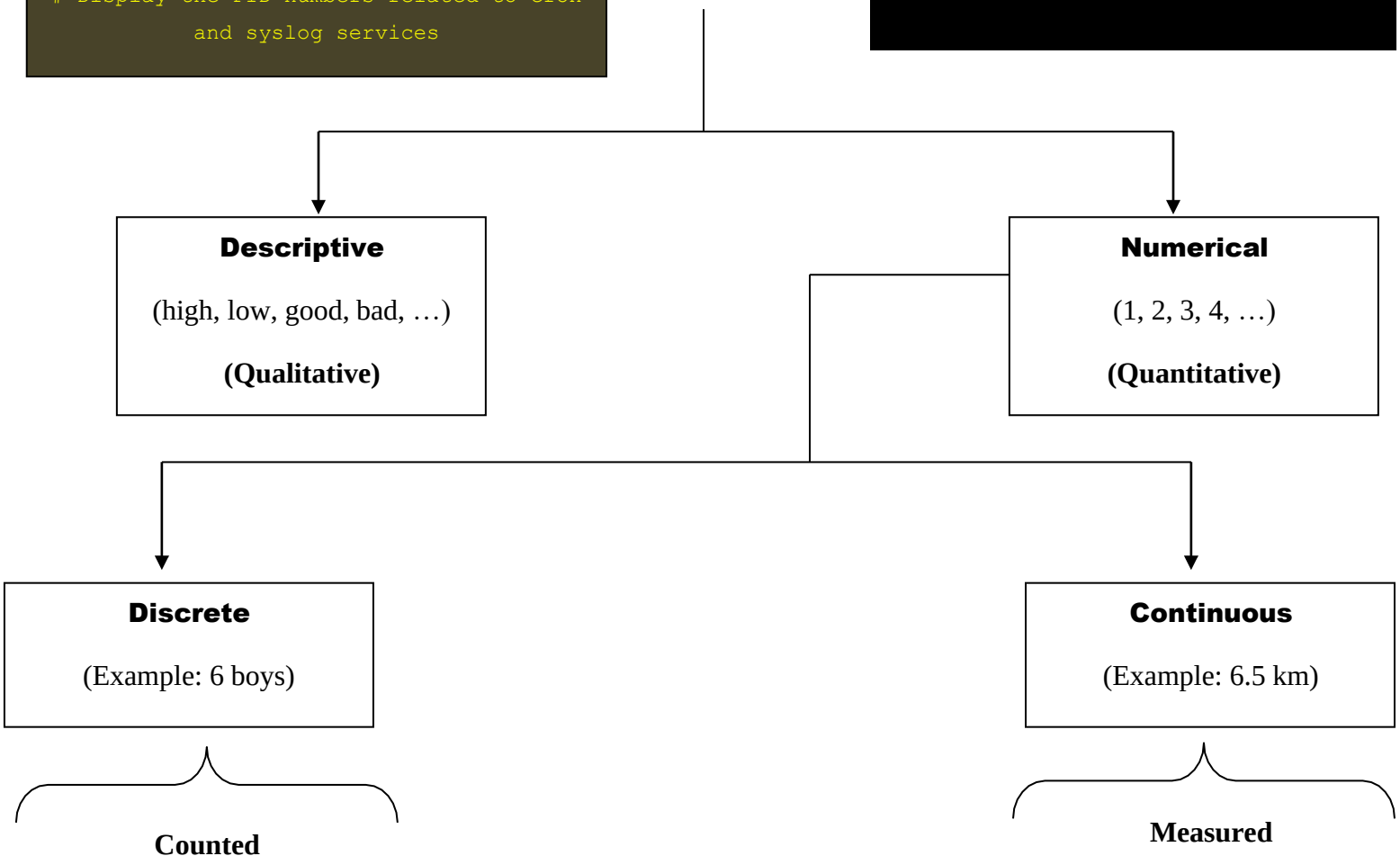
The process of creating a hardcode

Concurrent = two queues one coffee machine

Parallel = two queues two coffee machines

```
ps aux | grep '(cron|syslog)'
# Display the PID numbers related to cron and syslog services
```

Batch file: A script file that stores commands to be executed one after another



"Data" comes from a singular Latin word, **datum**, which originally meant "**something given**." Its early usage dates back to the 1600s. Overtime "data" has become the plural of datum.

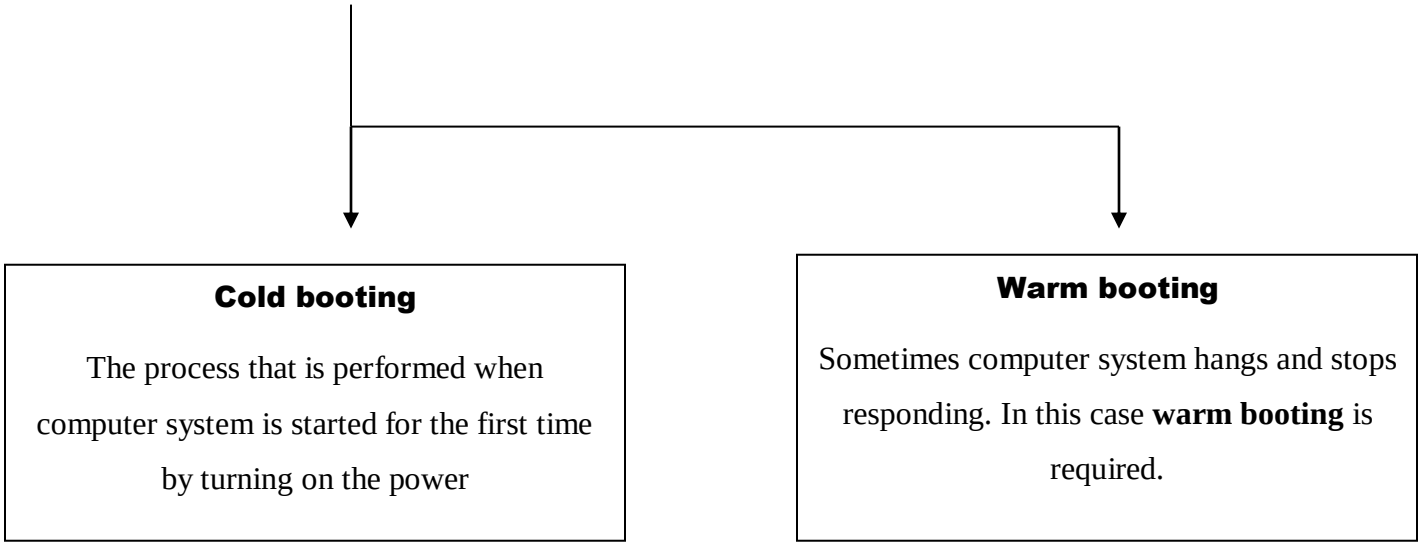
```
ls | sed "s:^:`pwd`/:"
# List file absolute paths (excluding hidden files)

find $pwd -maxdepth 1 | xargs ls -ld
# List file absolute paths (including hidden files)
```

Bugfairy: A person who reports a bug to a developer

Central Processing Unit:

- Takes all Decisions and performs all calculations.
- Controls all units of the computer and converts data into information

Booting: The process of loading the operating system



- **Data definition** → value assignment
- **Data usage** → accessing the value

Control Flow Analysis	Data Flow Analysis
Determine the control structure of a program and build control flow graphs	Determine the flow of data values and build data flow graphs

- **Native compiler:** generates an executable code for a platform on which the **compiler** is running.
- **Cross compiler:** generates an executable code for a platform other than the one on which the **compiler** is running.

The Most Significant Failures When AI Turned Rogue, Causing Disastrous Results

> Artificial intelligence will reach human levels by around 2029. Follow that out further to, say, 2045, we will have multiplied the intelligence, the human biological machine intelligence of our civilization a billion-fold.
>
> **— Ray Kurzweil**

Artificial intelligence (sometimes called machine intelligence) is a part of computer science that emphasizes the creation of intelligent machines with generalized human cognitive abilities that work and reacts like intelligent beings. Artificial intelligence has made a major breakthrough in the processes, including learning (the acquisition of information for using the data), reasoning (using rules to reach definite conclusions) and self-correction −and advancements are accelerating to present a range of new functionality for businesses. But nothing in this world can be made perfect; hence everything accompanies some notable failures and fallacies in them. Here we list some of the significant AI failures from the last decade that hint that the companies need to work harder and keep coming up with better and improved versions of their innovations.

Occam learning: An algorithmic learning approach in which the learner's goal is to produce a concise representation of received training data.

Opportunistic search: A search methodology used by systems that don't have a predetermined solution to a problem

Monotonic Reasoning	Non-monotonic Reasoning
Once a conclusion is reached, it will remain unchanged even if further data is added to the existing data	Conclusion is invalidated if further data is added to the existing data

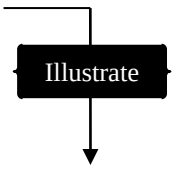
Nearest Neighbor

↓

A **methodology** for predicting or classifying observations based on the values of past observations that are "near" the target value in some aspect

Markov Decision Process (MDP): A mathematical approach for describing decision making in scenarios where outcomes are partially random and partly controlled by the decision maker

Evidence Flows



How the evaluations or possibilities for other findings are affected by a specific piece of data

Alan Mathison Turing was an English mathematician, computer scientist, logician, cryptanalyst, philosopher and theoretical biologist. Turing's Turing machine, which can be considered a model of a general-purpose computer, was crucial in the development of theoretical computer science by formalizing the ideas of algorithm and computing. Turing is usually regarded as the father of theoretical computer science and AI.

Genetic Algorithm: An approach for evaluating ML models based on methodologies adapted from the domain of genetics

In-sample Testing: A method for evaluating error rates using the same data that was used for developing the model

Problem Reduction: The process of breaking down a complex problem into smaller, manageable problems

Loebner Prize: An annual prize given to the computer software that best mimics natural human behavior

Introduction:

From self-driving cars to industrial robots, all complex real world problems are being solved with applications of intelligence (AI). Artificial intelligence (AI) is progressing rapidly and makes it possible for machines to think like humans and mimic their actions − adjust to new inputs and perform human-like tasks by processing large amounts of data and recognizing patterns in the data. While science fiction often renders AI as robots (a machine − especially one programmable by a computer − capable of carrying out a complex series of actions without conscious thought or attention) with human-like characteristics, AI can encompass anything from missile guidance to tumor detection to face recognition.

The applications for artificial intelligence are countless and **John McCarthy**, who coined the term in 1956, defines it as: "the science and engineering of making intelligent machines." The study and design of intelligent agents − where an intelligent agent is a system that becomes aware or conscious of its environment and takes actions which maximizes its chances of success − can be applied to many sectors and industries including computer science, psychology, philosophy, neuroscience, cognitive science, linguistics, operations research, economics, control theory, probability, optimization, and logic. The simulation of human intelligence in machines is being tested and used in the maintenance or improvement of health industry for dosing drugs and different treatment in patients, and for surgical procedures in the hospital operating room.

A property of machines: the intelligence that the system demonstrates − today is properly known as **Weak Artificial intelligence**, in that it is designed to perform a narrow task (such as web searches, control systems, scheduling, data mining, logistics, speech recognition, facial recognition and many others). However, the long-term goal of many technical researchers is to create Strong Artificial intelligence. While Weak Artificial intelligence may outperform humans at whatever its specific task is, like playing games or solving mathematical problems, **Strong Artificial intelligence** would outsmart humans at nearly every cognitive task.

In little over a decade, Artificial intelligence (a wide-ranging tool that enables people to rethink how we integrate information, analyze data, and use the resulting insights to improve decision making) has made leaps and bounds. Every single day, a new thousand word post showcase the most recent advancement in Artificial intelligence. Being Artificial intelligence has made remarkable breakthroughs, and many scientists dream of creating the Master Algorithm proposed by **Pedro Domingos**− which can solve all problems envisioned by humans − failure is at the core of human advancement − notable failures are emerging. From self-driving car accidents to Face ID hacks, AI didn't have a perfect year.

Transformational Grammar:
A series of transformations on a basic set of sentences can be used to construct any sentence

The Most Significant Failures When Al Turned Rogue, Causing Disastrous Results:

- **1959:** AI designed to be a General Problem Solver failed to solve real world problems.
- **1982:** Software designed to make discoveries, discovered how to cheat instead.
- **1983:** Nuclear attack early warning system falsely claimed that an attack is taking place.
- **2010:** Complex AI stock trading software caused a trillion dollar flash crash.
- **2011:** E-Assistant told to "call me an ambulance" began to refer to the user as Ambulance.
- **2013:** Object recognition neural networks saw phantom objects in particular noise images.
- **2015:** An automated email reply generator created inappropriate responses, such as writing "I love you" to a business colleague.
- **2015:** A robot for grabbing auto parts grabbed and killed a man.
- **2015:** Image tagging software classified black people as **gorillas**.
- **2015:** Medical AI classified patients with asthma as having a lower risk of dying of pneumonia.
- **2015:** Adult content filtering software failed to remove inappropriate content, exposing children to violent and sexual content.
- **2016:** AI designed to predict recidivism acted racist.

A **model's resubstitution error**

The error measured on the data used to train it

- **2016:** An AI agent exploited a reward signal to win a game without actually completing the game.
- **2016:** Video game **NPCs** (non-player characters, or any character that is not controlled by a human player) designed unauthorized super weapons.
- **2016:** AI judged a beauty contest and rated dark-skinned contestants lower.
- **2016:** A mall security robot collided with and injured a child.
- **2016:** The AI "Alpha Go" lost to a human in a world-championship-level game of "Go."
- **2016:** A self-driving car had a deadly accident.
- **2017:** Google Translate shows gender bias in Turkish-English translations.
- **2017:** Facebook chat bots shut down after developing their own language.
- **2017:** Autonomous van in accident on its first day.
- **2017: Google Allo** suggested man in turban emoji as response to a gun emoji.
- **2017:** Face ID beat by a mask.
- **2017:** AI misses the mark with **Kentucky Derby** predictions.
- **2017: Google Home Minis spied on their owners.**
- **2017:** Google Home outage causes near 100% failure rate.
- **2017:** Facebook allowed ads to be targeted to "**Jew Haters**".
- **2018:** Chinese billionaire's face identified as **jaywalker**.
- **2018:** Uber self-driving car kills a pedestrian.
- **2018:** Amazon AI recruiting tool is gender biased.
- **2018: Google Photo** confuses skier and mountain.
- **2018:** LG robot Cloi gets stagefright at its unveiling.
- **2018: IBM Watson** comes up short in healthcare.

Scrubmate:

An intelligent housekeeping robot capable of doing tasks such as floor cleaning and restroom cleaning

Tabula Rasa Theory

The theory that humans are born with no built-in mental content and they learn everything through experience or observation

While these are only a few instances of failures that have been observed so far, they are pieces of evidence to the fact that Artificial intelligence (the simulation of human intelligence processes by machines, especially computer systems) has the potential to develop a will of its own that may be in conflict with members of the human race. This is definitely a warning about the potential dangers of Artificial intelligence which should be addressed while exploring its potential interests.

"I believe there is no deep difference between what can be achieved by a biological brain and what can be achieved by a computer. It therefore follows that computers can, in theory, emulate human intelligence — and exceed it."

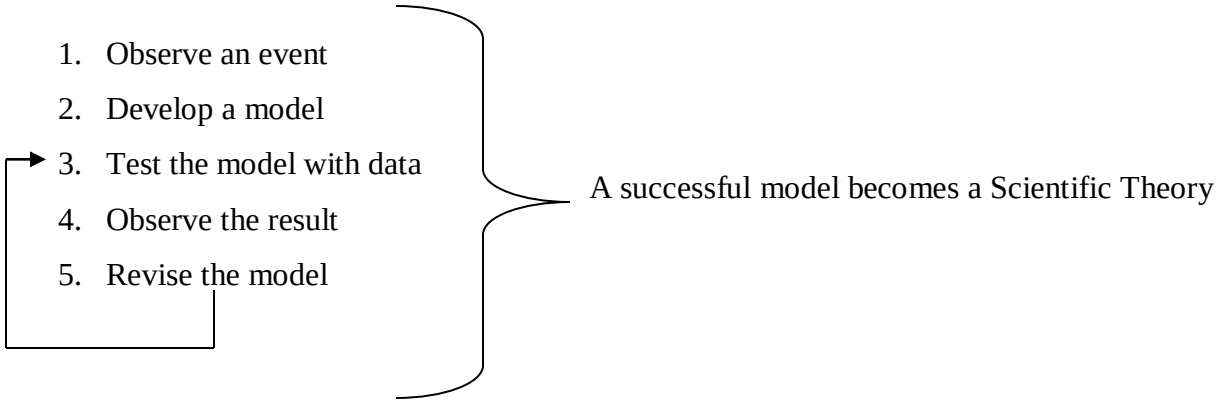
– Stephen Hawking

Artificial intelligence in general, context remains a challenge. **Despite Its Many Failures, why is artificial intelligence important?**

- Artificial intelligence automates repetitive learning and discovery through data.
- Artificial intelligence analyzes more and deeper data.
- Artificial intelligence adds intelligence to existing products.
- Artificial intelligence adapts through progressive learning algorithms to let the data do the programming.
- Artificial intelligence gets the most out of data.
- Artificial intelligence achieves unbelievable accuracy through deep neural networks – which was previously impossible. For example, your interactions with **Amazon Alexa**, **Google Search** and **Google Photos** are all based on deep learning – and they keep getting more precise the more we use them.

The threat of AI-charged job loss is spreading (AI and automation will eliminate the most mundane tasks). No matter what industry you're in, AI-powered bots (which can answer common questions and point users to FAQs and knowledge base articles) and software are taking a crack at it. Artificial intelligence seems to be ringing the death sound of a bell for all manner of jobs, tasks, chores and activities. From hospitality, to customer service, to home assistants, no job feels safe. Naturally, this has made people worried about the future. But is Artificial intelligence ready to take over our jobs, or even likely to do so ever? Prevalent AI- charged failures would suggest not.

Natural Language Processing → developed so that users can communicate with computers in human language.



"I find it useful to distinguish what I will call "strong" AI from "weak" or "cautious" AI. According to weak AI, the principle value of the computer in the study of the mind is that it gives us a very powerful tool. For example, it enables us to formulate and test hypothesis in a more rigorous and precise fashion. But according to strong AI, the computer is not merely a tool in the study of the mind; rather, the appropriately programmed computer really *is* a mind, in the sense that computers given the right programs can be literally said to *understand* and have other cognitive states."

[Searle, 1980, **Minds, Brains and Programs**]

Some definitions of AI. They are organized into 4 categories:

Systems that think like humans.	Systems that think rationally.
Systems that act like humans.	Systems that act rationally.

Lifecycle of a ML Project

Conception → Experimentation → Productionizing → Usage

Potential Risks of AI
Superintelligence (a hypothetical agent that possesses intelligence far surpassing that of the brightest and most gifted human minds)
Robotic supremacy over humans
Skynet Scenarios and AI Terrorism
A unemployed Future and Accelerated Hacking
Autonomous weapons controlled by artificial intelligence may trigger the nuclear holocaust.
The WALL-E Dystopia

John McCarthy was a computer scientist and cognitive scientist from the **United States**. McCarthy was one of the pioneers of the artificial intelligence field. He co-wrote the paper that originated the phrase "**artificial intelligence**", established the **Lisp programming language** family, influenced the design of the **ALGOL programming language**, popularized time-sharing, devised garbage collection, and was a key figure in the early development of AI.

Generate and Test Method:

A problem solving strategy that involves generating a possible solution and testing to see whether it works

Process of Computer vision

Real world → Capture image → Process image → Evaluate and deduce conclusion

Conceptual Dependency

A natural language processing theory that focuses on the representation of a language's semantics and captures the hidden meaning of a sentence and explains it very clearly

"First the machines will do a lot of jobs for us and not be super intelligent. That should be positive if we manage it well. A few decades after that though the intelligence is strong enough to be a concern"

(Bill Gates)

"I think we should be very careful about artificial intelligence. If I had to guess at what our biggest existential threat is, it's probably that"

(Elon Musk)

Character Recognition: A computer's capability to identify a character's image as a character

Adaptive Fuzzy Associative Memory: A neural network that can adapt to changing input and effectively stores both content and correlations of patterns

Advice Taker: A sophisticated AI system capable of learning from experience in the same way as humans does

Edward Albert Feigenbaum is a computer scientist who works in artificial intelligence and was a joint winner of the ACM Turing Award in 1994. He is frequently referred to as the "Father of Expert Systems."

Analytical learning: An analytical approach to learning that uses previous information as a foundation for designing and developing ideas

Q-learning: A value-based reinforcement learning approach aimed at determining the best plan of action in a particular scenario

Akaike Information Criteria (AIC):

A data-driven metric for determining which of the several models is most probably the better fit for a particular dataset

Autoepistemic Logic

A kind of logical thinking that helps us understand and describe many of the observed occurrences in our understanding of the world

Chunking

The process of gathering little bits of data and arranging them into more accessible or informative larger units

Brier score: An assessment metric for checking the accuracy of probabilistic forecasts

- A Brier score of 0 → Perfect accuracy
- A Brier score of 1 → Perfect inaccuracy

Term Frequency:

$$\textbf{TF}\ (\text{'to'}) = \frac{\text{Number of times the word "to" appears in a document}}{\text{Total number of words in the document}}$$

Inverse Document Frequency:

$$\textbf{IDF}\ (\text{'to'}) = \log\left(\frac{\text{Number of documents}}{\text{Number of documents containing the word "to"}}\right)$$

Concept learning

Learning how to distinguish and classify things

Concept Drift: The statistical features of the target variable that the model is attempting to forecast change in unexpected ways over time

Document X: I am reading comedy book

Document Y: I am watching comedy movie

- Number of words in **Document X**: 5
- Number of words in **Document Y**: 5

Closed World Assumption

What is unknown to be true is presumed to be untrue

Terms	Term Frequency (Document X)	Term Frequency (Document Y)
I	$\frac{1}{5}$	$\frac{1}{5}$
am	$\frac{1}{5}$	$\frac{1}{5}$
reading	$\frac{1}{5}$	0
comedy	$\frac{1}{5}$	$\frac{1}{5}$
book	$\frac{1}{5}$	0
watching	0	$\frac{1}{5}$
movie	0	$\frac{1}{5}$

Terms	Inverse Document Frequency
I	$\log\left(\frac{2}{2}\right) = \log(1) = 0$
am	$\log\left(\frac{2}{2}\right) = \log(1) = 0$

Decision Theory

A logical theory of mathematics that describes how to make rational decisions when the results are unknown

Data reduction: A method for increasing storage efficiency by reducing the size of data and representing it in a much lower size

Plausible reasoning: a method of deducing new conclusions from given known assumptions

Discourse Reasoning: A method for understanding a text or a conversation using natural language processing

reading	$\log\left(\frac{2}{1}\right) = \log(2)$
comedy	$\log\left(\frac{2}{2}\right) = \log(1) = 0$
book	$\log\left(\frac{2}{1}\right) = \log(2)$
watching	$\log\left(\frac{2}{1}\right) = \log(2)$
movie	$\log\left(\frac{2}{1}\right) = \log(2)$

Low-code AI

Enables anyone with little coding experience to develop AI systems

Distributed Problem Solving: The application of multiple systems to address a specific problem

Distributed AI: A branch of AI study focused on developing dispersed solutions to complex learning and decision-making challenges

Terms	TF-IDF score = TF * IDF (Document X)	TF-IDF score = TF * IDF (Document Y)
I	$\frac{1}{5} * 0 = 0$	$\frac{1}{5} * 0 = 0$
am	$\frac{1}{5} * 0 = 0$	$\frac{1}{5} * 0 = 0$
reading	$\frac{1}{5} * \log(2)$	$0 * \log(2) = 0$
comedy	$\frac{1}{5} * 0 = 0$	$\frac{1}{5} * 0 = 0$
book	$\frac{1}{5} * \log(2)$	$0 * \log(2) = 0$
watching	$0 * \log(2) = 0$	$\frac{1}{5} * \log(2)$
movie	$0 * \log(2) = 0$	$\frac{1}{5} * \log(2)$

Fault tree analysis (FTA):

A method of deductive failure analysis in which a sequence of checks is conducted logically to identify the cause of failure

Means-ends analysis: A problem-solving strategy that addresses the hurdles that exist between the initial problem state and the desired outcome

5 Major Reasons for AI Failures:

Domain theory: A mathematical theory that defines a programming language's data values and fundamental operations

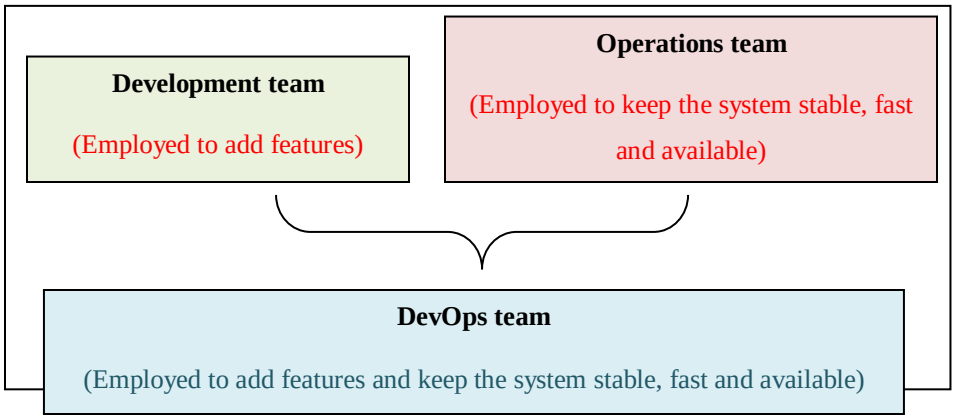
- Issues of correctness, completeness and appropriateness of data
- Incorrectly coded rules
- Misunderstanding of data relationships
- Propagation of false positives at the feedback stage
- Formatting and data reconciliation issues

Combinatorial explosion: The rapid increase in the complexity of a problem as the number of possible combinations of inputs increases

"Once computers can effectively reprogram themselves, and successively improve themselves, leading to a so-called technological singularity or intelligence explosion the risks of machines outwitting humans in battles for resources and self-preservation cannot simply be dismissed" **(Gary Marcus)**	"I don't want to really scare you, but it was alarming how many people I talked to who are highly placed people in AI who have retreats that are sort of 'bug out' houses, to which they could flee if it all hits the fan" **(James Barrat)**
"One can imagine such technology outsmarting financial markets, out-inventing human researchers, out-manipulating human leaders, and developing weapons we cannot even understand" **(Max Tegmark)** **Church's Thesis:** AI is achievable and can be implemented in computing machines	"We cannot blithely assume that a superintelligence will necessarily share any of the final values stereotypically associated with wisdom and intellectual development in humans — scientific curiosity, benevolent concern for others, spiritual enlightenment and contemplation, renunciation of material acquisitiveness, a taste for refined culture or for the simple pleasures in life, humility and selflessness, and so forth" **(Nick Bostrom)**
"The development of full artificial intelligence could spell the end of the human race. It would take off on its own, and re-design itself at an ever-increasing rate" **(Stephen Hawking)**	"We're still pretending that we're inventing a brain when all we've come up with is a giant mash-up of real brains. We don't yet understand how brains work, so we can't build one" **(Jaron Lanier)**

Is DevOps a good career?

> DevOps and its resulting technical, architectural, and cultural practices represent a convergence of many philosophical and management movements (including): Lean, Theory of Constraints, Toyota production system, resilience engineering, learning organizations, safety culture, Human factors, high- trust management cultures, servant leadership, organizational change management, and Agile methods.
>
> — **Gene Kim**



DevOps (a set of software development practices that combines software development (Dev) and information technology operations (Ops) to shorten the systems development life cycle while delivering features, fixes, and updates frequently in close alignment with business objectives) is becoming the standard way of working for Enterprises. Among the few powerful trends we had experienced in the recent times, one is undoubtedly the adoption of DevOps practices – and adoption of DevOps within the organization is rising on a broader scale, and Enterprises are trending toward it. DevOps builds upon best practices to help drive enterprise performance in modernizing environments. It offers organizations a new way to move the business forward and turn technology into a strategic advantage. An increasing number of businesses recognize the power that DevOps can bring a natural extension for Agile and continuous delivery approaches.

"At its essence, DevOps is a culture, a practice, a philosophy."

DevOps expertise is in high demand. Job postings with "DevOps" in a title or keyword are sprouting up everywhere. DevOps is an enterprise software development phrase emerging from combination of IT teams, process and products to enable the continuous delivery of value to end users. It is a firm bond between development and operations that emphasizes a shift in mindset, better collaboration, and tighter integration and aims to create a culture and environment where building, testing, and releasing software can happen rapidly, often, and more reliably, so organizations can solve critical issues quickly, and better serve their customers and compete more effectively in the market.

Everything as Code: The practice of treating **operations, infrastructure, security, configuration management and compliance** as code

Canary Release: A deployment strategy that makes new software feature available to a small percentage of users – **making sure that it is safe and functional** – before being rolled out to everyone

What is DevOps?

"A **software development method** formed out of a fundamental need that stresses communication, collaboration and integration between software developers and IT professionals." DevOps could be explained simply as operations working together with engineers to get things done faster in an automated and repeatable way.

History of DevOps

Application Release Automation

The process of packaging and deploying an application or update of an application from development to production environment

At the 2008 Agile Toronto conference, **Andrew Shafer** and **Patrick Debois** introduced the term intheir talk on "**Agile Infrastructure**". Since 2009, the DevOps term has been steadily promoted based on a simple philosophy — business works best when efforts being coordinated and collaborative — and brought into more mainstream usage through a series of "**DevOpsDays**", which started in Belgium and has now spread into Web-enabled sphere to resolve the conflict between the software developers and the operations teams when it comes to getting great work done quickly. In recent years, more tangential DevOps initiatives have also evolved, such as OpsDev, WinOps, and BizDevOps to encourage the communication between software developers and IT Operations to increase the speed at which applications being delivered.

Benefits of DevOps

Test-driven development	Behavior-Driven Development
A **software development practice** that focuses on writing the test cases prior to writing any code	A **software development practice** that focuses on understanding the intended behavior of the software and the needs of the consumers

The technical benefits include:

- Continuous software delivery
- Less complexity to manage

Business intelligence: **The process by which enterprise organize, analyze and transform raw data into useful business insights**

Lead time

↓

The time taken between the start and completion of a process

- Faster resolution of problems The cultural benefits include:
- More productive teams
- Higher employee engagement
- Greater professional development opportunities

The business benefits include:

Unit testing

The process of verifying the individual units of **source code** to validate that each unit of the source code performs as expected

- Faster delivery of features
- More stable operating environments
- Improved communication and collaboration
- More time to innovate (and not fix / keep up)

- **Smoke Testing:** Check whether the build software can be send to testing team for testing or not
- **Sanity Testing:** Check whether the code changes that are made are working as properly or not
- **Shakeout Testing:** This test is basically carried out to check the networking functionality, database connectivity and the integration of modules

Features of DevOps

- **Source control:** Software developers need to safely store their code and keep track of source- code history and versions. For this reason alone, source control is of critical importance.
- **Issue tracking system:** An issue tracking system allows everyone involved to track current issues, estimates, and deadlines.
- **Build system:** The build system supports continuous integration by building the software,running unit and integration tests, deploying to the integration environment, and performing any other automated checks defined for new versions of the software.
- **Monitoring system:** Monitoring systems continuously track all autonomous systems within the DevOps environment, notifying necessary maintenance staff if a system failure occurs.
- **Communications system:** The constant exchange of information is important so email, wiki, and a real-time chat system being enabled for effective communication and collaboration among all members of the project team.

- **Integration environment:** The integration environment hosts all the virtual machines that make up our DevOps environment
- **Code review system:** To make sure software quality, every line of code being reviewed by an experienced developer. The practice of reviewing code also accelerates career growth and learning.
- **Documentation system:** Regrettably, documentation often remains an afterthought in production software projects. To ensure that documentation being written throughout the project, an automated system being developed to allow developers to write documentation easily, along with source code.

Build Agent

A piece of software that runs a series of software build tasks called a job

DevOps Goals

- Improved deployment frequency
- To make faster time to market
- Less failure rate to new releases
- Short lead time between fixes
- Improve mean time to recovery

Software build

The construction of **executable applications** from source code

Build Automation

The process of automating the software builds

Capacity Test

A measure of the maximum number of users or transactions a software application can smoothly handle

Is DevOps a good career?

DevOps practitioners are among the highest paid IT professionals today, and the market demand for them is growing rapidly because organizations using DevOps practices are overwhelmingly high-functioning to deliver IT services that offer value to the business. According to a study on the application economy and the role of **DevOps**, 88% of enterprise IT organizations and LOB (line of business) executives already have planned to adopt DevOps sometime within the next five years to accelerate delivery of apps and offer customers with higher-quality software. In the last two years, listings for DevOps jobs at Indeed.com increased 75 percent. On LinkedIn.com, mentions of DevOps as a skill increased 50 percent. In a recent survey by Puppetlabs, half of

their 4,000-plus respondents (in more than 90 countries) said their companies consider DevOps skills when hiring.

Android	
Developer	Various (mostly Google and the Open Handset Alliance)
Written in	Java (UI), C (core), C++ and others
OS family	Unix-like (Modified Linux kernel)
Working state	Current
Source model	Open source (most devices include proprietary components, such as Google Play)
Initial release	September 23, 2008; 11 years ago
Latest release	Android 10 / September 3, 2019; 9 months ago
Latest preview	Android 11 Developer Preview 4 (RPP4.200409.015) / May 6, 2020; 38 days ago
Repository	android.googlesource.com
Marketing target	Smartphones, tablet computers, smart TVs (Android TV), Android Auto and smartwatches (Wear OS)
Available in	100+ languages
Update method	Over-the-air
Package manager	APK-based
Platforms	32- and 64-bit ARM, x86 and x86-64
Kernel type	Linux kernel
Userland	Bionic libc, mksh shell, Toybox as core utilities (beginning with Android 6.0)

Default user interface	Graphical (multi-touch)
License	Apache License 2.0 for userspace software GNU GPL v2 for the Linux kernel modifications
Official website	www.android.com

"I think right now it's a battle for the mindshare of developers and for the mindshare of customers, and right now iPhone and Android are winning that battle."

− Steve Jobs

Android application development is one of the hottest topics in the present time. To be up-to-date with the latest trends in mobile application development, one can perceive by chance or unexpectedly a plethora of tech blogs all over the internet. Contemplating Android application development is a great choice as per current market scenario and importance of Android application development for businesses of today is expanding itself, to wearable, automobiles and other areas. Applications like WhatsApp, Facebook, Twitter, Amazon etc. have brought the world around us in our handset. In a statistical study that spans the America, Europe, Asia, and the Middle East, GlobalWebIndex reports that Android tablets outnumber Apple iPad by more than 34 million and has now garnered the interest of a million smart phone users and it powers hundreds of millions of mobile devices in more than 190 countries of the world.

More than a million applications are available for download at the digital distribution platform operated by Google (double the number of apps that were available in the last few years). And more than 9 million developers write code using Java, XML (the languages that empowers an array of software intended for mobile devices that features an operating system, core applications and middleware). With the increase in the number of Android based smartphones (the devices that we started to use just for the communication purpose (i.e. for talking and messaging),

abruptly became the most powerful and dependable source of our day-to-day living) and owing to popularity of android and access of internet over mobiles, people using android smart-phones demand for new Android applications, this in turn creates an outstanding career in technology innovation (to push the boundaries of hardware and software forward to bring new capabilities to users and developers) and a demand for better applications and update for existing one.

The Mobile Application Development is the future of Software Development and Android is on the path of proving the same - according to **Google's Eric Schmidt**. Companies like Nokia, BlackBerry, Samsung, HTC, Motorola, Google and many others are going wild with their innovations to alter the software applications according to their requirements to get in touch with millions of users all over the world including their potential customer and the global client base. This adds a big sign of scope for the Android market would be beaming with lots of opportunities in the nearby future.

Access control: The process of ensuring that only authorized users can access corporate data and resources in permissible ways

Introduction

Android is the world's most popular open source mobile operating system (OS) based on the Linux Kernel − which run on 53 percent of all smartphones in the United States and on 80 percent of all smartphones worldwide − developed by Android Incorporation (a Palo Alto-based startup company, founded in 2003) and later after acquired by and further advanced by coalition of hardware, software and telecommunications companies i.e., open hand set alliance (a group of 84 technology and mobile companies including Dell, Motorola, Samsung Electronics, Sony, Intel, LG Electronics, **Qualcomm**, Broadcom, HTC, Sprint, Texas Instruments and Japanese wireless carriers KDDI and NTT DoCoMo etc.) − led by Google − designed primarily for touchscreen mobile devices such as smartphones and tablet computers. But now this technology is growing at such a rapid pace that it is going to hit the markets of Television, Cars and Wrist Watches very soon too.

Active program

The currently running program

Application Programming Interface:

A software interface that allows applications to connect with one another

Android Architecture

Data Archiving: The process of storing older data in long-term storage systems so that it can be retrieved and analyzed in the future

1. Linux Kernel

What is a Kernel? The basic layer is the Linux kernel. The whole Android OS built on top of the Linux Kernel with some further architectural changes made by Google. It is the core part of the Android Operating System that acts as an abstraction layer between the hardware and the rest of the software stack – which consists of drivers (i.e., a well-defined set of instructions – what we call programs or software written in C language that installed into mobile phones and stored in the form of files in the phone) – that tells your mobile phone how to communicate with its hardware components such as camera, display etc. – without which keypad, Bluetooth, Audio, Wi-Fi, Camera won't work properly and it is responsible for:

- **Inter Process Communication** − A Mechanism which allows applications running in different processes to share data and communicate with each other i.e., a mechanism which allows an application running in a process to send requests and receive responses from an application running in another process.
- **Power Management** (conserves power in the cost of performance and holds the device not to get to sleep state).
- **Memory Management** (make the best or most effective use of memory).

Android uses the Linux Kernel for all its core functionality such as Memory management, process management, networking, security settings etc.

Software Architect

An expert in software development who uses available technologies to solve business problems

2. Libraries

The next layer is the Android's native libraries. It is this layer that enables the device to handle different types of data. These libraries are a Collection of pre-written non-volatile data (written in C / C++ language) and pre-compiled programming codes – which support the well-functioning of android operating system.

Clear data → Unencrypted data

Some of the important native libraries include the following:

- Surface Manager / Screen Manager that supports the display screen.
- OpenGL (Open Graphics Library) that supports 3 dimensional graphics.
- SGL (Scalable Graphics Library) that supports 2 dimensional graphics.
- Media Framework that supports recording and playback of audio and video and image formats (MP3, JPG, JPEG, PNG, GIF etc.)
- Free Type that is responsible for font support (i.e., font size, color etc.)
- SSL (Secured Sockets layer) / TLS (Transport Layer Security) that is responsible for internet security and support network applications.
- WebKit that supports the display of web pages (i.e., supports inbuilt browser)
- SQLite that is responsible for storage of user data.
- Bionic is the standard C library that supports embedded Linux-based devices in mobile phones.

Background task: A task that continues to execute even when the user is not interacting with it

3. Android Run Time (ART)

Android Runtime consists of Core Java libraries and Dalvik Virtual machine.

- Java Core Libraries that consists of Java packages that enable Android application developers to write Android applications using standard Java programming language.
- DVM (**Dalvik Virtual Machine**) that is responsible to run android application.

Change management: The process of organizing and managing software changes over a network

4. Application Frame Work

Software Frame work (**written in Java language**) that supports the features of android applications and manage the basic functions of phone like resource management, voice call management etc.

Important blocks of Application framework are:

- Content Provider that enable applications to get access data from other applications (such as Contacts), or to share their own data.

- Notifications Manager that enables all applications to display custom alerts in the status bar.
- Activity Manager that manages the life-cycle of applications and provides a common navigation back stack.
- Window Manager that organizes the display screen for the application.
- Location Manager that provides the periodic updates of the geographical location of the mobile device using GPS (Global Positioning System which is a satellite-based navigation system) or cell tower.
- View Manager that manages the Application User Interface.
- Package Manager that provides information about the list of installed apps in Android Mobile Device.
- Telephony Manager that provides information about the Telephony Services (such as Phone Network, SIM Serial Number, IMEI Number etc.)
- XMPP (Extensible Messaging and Presence Protocol) that supports Online Chat Application (like Yahoo Messenger etc.)
- Resource Manager that manages the various types of resources we use in our Application and provides access to non-code resources such as localized strings, graphics, and layout files.

Batch processing	Real-time processing
Data takes time to be processed	Data is processed immediately

5. Applications

Applications are the top layer in the Android architecture. **Examples of such applications are:**

- SMS client app
- Dialer
- Web browser
- Contact manager
- Facebook
- WhatsApp

Beta test →

Make a prerelease version of **software application** available to a selected group of clients and users in order to test it.

Bitmap: An array of bits that is used to represent an image

Android Application Development Tools and IDE's:

- Android SDK (Software Development Kit) - It contains debugger, libraries, emulator, sample code, documentation and tutorials.
- Android Studio by Google (official IDE for developing Android Apps)
- Eclipse IDE using ADT plugin
- IntelliJ IDEA IDE
- NetBeans IDE

Bridge: A network device that joins together numerous local area networks to establish a larger local area network

Usage of mobile phones in India has rapidly increased from the past year and counting is still on. Out of the six billion smart phone devices in the world, close to one billion is being used in India. This comes to about 70% of our current population of India. Lots and lots of startups and other Mobile Application Development industries in India are considering Android Application Development as one of the best remunerative business opportunities. Scope of Android App Development in India is huge since every website or company in India needs its own android app (especially if it is providing a web-based service) to make their business plan into action and for capture their services in phone.

Certificate authority: A reputable entity responsible for storing, signing, issuing, distributing and revoking digital certificates

The bright future of the App Development in India can better understand with this one example. The telecommunications companies such as idea, Vodafone, MobikWik, FreeRecharge, Aircel and other cellular depends on the third-party app like, Paytm or free charge for the recharge. Thus they are making their own apps to earn direct profit from it and this is the golden opportunity for the Android Developers. In essence, India considered as a country with several globally recognized IT hubs and Android is a choice at the best for exploration in India.

Benefits of Choosing Android Application Development:

i. Android is Open Source
ii. Adaptable User Interface
iii. Massive Mobile App Market

CADAM

(Computer-Aided Design and Manufacturing)

The use of computers in product design and manufacturing

Content migration:

The process through which content is moved from one content management system or platform to another

- Google PlayStore - contains more than 2.7 million android apps
- Amazon Appstore - contains 800,000+ apps
- Aptoide - contains more than 750,000 apps
- 1Mobile Market - contains more than 800,000 apps
- Opera Mobile Store - contains more than 300,000 apps
- Mobango-contains over 100k mobile apps
- GetJar - contains over 850,000 apps

Pseudo code → neither an algorithm nor a program – an informal way of program description

{ p-code }

Advantages:

Easy to read, understand and modify.

Source computer: The computer in which a software program is being assembled or generated from machine readable codes

Idempotence: A property of certain operations in mathematics and computer science – such that no matter how many times we execute them, we get the same result

Example:

Pseudo code to perform the basic arithmetic operations:

```
read n1, n2
sum = n1 + n2
diff = n1 - n2
mult = n1 * n2
quot = n1 / n2
print sum, diff, mult, quot
end
```

Data-flow analysis: The process of analyzing how the value of a variable changes over time while the program is running

Glitch: An error in software program that causes the program to behave in unintended ways

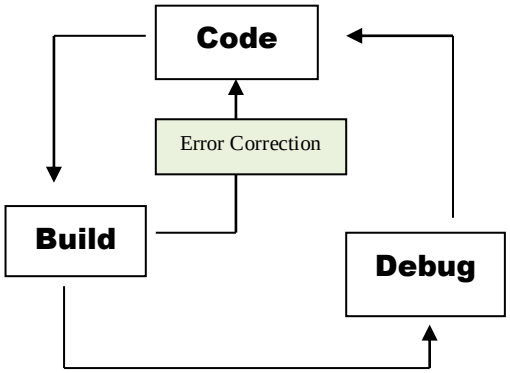
Glue code: A code that connects together different incompatible software components

Heuristic evaluation: The process of identifying problems in the user interface design and reporting issues

Hardware abstraction layer: A programming layer that enables the computer operating system to interact with a hardware device

Number of iterations:

- For a list of n elements, **Binary Search** can execute at most $\log_2 n$ times!!
- **Linear Search**, on the other hand, can execute up to n times!!



Program Development Flow

Nonexecutable statement

(Programming command that is not executed)

```
main()

{

int x = 2;
int y =3;
int z;

z = x + y;


}
```

Nonexecutable statements

Obfuscation

The act of creating **Programming source code** that is purposely complicated and made difficult for human beings to understand – help hide information about what the code performs

Non-Disclosure Agreement: An agreement developer makes to an enterprise organization – indicating he will not reveal any of the organization's intellectual property or secrets

Input size

$$T(n) \approx c_{op} C(n)$$

Running time

Number of times basic operation is executed

Execution time for basic operation

Operation code: A code that tells a computer what task to perform

Persistent memory: A memory that can be accessed even after the process that created it has terminated

Time complexity: **The time required to solve a problem of a specified size**

Expressed in terms of the number of operations used by the algorithm

Space complexity: **The computer memory required to solve a problem of a specified size**

CPU scheduling

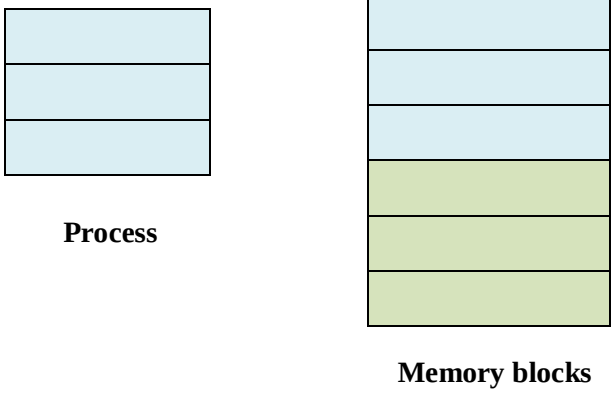
A method which permits one process to use the CPU while the execution of another process is on hold

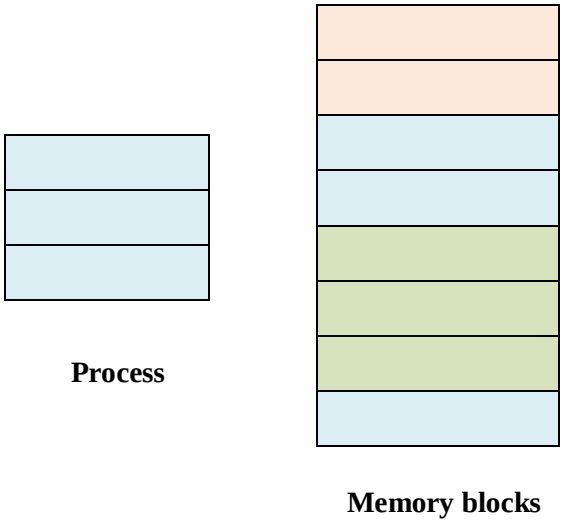
Preemptive Scheduling	Non-Preemptive Scheduling
The CPU allocated to the process can be taken back at any time during the execution of the process	The CPU allocated to the process cannot be taken back until the process completes its execution

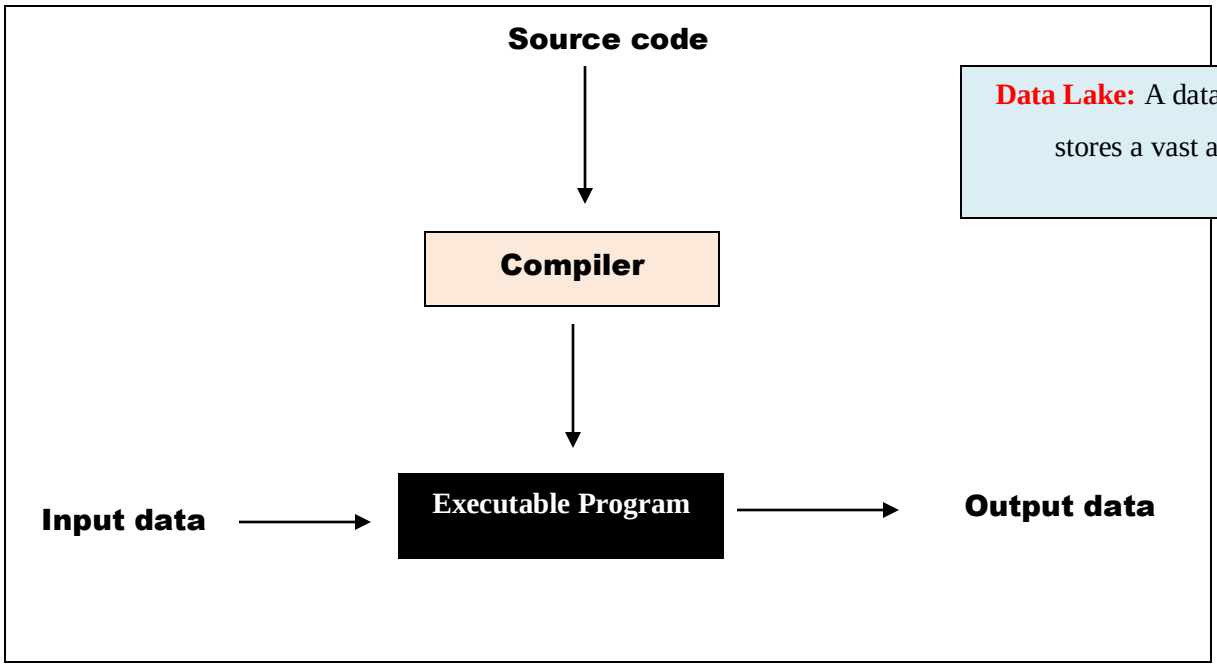
Causation

A change in one variable causes a change in another variable – which means one variable is dependent on the other

Contiguous memory allocation	Noncontiguous memory allocation
Allocation of consecutive blocks of memory to a process	Allocation of separate blocks of memory to a process







Data Lake: A data storage repository that stores a vast amount of raw data

Computer Vision

The field of computer science that focuses on extracting information from images

Generalizability

Making predictions based on past observations

Data governance: A set of principles and practices that ensure the effective and efficient gathering, storing, processing and usage of data in permitting an enterprise organization to reach its goals

Data velocity

↓

The speed at which data is processed

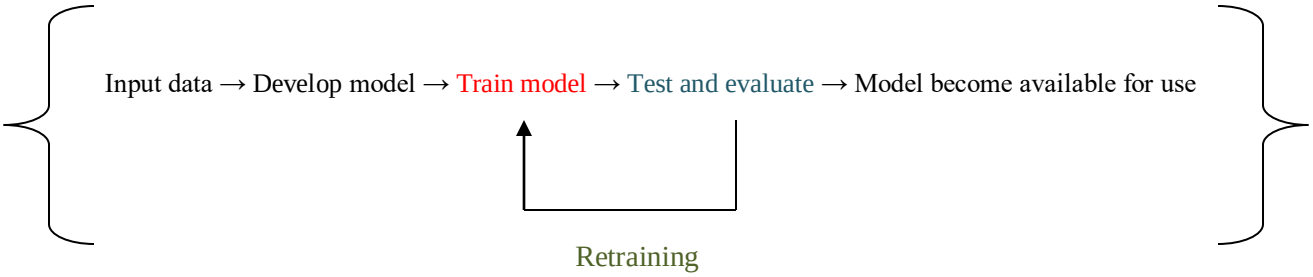
- **Target variable:** The output value which the ML model is trying to predict
- **Features:** The data used by the ML model to predict the target variable

Probability ⟶ **How likely an event is to occur**

Retraining

↓

The process of retraining a ML model with new data



Quick-and-dirty

↓

A software code written in a short time at the sacrifice of code quality – containing bugs– making it very difficult to maintain

Pseudo-operation

↓

The act of software sending instructions to a non-compiled hardware device

Overflow: A condition **that** occurs when a software program receives a value beyond the limits of its ability to handle. **For example**, if we try to fit 6 bottles in a box designed to hold 4 bottles, 2 bottles will "**overflow**" that space

at its core:

DevOps

(Development + operations)

- Communication
- Collaboration
- Integration

- Infrastructure as code
- Automation
- Continuous Integration and Deployment
- Release Management
- Continuous Monitoring
- Continuous Testing

Strong IT Performance

Benefits

- Measure of success
- Measure of failure
- Accelerate innovation time to market
- Compress testing time
- Reduce time spent troubleshooting
- Accelerate release cycles, release more frequently
- Improve application quality and performance
- Reduce downtime and unplanned outages

Page-description language: A computer language that tells a computer printer how to properly print pages sent by a computer

Underflow

A condition **that** occurs when **the result of** a mathematical operation is **smaller than** what the computing machine is capable of storing

Data type overflow error: An error that occurs when the size of data type used to store data is not large enough to hold the data.

Pastebin: A website that permits users to upload and share text, code, or other data online

DevOps

Plan	Code	Build	test	release	deploy	operate

Continuous Delivery

Continuous Integration

Agile Development

Without automation – there is no DevOps

A software library is a collection of computer programming codes used for software development

AI

Programmed to think

Robot

Programmed to do

Can AI System Work As Efficient As **Human Brain**

?????

Turing Test

↓

Created by **Alan Turing**

The **Computer** and **human** have separate discussions with the translator.
If the translator can't guess which is the Computer or if the translator gets it wrong then the computer has **Artificial Intelligence**

Includes a translator, a human and a Computer

Configuration drift: **The phenomenon where running servers in a company infrastructure become increasingly different over time − usually due to manual changes and updates on individual servers**

Major AI Approaches

- Logic Based Approach
- Pattern Based Approach

Rapid Application Development

includes

- Define the user requirements
- Build a sample model of application and test it to ensure it meets the user expectations
- Create a workable application and gather feedback
- Optimize the application and deploy it

Waterfall model

includes

Does not work well for large projects

- Define the client requirements
- Plan the software architecture based on the client requirements
- Implement the software application based on the design
- Ensure that the software application functions as designed
- Maintain and support the application in a live environment

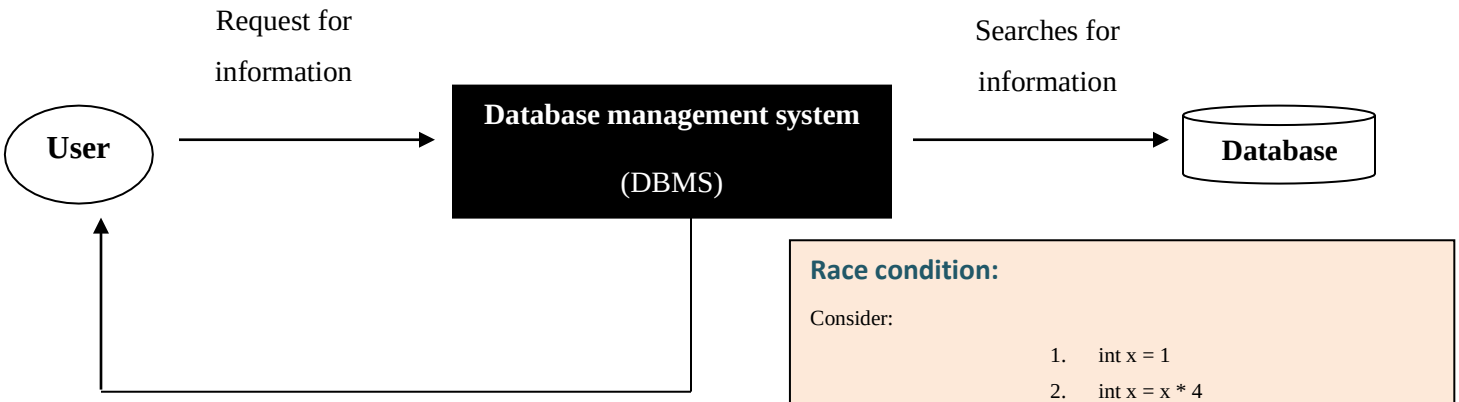
Configuration as a code: **Defining all the configurations of the servers as a code**

Release orchestration

Planning → Coding → Build → Testing → Release → Monitoring

Deployment automation

Advanced Persistent Threat: Highly Sophisticated Hacking technique used by hackers to gain access to a system on a network and remain inside for an extended length of time in order to steal highly sensitive data



Race condition:

Consider:

1. int x = 1
2. int x = x * 4
3. int y = x + 1

If the operations:

1, 2 and 3

are processed in the proper sequence

int y = 5

But if operation 3 is executed before operation 2 has completed

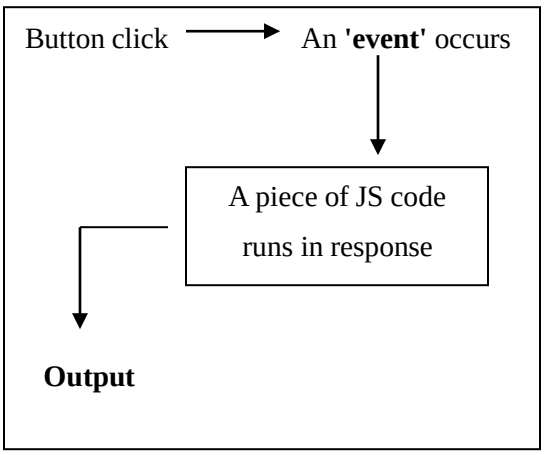
int y = 2

JavaScript

- **Usability** (faster User Interface)
- **Efficiency**
- **Event-driven**

- runs on the client's browser
- case-sensitive
- Interpreted, not compiled
- Integrates with HTML and CSS content

```
a = 1; // undefined variable → causes an error

int a = 1; // variable declared and assigned a value
                    ↳ No error
```

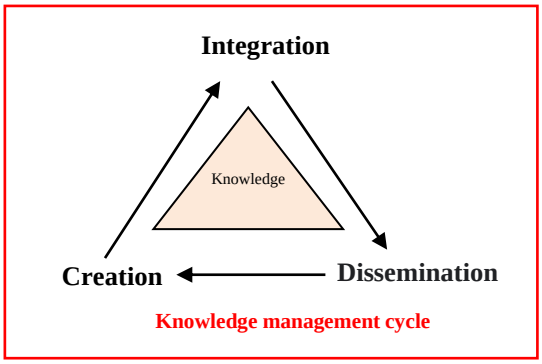


4 Comment Syntaxes

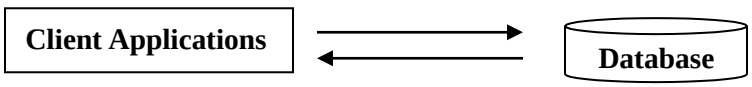
HTML	`<!-- comment -->`
CSS/JS/PHP	`/* comment */`
Java/JS/PHP	`// comment`
PHP	`# comment`

Objectives of the DBMS

- Self-describing
- Data Independence
- Multiple Views
- Multiple Users



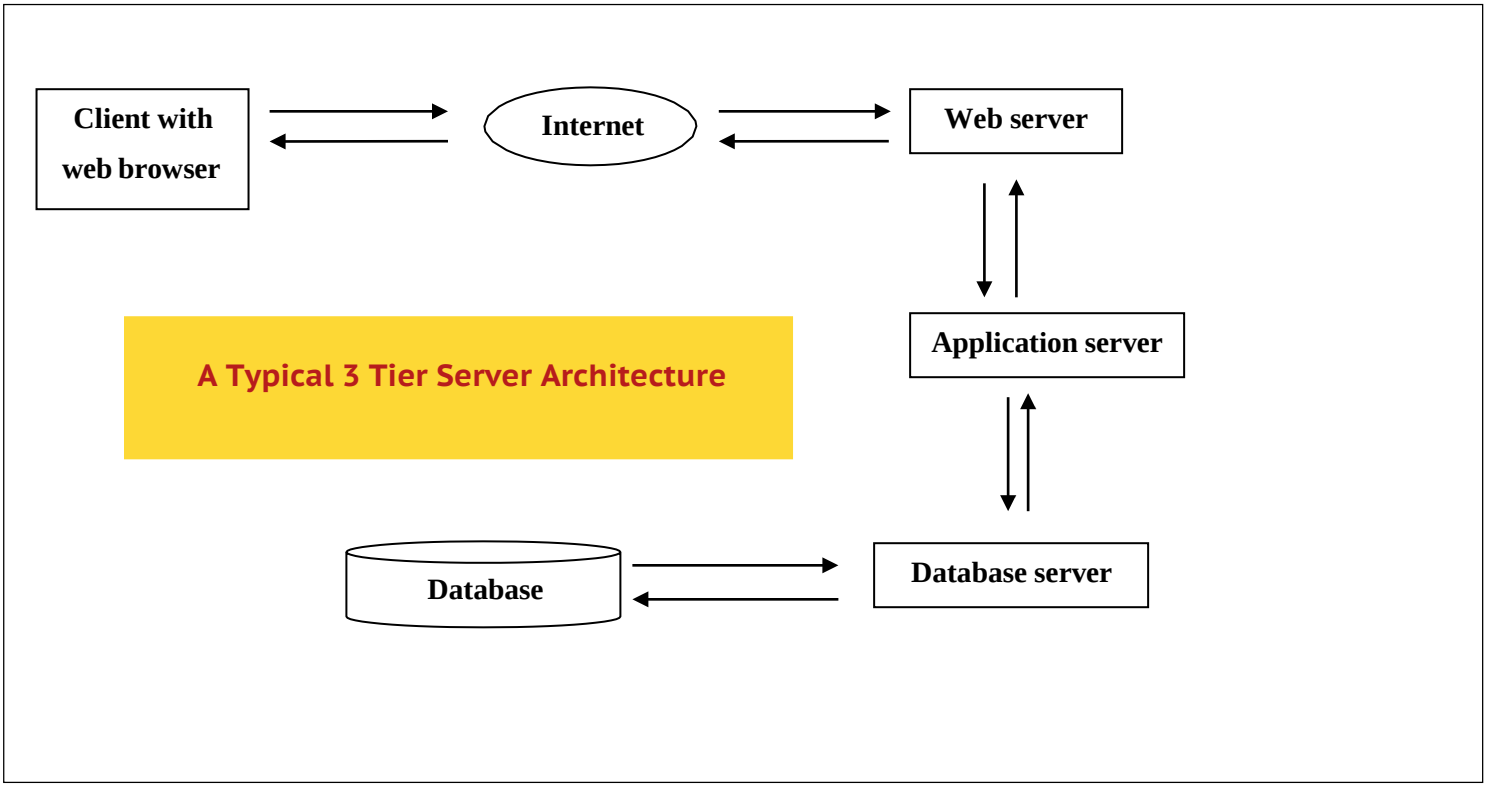
Two-tier Architecture



Distributed Denial-of-Service Attacks: A Cyberattack in which a perpetrator floods a website with high internet traffic in order to prevent targeted users from visiting the site and causing it to crash

Three-tier Architecture

Phishing: A bogus email from a bank asking you to verify your account details by clicking a link.



A Typical 3 Tier Server Architecture

Dynamic Host Configuration Protocol: A protocol that allows a server on a local network to assign a computer or other network device a temporary IP address

Errors

- **Memory errors** (memory incorrectly allocated, memory leaks, 'null pointer')
- **File system errors** (disk is full, disk has been removed)
- **Network errors** (network is down, URL does not exist)
- **Calculation errors** (divide by 0)

Exploit Kit: A toolkit used by cyber-offenders to exploit system vulnerabilities in order to carry out malicious operations

Human Firewall: Individuals who work towards reporting as well as preventing any data breaches, security violations or suspicious cyber attacks

Feature	C++	Objective C	Ada	Java
Encapsulation	Yes	Yes	Yes	Yes
Inheritance	Yes	Yes	No	Yes
Multiple Inheritance	Yes	Yes	No	No
Polymorphism	Yes	Yes	Yes	Yes
Concurrency	Poor	Poor	Difficult	Yes
Garbage Collection	No	Yes	No	Yes
Genericity	Yes	No	Yes	No
Class Libraries	Yes	Yes	Limited	Yes

Hyperconverged Infrastructure

↓

(Storage + Compute + Networking + Management)

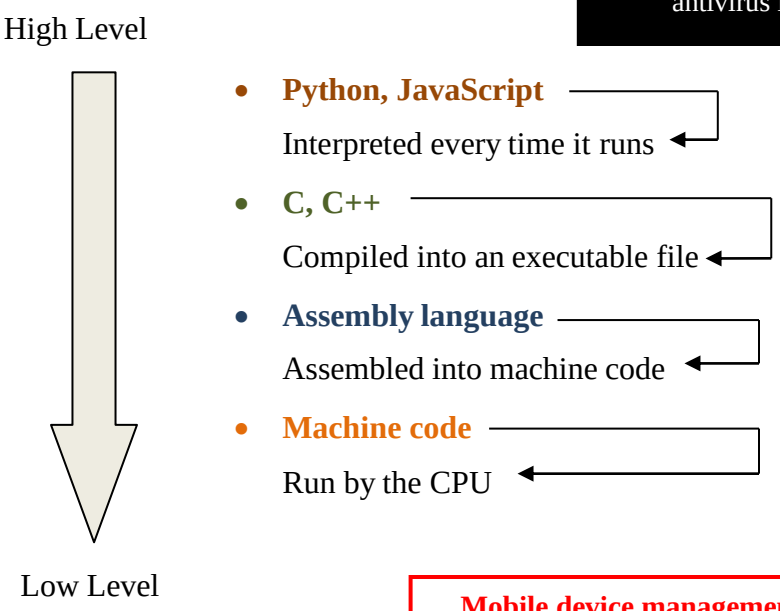
The Information Technology Infrastructure Library:

A set of detailed guidelines and procedures for providing IT services

Programming Languages

- **Interpreted languages:** An interpreter program takes in commands, checks syntax and translates to machine language at runtime. Examples of common interpreted languages are PHP, Ruby, Python, and JavaScript.
- **Compiled languages:** Programs are translated and saved in machine language by a compiler. At runtime no additional interpretation is necessary. Examples of common compiled languages are C, C++, Erlang, Haskell, Rust, and Go.

Managed Antivirus: A service in which one business organization manages all of another organization's antivirus installations on an ongoing basis



Mobile device management: The secure management of mobile devices that have access to sensitive business data

Machine language	Natural language	High-level language
Easy for computer	Easy for human	Some difficulty for both

Network News Transport Protocol: The protocol for sending, receiving, and posting network news broadcasts

Integrated Development Environments (IDE)

Consists of

- a source code editor
- build automation tools
- a debugger

Managed Detection and Response: Outsourced service that offers threat hunting services to assist enterprise organizations to protect their IT systems and respond to threats once they have been detected

Basic operation cycle of a computer processor

Instruction Phase

- **Fetch:** get an instruction from Main Memory
- **Decode:** translate it into computer commands

Execution Phase

- **Execute:** actually process the command
- **Store:** write the result to Main Memory

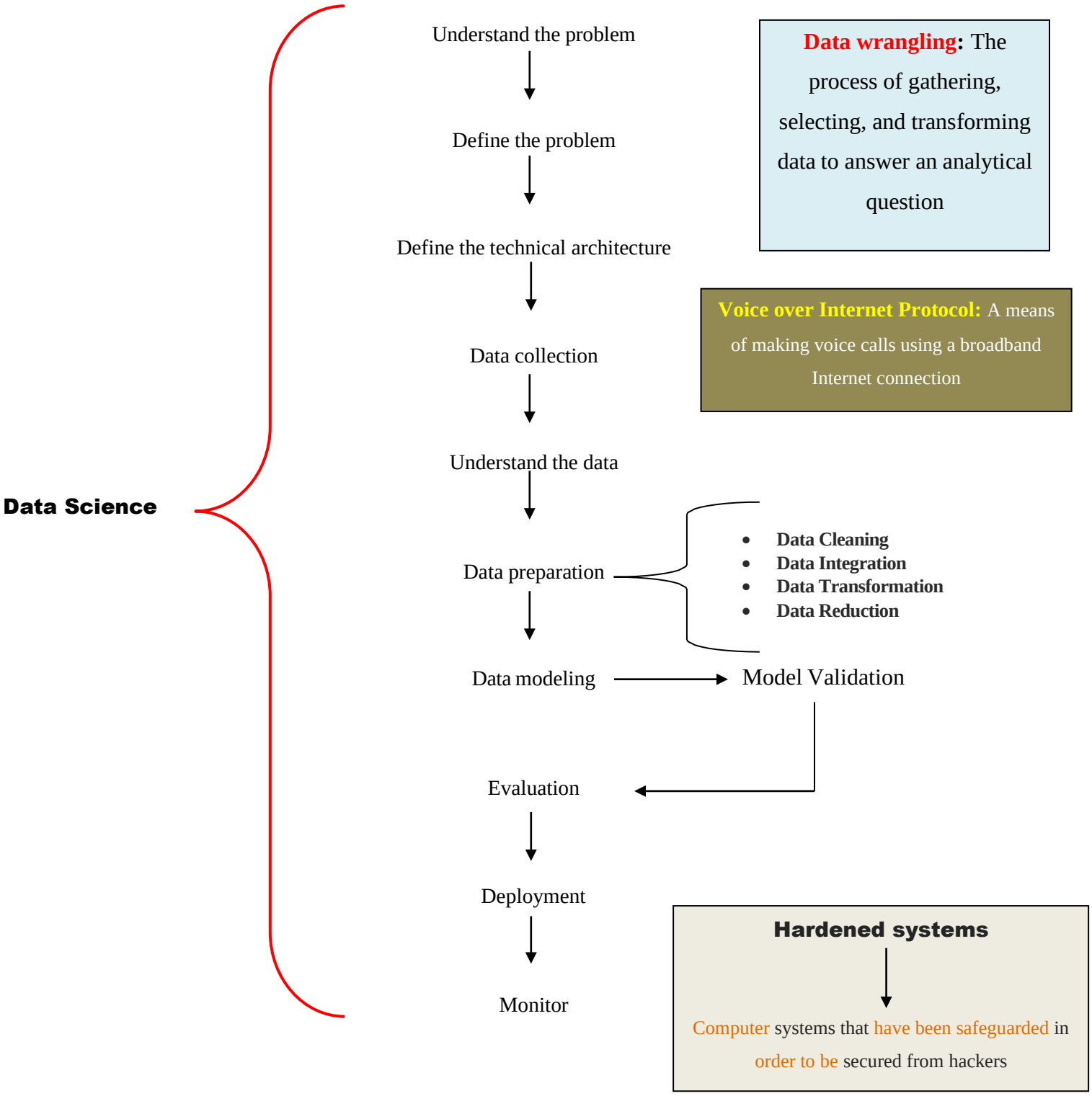
Storage Area Network: A network of storage devices that can be accessed by multiple servers

Semantic Errors → Interpreters and Compilers cannot notice them, but on execution, they cause unexpected results.

Pascal Program:

```
program Hello;
begin
  writeln ('Hello, World!')
end.
```

→ Outputs "Hello, World!" and then exits



- ETL (**Extract, Transfer and Load**) process → receive input from any source and output the **transformed data** to an analytic data model.

Network Operations Center: The team within an enterprise organization in charge of continuously monitoring a network's performance and health

Security Operations Center: The team within an enterprise organization responsible for monitoring and protecting against threats and cyber attacks

Data Preprocessing

Data Integration

- Integration of data from various sources
- Bringing integrated data into common format

Data Cleaning

- Dealing with missing data
- Remove Noise from data
- Remove outliners from data
- Remove duplicate data
- Remove inconsistencies from data

Data Transformation

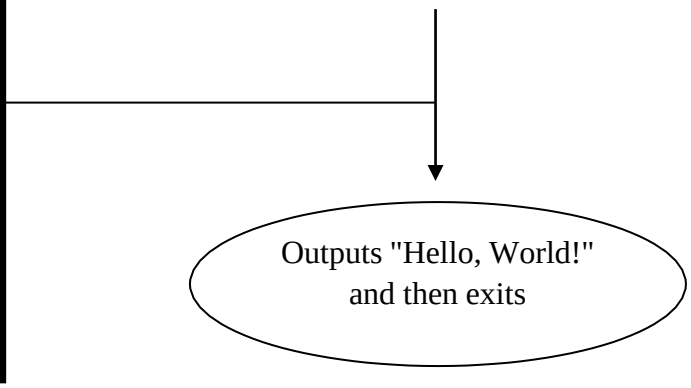
- Feature scaling
- Dealing with categorical data
- Dealing with imbalanced data
- Dimension reduction
- Feature engineering
- Train test split of data

go Program:

```
package main
import "fmt"

func main() {
    fmt.Println("Hello, World!")
 }
```

C# Program:

```
using  System;
class HelloWorld {
  static void Main() {
    Console.WriteLine("Hello, World!");
  }
}
```



Software-defined networking: A networking architecture that makes networks more flexible and manageable

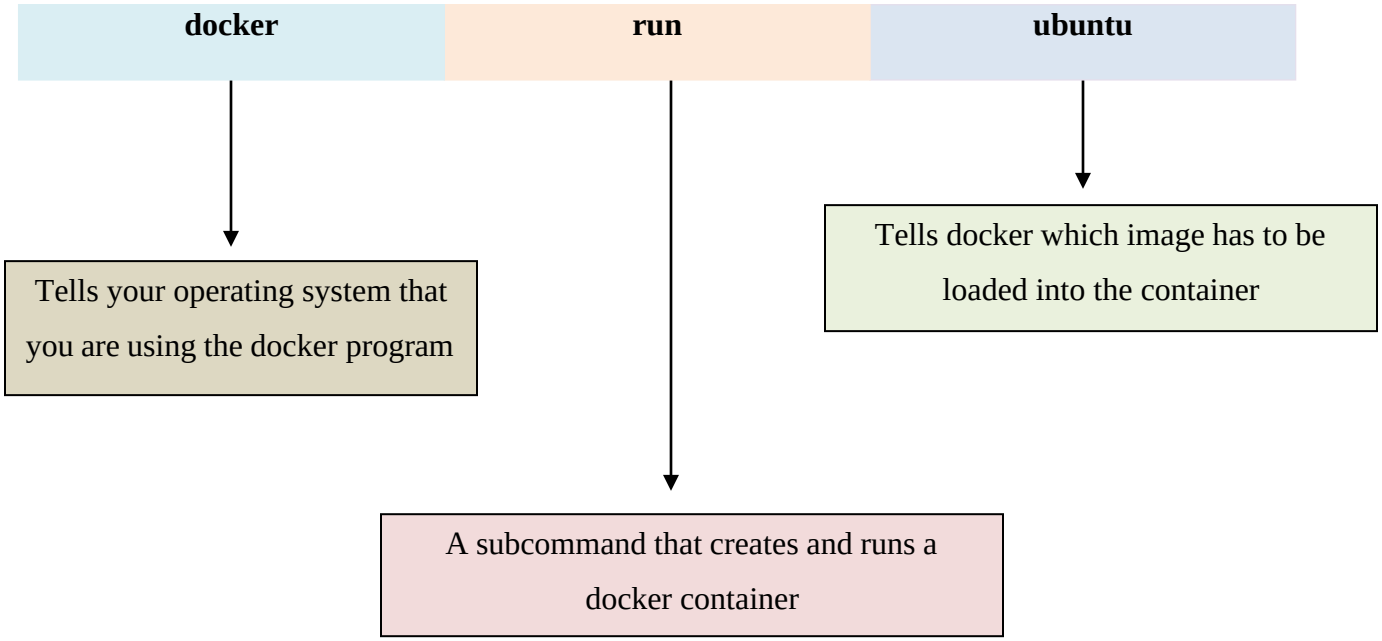
Software-defined security: A computer network security architecture and methodology for controlling information security

In case of Virtual machines	In case of Docker
16 GB Memory { VM1 (6GB), VM2 (4GB), VM3 (6GB) } VM1: • Memory Used: 4GB • Memory wasted: 2GB VM2: • Memory Used: 3GB • Memory wasted: 1GB VM3: • Memory Used: 2GB • Memory wasted: 4GB Total Memory Used: 9GB Total Memory Wasted: 7GB **7GB Memory is blocked and cannot be allotted to a new VM**	**16 GB Memory** { Container 1 (4GB), Container 2 (3GB), Container 3 (2GB) } Total Memory Used: 9GB **7GB Memory can be allotted to a new Container**

Run

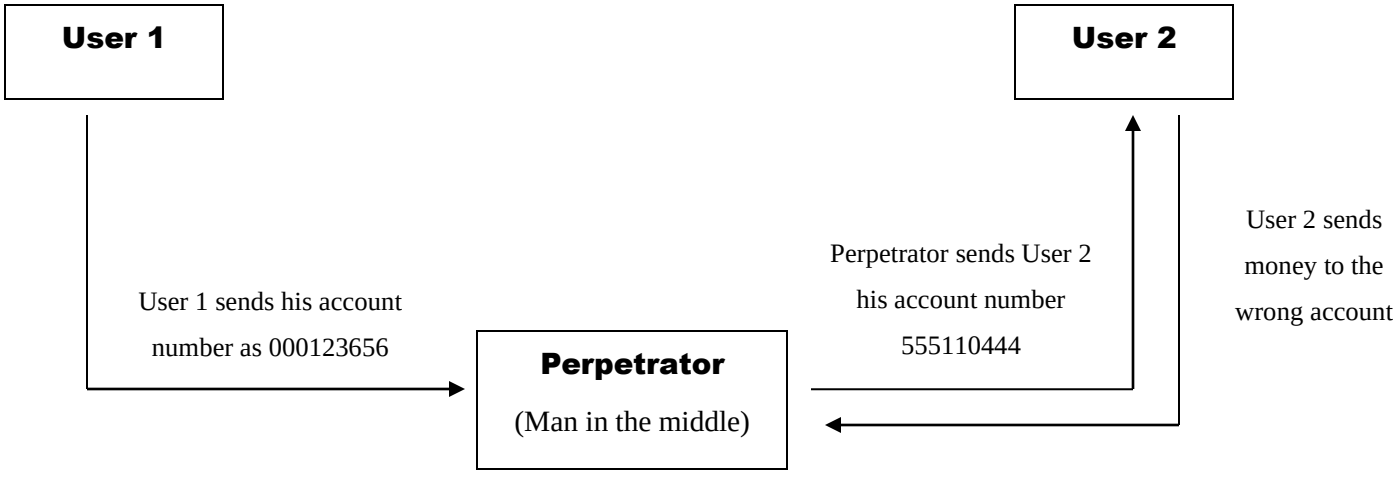
Docker image ⟶ Docker container

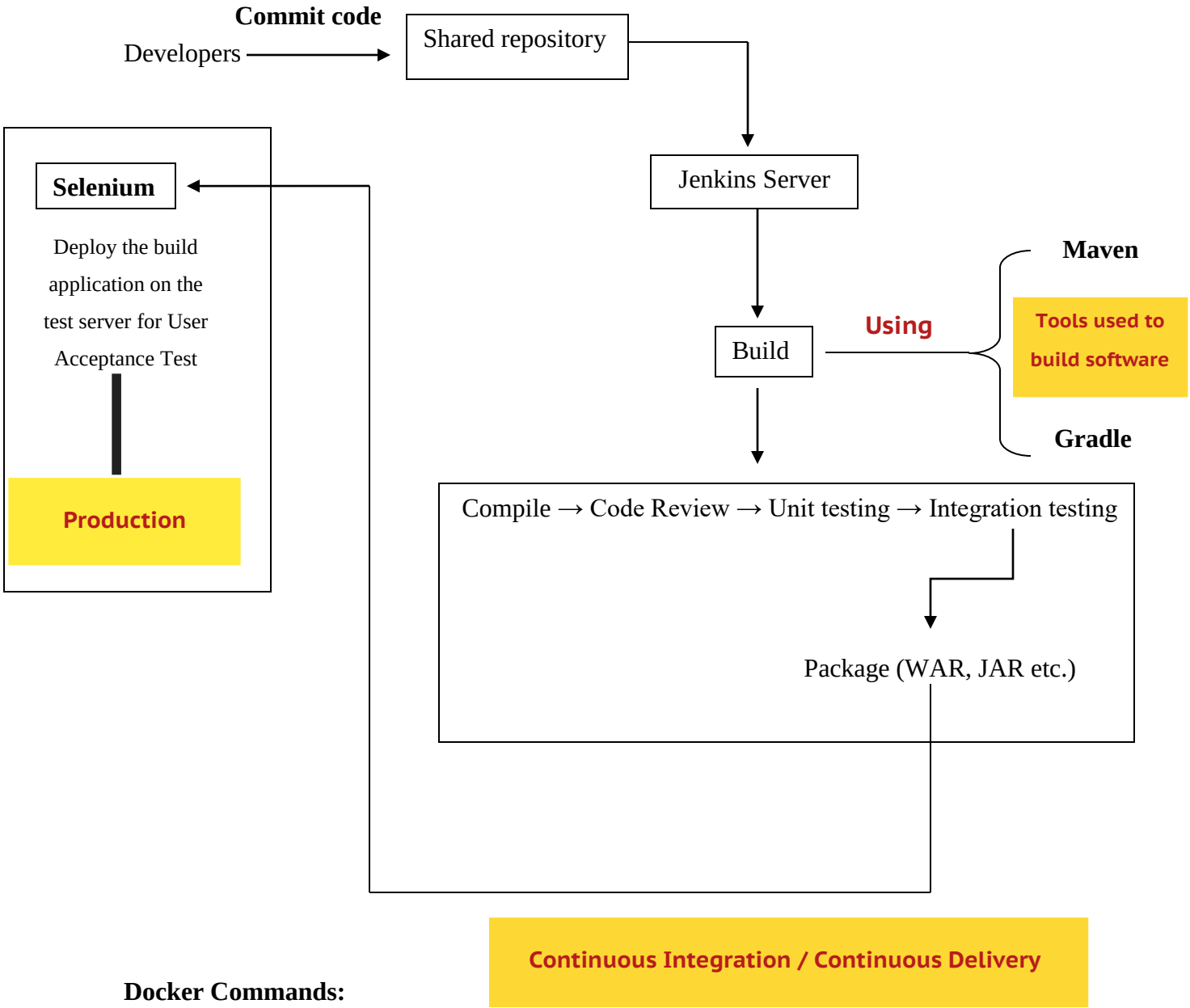
Docker command:



- **Recovery Point Objective:** The maximum amount of data loss that a company can tolerate
- **Recovery Time Objective:** The maximum amount of downtime a company can tolerate

Man in the Middle Attack





Docker Commands:

`docker --version`	get the installed docker version
`docker pull hello-world`	download the image "**hello-world**" from the docker repository (**hub.docker.com**)
`docker images`	list all the images that are locally stored with the docker engine
`docker run hello-world`	create a container from the image "**hello-world**"
`docker container ls -a`	list all containers

`docker container ls -a -s`	list the size for all containers
`docker rmi 515d5e66f68a`	remove the docker image "**hello-seattle**" with image id "**515d5e66f68a**"
`docker rm d9bf06498bb2`	remove the docker container with container id "**d9bf06498bb2**"
`docker history hello-world`	display the history of the image "**hello-world**"
`docker info`	get detailed information about docker installed on the system including the kernel version, number of containers and images, etc.
`docker volume create`	create a volume which docker container will use to store data
`docker volume ls`	list all the volumes known to Docker
`docker logs c70201336fd8`	display the logs of the docker container with contained id "**c70201336fd8**"
`docker search hadoop`	search for docker image "**hadoop**" on dockerhub
`docker network ls`	list all docker networks
`docker login`	login into docker repository (**hub.docker.com**)
`docker logout`	logout from docker repository (**hub.docker.com**)
`docker start c70201336fd8`	start the docker container with container id "**c70201336fd8**"
`docker stop c70201336fd8`	stop the docker container with container id "**c70201336fd8**"
`docker restart c70201336fd8`	restart the docker container with container id "**c70201336fd8**"
`docker inspect c70201336fd8`	get detailed information about the docker container with container id "**c70201336fd8**"
`docker stats c70201336fd8`	get the statistics of the docker container with container id "**c70201336fd8**"
`docker image ls`	List all images that are locally stored with the docker engine.

`docker system prune`	delete all unused containers, unused networks, and dangling images
`systemctl status docker`	check the Docker service
`systemctl start docker`	start the Docker service
`docker image prune`	remove unused images
`docker save hello-world > hello-world.tar`	save the image "**hello-world**" to a tar archive
`docker load < hello-world.tar`	load the image "**hello-world**" from the saved tar file
`docker export a27999b71e62 > hello-world.tar`	export the docker container with container id " **a27999b71e62**" as a tar archive
`docker import hello-world.tar`	import the contents from hello-world.tar

Vishing: The fraudulent practice of making bogus phone calls or leaving voice messages to conduct phishing attacks

jQuery

- Simplifies JavaScript
- Offers built-in animation effects
- Offers useful functions to capture a wide variety of events

There are two ways to create a thread in Java:

- By implementing the Runnable interface
- By extending the Thread

Unified Threat Management

A method of information security in which a single hardware or software installation provides multiple security features and services

Benefits of Cybersecurity:

- Protection for Data and Networks
- Prevention of Unauthorized User Access

Annotation:

The technique of labeling the unlabeled data. **For example**, assigning a value of 4 to the image of 4 in the handwritten digit recognition task

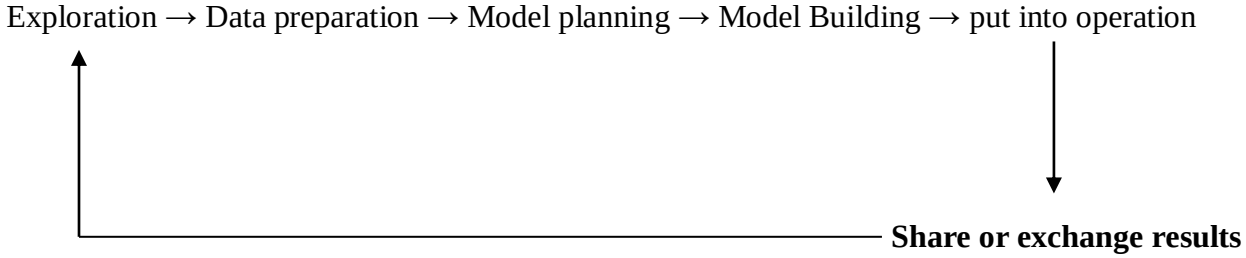
Bias Error:

Inaccuracy resulting from **algorithm's ability** to repeatedly learn the incorrect idea by not accounting for all the information in the data

Is this X or Y?	Classification Algorithm
Is this strange?	Anomaly Detection Algorithm
How much or how many?	Regression Algorithm
How could this be coordinated?	Clustering Algorithm
What should I do next?	Reinforcement Learning Algorithm

Pigeon Hole Principle:

There must be more than one item in at least one container if "x" items are placed in "y" containers, where x > y



Data Science Lifecycle

Java Programming Language

Paradigm	Multi-paradigm: generic, object-oriented (class-based), imperative, reflective
Designed by	**James Gosling**
Developer	Sun Microsystems
First appeared	May 23, 1995; 25 years ago
Stable release	Java SE 14 / March 17, 2020; 2 months ago
Typing discipline	Static, strong, safe, nominative, manifest
Filename extensions	.java, .class, .jar
Website	oracle.com/java/

James Gosling

Creator of Java

Influenced by
CLU, Simula67, LISP, SmallTalk, Ada 83, C++, C#, Eiffel, Mesa, Modula-3, Oberon, Objective-C, UCSD Pascal, Object Pascal

Influenced
Ada 2005, BeanShell, C#, Chapel, Clojure, ECMAScript, Fantom, Gambas, Groovy, Hack, Haxe, J#, Kotlin, PHP, Python, Scala, Seed7, Vala

Java is one of the most used **class-based, robust, secure popular programming languages** used in the development of virus-freesystems **because:**

- Java avoids using **pointers** because they allow for direct access to memory locations, which poses a security risk
- Java programs run within a **virtual machine sandbox** where a security policy is in place, preventing the Java code from opening network connections, reading from or writing to the hard drive, and other malicious activities

and a open-source and free high level **object-oriented** programming language and a computing platform for application development conceived by **James Gosling**, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at **Sun Microsystems**, Inc. in 1991 to create programs to control consumer electronics (**which is now a subsidiary of Oracle Corporation**) and released in 1995, runs on a variety of platforms, such as Windows, Mac OS, and the various versions of **UNIX**, used in internet programming, mobile devices, games, e-business solutions etc., because of its **reliability**, high performance, simplicity and easy to use and **quick to learn** and rigid versus extensibility. Since **Java** has a runtime environment **(JRE)** and **API**, it is called a **platform**. As a language that has the **Object-Oriented feature**, Java supports:

- Polymorphism
- Inheritance
- Encapsulation
- Abstraction
- Classes
- Objects
- Instance
- Method
- Message Passing

Advantages:

```
/*
This is a simple Java program.
Call this file "MyClass.java".
*/

public class MyClass {
// The program begins with a call to main()
public static void main(String args[]) {
// This declares integer variables 'a' and 'b'
int a, b;
// This assigns 'a' the value 10 and 'b' the value 30
a = 10; b = 30;
if(a < b) System.out.println("a < b");
a = a * 3; // a = 10 * 3 = 30
if(a == b) System.out.println("a = b");
a = a * 3; // a = 30 * 3 = 90
if(a > b) System.out.println("a > b");
// This won't display anything
if(a == b) System.out.println("We would not see this");
}
}
```

- Object Oriented
- Platform Independent
- Simple
- Dynamic
- Secure
- Multi-threaded
- Architecture-neutral
- Portable
- Robust

Output:

```
a < b
a = b
a > b
```

```
// Compute the area of a rectangle

public class MyClass {
public static void main(String[] args) {
int l, b, a;
l = 12; // Length of rectangle
b = 13; // Breadth of rectangle
a = l * b; // compute area
System.out.println("Area of rectangle is: " + a);
}
}

// Output:
            Area of rectangle is: 156
```

Types of Java Applications:

- Standalone Application
- Web Application
- Enterprise Application
- Mobile Application

Java Language Keywords:

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

Text file named MyClass.java

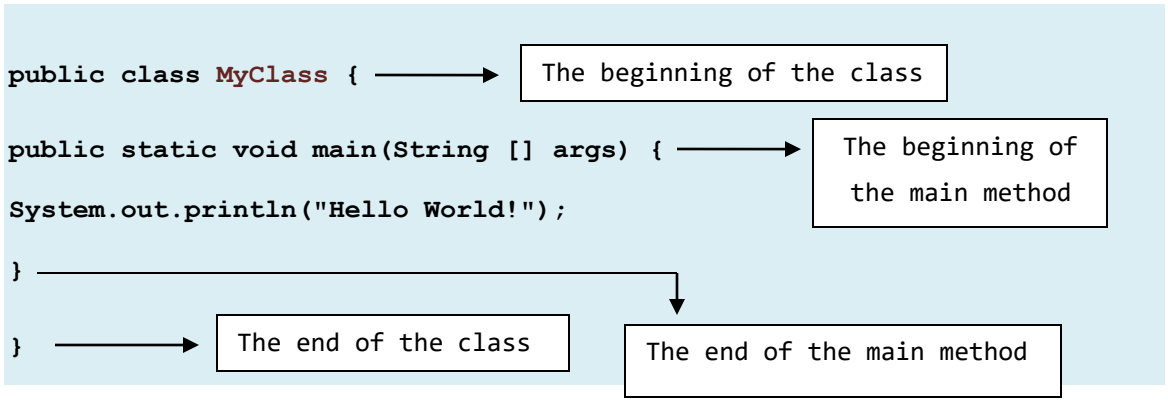
```
// Declare a class with name "MyClass"
public class MyClass {
// Declare the main method at which the program will begin executing
public static void main(String[] args) {
// println() statement which displays the text "Hello World!"
System.out.println("Hello World!");
} // ends main()
} // ends the MyClass class definition
```

```
byte x = 25;
x = x * 3; // Error: Cannot assign an int to a byte


byte x = 25;
x = (byte)(x * 3); // Outputs the value of 75
```

Process of Java program execution:

All method names should start with a Lower Case letter. For all class names the first letter should be in Upper Case.

A Java program:



is written using **Text Editor**, such as [**Notepad++**, Notepad] and saved with **.java extension.**

File saved with **.java extension** is called **Source Program** or Source Code.

```
// MyClass.java
```

Single line comment

The source filename "**MyClass.java**" must match the **class name** in the source file. If they don't match, we will get a **compilation error**.

```
public class MyClass {
public static void main(String [] args) {
System.out.println("Hello World!");
}
}

/* Because the class name is "MyClass" the source file should be named
as MyClass.java */
```

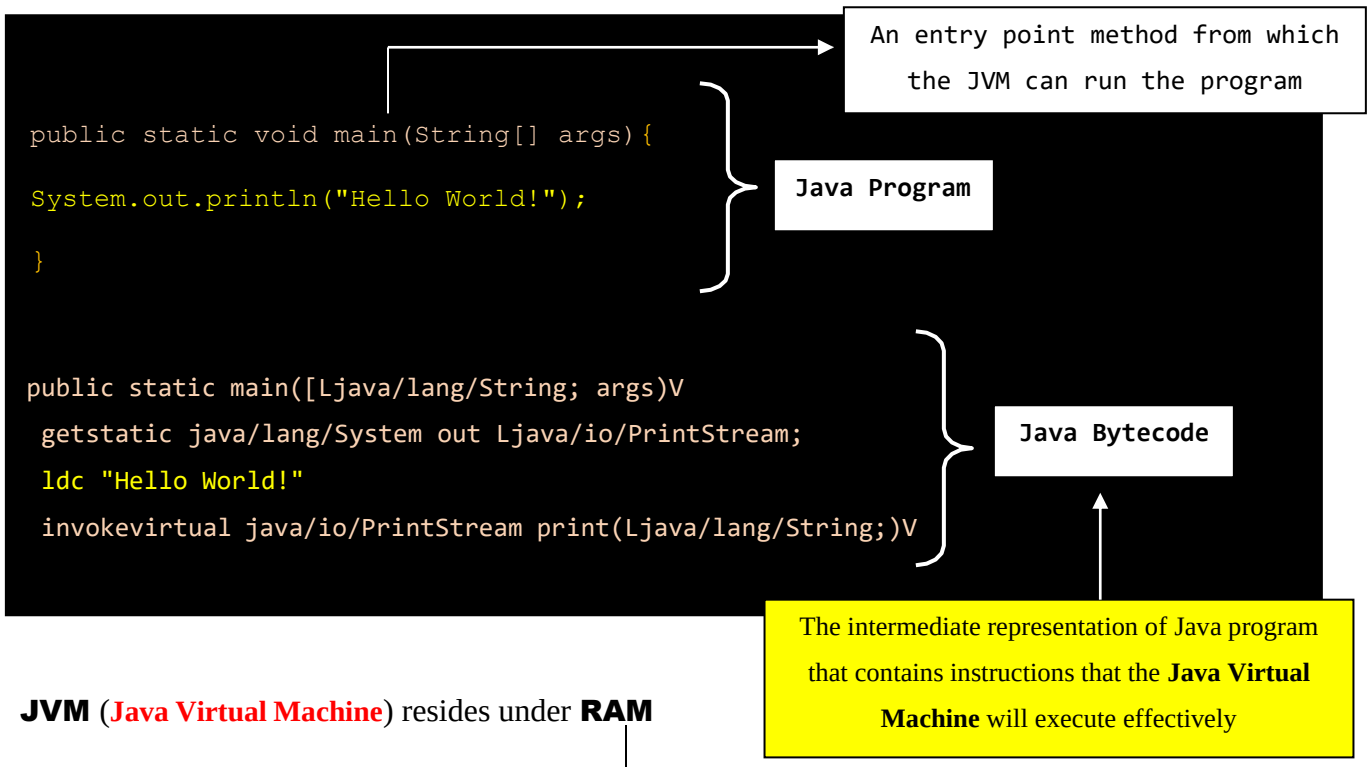
Multi-line comment

and sent to the **java compiler** (i.e., **javac compiler**) where the source program is compiled i.e., the program is entirely read and translated into **Java byte codes** (but not into **machine language**). If the **javac compiler** finds any error during compilation, it provides information about the error to the programmer. The programmer has to review code and check for the solution. And if there are no errors the translated program (i.e., **java byte codes** – a highly optimized set of instructions) is stored in computers main memory as **MyClass.class** and since the **java byte codes** cannot be trusted to be correct. Therefore before execution they are verified and converted to machine level language i.e., machine code sequence of 0s and 1s by **Java run-time system**, which is called the **Java Virtual Machine** (**JVM**) and is executed by a Java interpreter and

```
Hello World!
```

is displayed on the console screen.

```
// Converting int datatype to String
public class MyClass {
public static void main(String[] args) {
int a = 54; // Declare integer variable 'a'
String x = String.valueOf(a);
System.out.println(x);
// Output: 54
// string now equals "54"
}
}
```



JVM (**Java Virtual Machine**) resides under **RAM**

> **Random Access Memory** – the stuff that boosts up your computer to run faster and allows your computer to perform many tasks at the same time

and **JVM** comprises:

- **Class Loader:** Loads .class file that contains Java bytecodes.
- **Byte Code Verifier:** Verifies byte codes.
- **Execution Engine:** Translates java byte codes to machinecodes and executes them.

In the statement:

```
public class MyClass
```

The word "`MyClass`" implies: name of the class is **MyClass** and this class is public. **public** means that the class "**MyClass**" can be accessed by any other class in any package.

Java program processing starts from the **main() method** which is a mandatory part of every Java program. `public static void main(String [] args)`

main method → A collection of statements or methods like `System.out.println()` that are grouped together to perform an operation

and this **main method** is public indicating that a method may be called from anywhere, including outside of a program. Every Java program should have one **main method**.

main method in java functions like **main function** in C and C++

Like **C** and **C++**, **Java** is also a case sensitive language i.e., capital letters (or **upper case letters**) must be avoided to prevent the display of error on the screen. **For example:** If the statement:

```
PUBLIC static void main(String [] args)
```

is written instead of the statement:

```
public static void main(String [] args)
```

Compilation Error will be displayed on the console screen.

Each **code statement** must end with a **semicolon**. If we forget to end each **program statement** within the body of **main method** with a **semicolon " ; "** – Error will be displayed on the screen.

Marks the end of the statement

In the statement:

```
System.out.println("Hello World!");
```

- **System** → name of a standard class that contains variables and methods for supporting **simple keyboard and character output** to the display.
- **Dot Operator ". "** → provides access to the **out** static field within the **System** class
- **out** → represents the standard output stream. It is the name of the static field within the **System** class containing the standard output functionality.

- **Dot Operator ". "** → provides access to the **println method** within the **out** static field
- **println("Hello World!")** → output method of the **Java language** which makes provision to print the String literal "Hello World!" that is passed as a parameter into the println method.

The double quotation marks on each end of the text **Hello World!** delimit the text as a String

Semicolon marks the end of the println method

If the word "**public**" in the statement:

```
public static void main(String [] args)
```

is replaced by the word

private or

protected

Then **compilation error** will be flagged on the screen because if the method is declared **private** or **protected** then this method does not make itself available to **JVM** for execution.

- **main** → implies the name of the method
- **static** means the **main method** is the part of the class **MyClass**

Why static?

Because the program execution begins from the **main method** and if the **main method** is not declared static then the execution of the program does not take place.

- **void** → implies the **main method** does not return any value i.e., **main method** return nothing when it completes execution.

- **String [] args** → While running the program if we want to pass something to the **main method**, then this parameter is used as the way of taking input from the user – so we can pass some strings while running the program if we want. Moreover, **JVM** cannot recognize the method:

```
public static void main(String [] args)
```

as method if the parameter **String [] args** is not included.

If the word **"args"** in the statement:

```
public static void main(String [] args)
```

is replaced by another word say **"jamesgosling"** or **"java"**

```
public static void main(String [] jamesgosling)

                or

public static void main(String [] java)
```

No error will be displayed on the screen

Most Java programmers prefer **args** and **argv** i.e., the statements:

```
• public static void main(String [] args)
• public static void main(String [] argv)
```

are preferred.

Java Modifiers

- **Access Modifiers** – default, public, protected, private
- **Non-access Modifiers** – final, abstract, strictfp

Java Variables

- Local Variables
- Class Variables (**Static Variables**)
- Instance Variables (**Non-static Variables**)

NoOps:

The idea that automation of deployment, monitoring and management of applications can completely eliminate the need for an operations team to manage the software environment

Non-Functional Testing: The process of verifying the way software application works — and how well it works by examining things like performance, usability, reliability and accountability of the software application

Shift Left: The practice of testing the software application in the early stages of software development process − allowing developers to identify and resolve bugs before they become critical

```
public class MyClass {
public static void main(String... args) {
      System.out.println("Hello World!"); // Output: Hello World!
 }
}
```

```
import java.util.*; // import java.util package
public class MyClass {
public static void main(String[] args) {
Integer[] x = {11, 12, 13}; // object array of Integer
System.out.println(Arrays.asList(x).contains(11));
 // Output: true

System.out.println(Arrays.asList(11,12,13).contains(11));
 // Output: true

}
}
```

```
public class MyClass {
public static void main(String[] args) {
int[] array = new int[5];
for(int i = 0; i < array.length; i++) {
System.out.println("Hello World!");
}
}
}
```

Output:

```
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
```

```
public class MyClass {
public static void main(String[] args) {
// creating an array with integers
int[] array = {0, 1, 2, 3, 4, 5, 6, 7};
for(int x = array.length - 1; x >= 0; x--) {
System.out.println(array[x]);
}
}
}
```

Output:

```
7
6
5
4
3
2
1
0
```

```
import java.util.*;
public class MyClass {
public static void main(String[] args) {
int[] x = new int[]{11, 12, 13};
int[] y = new int[]{11, 12, 13};
// prints "false" because 'x' and 'y' refer to different objects
System.out.println(x.equals(y));
// prints "true" because the elements of objects 'x' and 'y' have the same values
System.out.println(Arrays.equals(x, y));
}
}
```

```
public class MyClass {
public static void main(String[] args) {
String x = System.console().readLine("Input your name and then hit Enter:%n");
System.out.printf("Hi, %s!", x);
}
}

// Output:
Input your name and then hit Enter:
John # entered name
Hi, John!
```

```
import java.util.*;
public class MyClass {
public static void main(String args[]) {
 // create and populate hash table
 Hashtable<Integer, String> map = new Hashtable<Integer, String>();
 map.put(88,"Albert");
 map.put(89, "John");
 map.put(90, "Mary");
 System.out.println("Before removing values: \n"+ map);
 // remove value for key 89
 map.remove(89);
 System.out.println("After removing values: \n"+ map);
}
}

// Output:
Before removing values:
{90=Mary, 89=John, 88=Albert}
After removing values:
{90=Mary, 88=Albert}
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println("\\".matches("\\\\"));
// prints true
}
}
```

```
public class MyClass {
public static void main(String[] args) {
int i = 0;
i = i++ + 1 + i++; // i = 0 + 1 + 1
System.out.println(i);
// Output: 2
}
}
```

```
public class MyClass {
public static void main(String[] args) {
int i = 0; // initializes i
i = i++ + 1 + i; // i = 0 + 1 + 1
i++; // adds 1
System.out.println(i);
// Output: 3
}
}
```

```
import java.util.regex.Pattern;
public class MyClass {
public static void main(String[] args) {
System.out.println("[1988".matches(Pattern.quote("[") + "1988"));
// Output: true
System.out.println("[1985".matches(Pattern.quote("[") + "1988"));
// Output: false
}
}
```

```
import java.util.Arrays;
public class MyClass {
public static void main(String[] args) {
// create an array with integers
int[] array = {27, 34, 12, 21, 39};
// sort the array
Arrays.sort(array);
System.out.println(Arrays.toString(array));
// prints [12, 21, 27, 34, 39]
}
}
```

```
x %= 2;
```

The %= obtains the **remainder** of x/2 and puts that result back into x

```
public class MyClass {
public static void main(String[] args) {
System.out.println("ALBERT".equalsIgnoreCase("albert"));
// prints true
}
}
```

```
public class MyClass {
public static void main(String[] args) {
String x = "apple";
System.out.println(x.replace('p','B'));
// Output: aBBle
}
}
```

```
public class MyClass {
public static void main(String[] args) {
String x = "theoretical physicist";
System.out.print(x.replace("physicist", "chemist"));
// Output: theoretical chemist
} // Both print() and println() can be used to print the output
}
```

```
public class MyClass {
public static void main(String[] args) {
String s = "rosa rubiginosa";
System.out.println(s.replaceAll("(\\w*os)","$1lic"));
// Output: roslica rubiginoslica
}
}
```

```
public class MyClass {
public static void main(String[] args) {
String x = new String("   Albert Einstein!!   ");
// trim() method removes whitespace from both ends of the string 'x'
System.out.println(x.trim());
// prints Albert Einstein!!
}
}
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println("[294".matches("[294"));
/* throws a syntax error because
   of the unclosed bracket
*/
System.out.println("[294".matches("\\Q[\\E294")); // prints true
/* enclosing the bracket in \Q and \E makes it
   possible for the pattern to match as expected
*/
}
}
```

```
public class MyClass {
public static void main(String[] args) {
// prints the maximum value of short
System.out.println(Short.MAX_VALUE);
// Output: 32767

// prints the minimum value of short
System.out.println(Short.MIN_VALUE);
// Output: -32768
}
}
```

```
public class MyClass {
public static void main(String[] args) {
// prints the maximum value of long
System.out.println(Long.MAX_VALUE);
// Output: 9223372036854775807

// prints the minimum value of long
System.out.println(Long.MIN_VALUE);
// Output: -9223372036854775808
}
}
```

```
public class MyClass {
public static void main(String[] args) {
int x = 987654;
int y = 987_654;
System.out.println(x == y); // prints true
}
}
```

```
public class MyClass {
public static void main(String[] args) {
Integer x = Integer.valueOf(2000);
Integer y = Integer.valueOf(x.intValue() + 2);
System.out.println(y);
// Output: 2002
}
}
```

```
public class MyClass {
public static void main(String[] args) {
String[] fruits = {"Apple", "Orange", "Grapes", "Mango"};
// Grapes is not printed
for(String x : fruits) {
if(x.equals("Grapes"))
continue;
System.out.println(x);
}
}
}
```

Output:

Apple
Orange
Mango

```
public class MyClass {
public static void main(String[] args) {
int sum = 0;
for(int x = 1; x <= 010; x++) {
sum = sum + x;
}
System.out.println("The sum of 1 to 10 is: " + sum);
}
}

// Output:
                          The sum of 1 to 10 is 36

1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55
              36
```

Since 010 is an octal number, which is equivalent to 8 in decimal, the program above actually prints the sum of the numbers 1 to 8.

Float Over-Flow

```
public class MyClass {
public static void main(String[] args) {

float x = 3.4e38f;
System.out.println((float) x*3); // Output: Infinity

/*
  maximum possible value of float is 3.4028235e+38
  since (float) x*3 exceeds this value the program outputs Infinity
*/

}
}
```

Float Under-Flow

```
public class MyClass {
public static void main(String[] args) {
float x = 1e-45f;
System.out.println((float) x/1000); // Output: 0.0

/*
  minimum possible value of float is 1.4e-45f
  since (float) x/1000 goes below this value the program outputs 0.0
*/

}
}
```

Double Over-Flow

```
public class MyClass {
public static void main(String[] args) {
double x = 1e308;
System.out.println((double) x*3); // Output: Infinity

/*
  maximum possible value of double is 1.7976931348623157e+308
  since (double) x*3 exceeds this value the program outputs Infinity
*/

}
}
```

Double Under-Flow

```
public class MyClass {
public static void main(String[] args) {
double x = 4.8e-323;
System.out.println((double) x/1000); // Output: 0.0

/*
  minimum possible value of double is 4.9e-324
  since (double) x/1000 goes below this value the program outputs 0.0
*/

}
}
```

```
public class MyClass {

public enum Names {
   Albert, John, Mary, James, David;
}

public static void main(String[] args) {

 // convert enum to a String
// Names.Albert to "Albert"
// returns the String representation of the enum

System.out.println(Names.Albert.name()); // Output: Albert
System.out.println(Names.Albert.toString()); // Output: Albert
}
}
```

```
public class MyClass {
public static void main(String[] args) {
StringBuilder x = new StringBuilder("Alan");
System.out.println(x.reverse().toString());
// Output: nalA
}
}
```

StringBuilder:

A class in **Java** used to assemble and modify strings

```
public class MyClass {
public static void main(String[] args) {
StringBuilder x = new StringBuilder("19");
System.out.println(x.append("0").append(5).toString());
// Output: 1905
}
}
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println("Albert ".concat("Einstein"));
// Output: Albert Einstein
}
}
```

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("66.77");
BigDecimal y = new BigDecimal("20.023");
System.out.println(x.add(y));
// Output: 86.793
}
}
```

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal(6);
System.out.println(x.compareTo(new BigDecimal(1)));
// x >1, prints 1
System.out.println(x.compareTo(new BigDecimal(6)));
// x = 6, prints 0
System.out.println(x.compareTo(new BigDecimal(20)));
// x < 20, prints -1
}
}
```

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("5");
System.out.println(x.pow(2)); // prints x*x
// Output: 25
}
}
```

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("37");
BigDecimal y = new BigDecimal("16");
System.out.println(x.subtract(y)); // prints x - y
// Output: 21
}
}
```

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("37");
BigDecimal y = new BigDecimal("16");
System.out.println(x.multiply(y)); // prints x * y
// Output: 592
}
}
```

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("37");
BigDecimal y = new BigDecimal("16");
System.out.println(x.divide(y)); // prints (x / y)
// Output: 2.3125
}
}
```

```
import java.math.BigDecimal;
public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("37");
BigDecimal y = new BigDecimal("16");
System.out.println(x.remainder(y)); // prints x % y
// Output: 5
}
}
```

```
import java.math.BigDecimal;
public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("37");
BigDecimal y = new BigDecimal("16");
System.out.println(x.max(y)); // prints the maximum number
// Output: 37

System.out.println(x.min(y)); // prints the minimum number
// Output: 16
}
}
```

- **Program 1.1**

  Java program to print the word "Hello Bill Gates" on screen

```
public class MyClass {
// main method begins execution of Java application
public static void main (String [] args) {
System.out.println("Hello Bill Gates");
} // end method main
} // end class MyClass
```

The output on the screen:

```
Hello Bill Gates
```

Through the use of inheritance, encapsulation, and polymorphism, the **Object Oriented Programming** methodology helps in organizing complex programs

- **Program 1.2**

  Java program to print the word " `****Hello silicon city****` " on screen

```
public class MyClass {
public static void main(String [] args) {
System.out.println(" ****Hello silicon city**** ");
}
}
```

The output on the screen:

```
****Hello silicon city****
```

```
// This is a one-line comment

/**
This is a
Multi-line
Comment */

/* This is a
Multi-line
  Comment */
```

Java comments

- **Program 1.3**

  Java program to print

*

*****

*****

*****

*****

on screen

```
public class MyClass {
public static void main(String [] args) {
System.out.println("\n    *   ");
System.out.println("\n  ***** ");
System.out.println("\n  ***** ");
System.out.println("\n  ***** ");
System.out.println("\n  ***** ");
}
}
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println(2.04 - .52);
// Output: 1.52
System.out.println(2.00 - 8 * .20);
// Output: 0.3999999999999999
}
}
```

The output on the screen:

*

*****

*****

*****

*****

If new line `\n` is not included in the above program then the output on the screen is:

****

- **Write a program to print the following outputs:**

(a)

```
*

****

**java**

****

*
```

(b)

Python code:

```
x = "Albert"
print(isinstance(x, str))
# Output: True

x = 1
print(type(x) == int)
# Output: True

x = 1.6
print(type(x) == float)
# Output: True
```

```
**********

* *

* Hello World! *

* *

**********
```

(c)

```
public class MyClass {
public static void main(String args[]) {
Double x = new Double(9.87654);
Double y = new Double("987654E-5");
System.out.println(x + " = " + y + " -> " + x.equals(y));
// Output: 9.87654 = 9.87654 -> true
}
}
```

```
Braces come in pairs!

Comments come in pairs!

All statements end with a semicolon!

Spaces are optional!

Must have a main method!

Java is done mostly in lowercase. Like C & C++ it's also a case-sensitive language
```

Answers:

a)

```
public class MyClass {
public static void main (String [] args) {
System.out.println("\n       *     ");
System.out.println("\n      ****   ");
System.out.println("\n   **java** ");
System.out.println("\n      ****    ");
System.out.println("\n       *      ");
}
}
```

Exception handler:

A code that specifies what a programme should do if something unusual happens to interrupt the normal flow of its operations

b)

```
public class MyClass {
public static void main (String [] args) {
System.out.println("\n          ***************         ");
System.out.println("\n               * *                ");
System.out.println("\n           * Hello World! *        ");
System.out.println("\n               * *                ");
System.out.println("\n         ***************          ");
}
}
```

c)

```
public class MyClass {
public static void main (String [] args) {
System.out.println("\n Braces come in pairs!");
System.out.println("\n Comments come in pairs!");
System.out.println("\n All statements end with a semicolon!");
System.out.println("\n  Spaces are optional!");
```

```
System.out.println("\n Must have a main method!");
System.out.println("\n Java is done mostly in lowercase. Like C & C++ it's also a case-sensitive language");
}
}
```

First Customer Ship:

The day that an item is made available to or shipped to the client

- **Program 1.4**

  Java program to find the area of the circle

```
public class MyClass {
public static void main (String [] args) {
double r, area;
r = 2.5;
area = 3.14 * r * r;
System.out.println("The area of the circle = " + area);
}
}
```

The output on the screen:

```
The area of the circle = 19.625
```

In `C language`, the statement:

```
printf("The area of the circle = %f ",  area);
```

make the provision to print the output on the screen.

HotJava(TM) Browser:

A Java(TM) programming language-based Web browser created by Sun Microsystems that is highly customizable

In `C++ language`, the statement:

```
cout<<"The area of the circle = "<< area;
```

make the provision to print the output on the screen.

whereas in the Java language, the statement:

```
System.out.println("The area of the circle =  " + area);
```

make the provision to print the output on the screen.

In the statement:

```
System.out.println("The area of the circle =  " + area);
```

There are two strings:

- The area of the circle =
- area

Plus operator (+) functions as the **concatenation operator** (concatenation means connecting two statements to produce a single statement) – which (here) concatenates the string:

```
"The area of the circle =  "
```

and the string:

```
area (which is 3.14 * r * r = 19.625)
```

```
public class MyClass {
public static void main(String[] args) {
int x = 5;
if(x > 0) {
String i = "The value of x is: " + x;
System.out.print(i);
}
}
}
```

Output:

```
The value of x is: 5
```

producing a String statement:

```
The area of the circle = 19.625
```

which will be displayed on the screen as the result.

Even though if we write `ARGS` instead of `args` i.e., even though if we express `args` in capital letter, **No error** will be displayed on the screen.

```
public static void main(String [] ARGS) ⟶ no error will be displayed on the console screen
```

Operator	Name	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value by another	x / y
%	Modulus	Returns the division remainder	x % y
++	Increment	Increases the value of a variable by 1	x++
--	Decrement	Decreases the value of a variable by 1	x--

- **Program 1.5**

  Java program to find the circumference of the circle

```
public class MyClass {
public static void main (String [] args) {
double r, circumference;
r = 2.5;
circumference = 2 * 3.14 * r;
System.out.println("The circumference of the circle = " + circumference);
}
}
```

The output on the screen is:

```
The circumference of the circle = 15.700000000000001
```

What will be the output of the following programs?

a)

JAR (Java Archive):

A platform-independent file format that combines multiple files into one

```
public class MyClass {
public static void main (String [] args) {
double l, b, area;
l=2.9;
b=2.5;
area = l*b;
System.out.println("The area of the rectangle = " + area);
}
}
```

Answer:

```
The area of the rectangle = 7.25
```

b)

Java Blend:

A tool that makes it easier for programmers to create database applications by mapping database records to Java objects and Java objects to databases

```
public class MyClass {
public static void main (String [] args) {
int a, b, c;
a= 3;
b=3;
c=3;
if ((a + b< c) || (b + c < a) || (a==b && b==c))
System.out.println(" The triangle is equilateral");
else
System.out.println(" The triangle is not possible");
}
}
```

JavaBeans:

Classes that combine several objects into one standardized object

Answer:

```
The triangle is equilateral
```

- **Program 1.6**

  Java program to convert the temperature in Celsius to Fahrenheit

```
public class MyClass {
public static void main(String [] args){
double C, F;
C=38.5;
```

```
F = 9*C/5 +32;
System.out.println("Temperature in Fahrenheit= " +F);
 }
}
```

The output on the screen:

```
Temperature in Fahrenheit= 101.3
```

JavaCheck:

A tool for determining if Java applications and applets adhere to a specification

- **Program 1.7**

  Java program to find the sum of two numbers

```
public class MyClass {

public static void main(String [] args) {

int a, b, sum;
a=1;
b=2;
sum = a + b;
System.out.println("The sum of a and b = " + sum);
}
}
```

Delegation:

An approach for extending and reusing the behavior of a class

The output on the screen:

```
The sum of a and b = 3
```

If you want to supply the values **for** a and b through the **key board**, then we have to rewrite the program as follows:

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int a, b, sum;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
sum = a + b;
System.out.println("The sum of a and b = " + sum);
}
}
```

JavaChip:

A microchip that will speed up the execution of Java programs when inserted inside or added to a computer

```
public class MyClass {
public static void main(String[] args) {
String x = "x: " + 3 + 3;
System.out.println(x);
// Output: x: 33
String y = "y: " + (3 + 3);
System.out.println(y);
// Output: y: 6
}
}
```

The output on the screen:

```
Enter any two Numbers:

If you enter two numbers 2 and 3

The sum of a and b = 5

will be outputted on the screen
```

Scanner is a class found in **java.util package**. So to use Scanner class, we first needto include:

java.util package

in our program.

```
import java.util.Scanner; // This will import just the Scanner class
import java.util.*; // This will import the entire java.util package
```

The statement:

```
Scanner scan = new Scanner(System.in);
```

implies: declaring an object of the **Scanner class** "scan" to read the values entered for a and b through the key board. And the statements:

```
a = scan.nextInt();

b = scan.nextInt();
```

imply: scan is an object of **Scanner class** and nextInt() is a method of the object "**scan**" that allows the object "**scan**" to read only integer values from the keyboard.

- nextInt() that allows the object "**scan**" to read only integer values from the keyboard, **methods** that allows the object "**scan**" to read other data types from the **keyboard** are listed below:

Methods	Datatype
nextInt()	Integer
nextFloat()	Float
nextDouble()	Double
nextLong()	Long
nextShort()	Short
next()	Single word
nextLine()	Line of Strings
nextBoolean()	Boolean

- **Program 1.8**

  Java program to find the square root of a number

i)

Executable content:

A program incorporated inside an HTML file

```
public class MyClass {
public static void main(String [] args) {
int x;
x = 233;
System.out.println("Square root of a number = " + Math.sqrt(x));
}
}
```

The output on the screen:

```
Square root of a number = 15.264337522473747
```

If you want to supply the value for `'x'` through the **key board**, then the above program should take the form:

```
import java.util.Scanner;   // import class Scanner
public class MyClass {
public static void main(String [] args) {
int x;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");

x = scan.nextInt();   // read in number from user

System.out.println("Square root of a number = " + Math.sqrt(x));   // display the result
}
}
```

The output on the screen:

```
Enter any Number:

If you enter the number 233
```

Square root of a number = 15.264337522473747

will be outputted on the screen.

Java Compatibility Kit (JCK):

A test suite, a collection of tools, and other specifications used to attest for a Java platform implementation's adherence to the appropriate Java platform specifications as well as Java Software reference implementations

ii)

```
public class MyClass {
public static void main(String [] args) {
double x;
x = 19.99d;
System.out.println("Square root of a number = " + Math.sqrt(x));
}
}
```

The output on the screen:

```
Square root of a number = 4.471017781221631
```

If you want to supply the value for 'x' through the **key board**, then the above program should take the form:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
double x;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");
x = scan.nextDouble();
System.out.println("Square root of a number = " + Math.sqrt(x));
}
}
```

The output on the screen:

```
Enter any Number:
If you enter the number 19.99
Square root of a number = 4.471017781221631
will be outputted on the screen.
```

Impersonation:

A thread's ability to operate with security information different from that of the process it belongs to

- **Program 1.9**

  What will be the output of the following program?

```
public class MyClass {
public static void main(String[] args) {
char c;
c = 'A';
System.out.println("ch= " + c);
}
}
```

```
import static java.lang.Math.PI;
import static java.lang.Math.sqrt;
public class MyClass {
public static void main(String... args) {
// prints PI value
System.out.println(PI);
// prints Square root value of 6.0
System.out.println(sqrt(6.0));
 }
}
```

The output on the screen:

```
ch=A
```

If you want to supply the value for `'c'` through the **key board**, then the above program should take the form:

```
public class MyClass {

public static void main(String[] args) throws Exception {

char c;

System.out.print("Enter a character: ");
```

```
c = (char)System.in.read();

System.out.println("ch= " + c);

}

}
```

The output on the screen:

Enter a character:

If you enter the character K

ch= K

will be outputted on the screen.

```
public class MyClass {
public static void main(String[] args) {
System.out.write('A');
System.out.write('\n');
}
}
// Output: A
```

Output the character "A" followed by a newline to the console screen

Note: **Exception** is a problem that arises during the execution of a program. When an exception occurs, program abnormally terminates and disrupts – throws Exception should be written after the statement "**public static void main(String[] args)**" so that the exceptions are thrown to the **operating system** to handle and the program will be successfully executed and the output will be displayed on the screen.

- **Program 2.0**

```
import java.util.Scanner;

public class MyClass {

public static void main(String [] args) {

String m;

Scanner in = new Scanner(System.in);

System.out.print("Enter the word: ");

m = in.nextLine();

System.out.println("The word you entered: " + m);

}

}
```

The output on the screen:

Enter the word:

If you enter the word dog

The word you entered: dog

will be outputted on the screen.

```
// Construct one String from another
public class MyClass {
public static void main(String args[]) {
char i[] = {'A', 'l', 'a', 'n'};
String a = new String(i);
String b = new String(a);
System.out.println(a);
// Output: Alan
System.out.println(b);
// Output: Alan
}
}
```

If the statement:

```
m = scan.nextLine();
```

is written instead of

```
m = in.nextLine();
```

Then we have to replace the statement:

```
Scanner in = new Scanner(System.in);
```

by the statement:

```
Scanner scan = new Scanner(System.in);
```

Otherwise **compilation error** will be displayed on the console screen.

- **What is the mistake in the following program?**

```
public class MyClass {

static public void main(String args []) {
double x;
x = 233;

System.out.println("Cube root of a number = " + Math.cbrt(x));
```

```
}
}
```

Output:

```
Cube root of a number = 6.1534494936636825
```

Answer:

There is no mistake in the above program.

The statement:

```
public static void main(String[] args)
```

can also be written as:

```
static public void main(String args [])
```

```
public class MyClass {
public static void main(String args[]) {
byte i[] = {65, 66, 67, 68, 69, 70 };
String x = new String(i);
System.out.println(x);
// Output: ABCDEF
}
}
```

- **Program 2.1**

  Java program to find the product of two numbers.

```
public class MyClass {

public static void main(String [] args) {

int a, b, product;

a=1;

b=2;

product = a * b;

System.out.println("The product of a and b = " + product);

}

}
```

The output on the screen:

```
The product of a and b = 2
```

If you want to supply the values for a and b through the **key board**, then we have to rewrite the above program as follows:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int a, b, product;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
product = a * b;
System.out.println("The product of a and b = " + product);
}
}
```


The output on the screen:

```
Enter any two Numbers:

If you enter two numbers 1 and 2
The product of a and b = 2
 will be outputted on the screen
```

If you want to assign the **floating point values** for a and b, then the above program should take the form:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
float a, b, product;
Scanner scan = new Scanner(System.in);
```

```
System.out.print("Enter any two Numbers: ");
a = scan.nextFloat();
b = scan.nextFloat();
product = a * b;
System.out.println("The product of a and b = " + product);
}
}
```

Java Enterprise API:

A simple-to-use API that enables the development of large-scale database and commercial applications that can exchange multimedia data with other applications both inside and outside of an enterprise organization via internet

The output on the screen:

```
Enter any two Numbers:
If you enter two floating point values 2.9 and 3.6
The product of a and b = 10.440001
will be outputted on the screen.
```

Javadoc Tool:

A program that converts source code documentation comments into **HTML API documentation**

If the **statement**:

```
System.out.println("The product of a and b = " + product);
```

is replaced by the **statement**:

```
System.out.println(a + " * " + b + " = " + product);
```

Then the **output on the screen** is:

```
2.9 * 3.6 = 10.440001
```

- **Note:** The word `public` in the statement:

```
public class MyClass
```

implies: that the `program` or the `data` within the program (such as `methods`, `variables` etc.) can be accessed directly by an **`external java program`**.

If replace the word **`public`** by **`private`** i.e.,

```
private class MyClass
```

is written instead of

```
public class MyClass
```

then the `program` or the **`data within the program`** (such as `methods`, `variables` etc.) cannot be accessed directly by an **`external program`**.

Java Media APIs:

A group of APIs that enable the incorporation of voice calls, 3D models, 2D typefaces, graphics, and pictures, as well as audio and video footage

If you want to insert a **`10 digit number`** for `a` and `b` i.e.,

`a=1000000000`

`b=3000000000`, then the statement:

```
int a, b, product;
```

should be replaced by the statement:

```
long a, b, product;
```

i.e.,

```
public class MyClass {
```

Optional packages:

The collection or groups of **Java platform edition APIs** that can be used with and perhaps supported by a compatible implementation. Optional packages may eventually become necessary in a given edition when the market demands them.

```
public static void main(String [] args) {
long a, b, product;
a=1000000000;
b=2000000000;
product = a * b;
System.out.println("The product of a and b = " + product);
}
}
```

The output on the screen:

```
The product of a and b = 2000000000000000000
```

- **What will be the output of the following program?**

```
public class MyClass {
static public void main(String args []) {
int x;
x = 2;
System.out.println("Square of " +x+ " is: " + Math.pow((x), 2));
}
}
```

Answer:

```
Square of 2 is: 4.0
```

- **Program 2.2**

  Java program to find the square of a number

```
public class MyClass {
public static void main(String [] args) {
int a, b;
```

Dummy parameter:

A parameter that is included in a **command** or **instruction** even when its value is insignificant because it is necessary to meet the system's requirements

```
a=2;
b = a * a;
System.out.println("The square of a = " + b);
}
}
```

The output on the screen:

```
The square of a = 4
```

If you want to supply the value for a through the **key board**, then we have to rewrite the above program as follows:

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int a, b;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any Number: ");
a = scan.nextInt();
b = a * a;
System.out.println("The square of a = " + b);
}
}
```

```
import static java.lang.Math.*;

public class MyClass {
public static void main(String... args) {
// prints the absolute value of -5.8

System.out.println(abs(-5.8));

// prints the lowest value
System.out.println(min(15, 20));

// prints the highest value
System.out.println(max(15, 20));
 }
}
```

The output on the screen:

```
Enter any number:

If you enter a number 3

The square of a = 9 will be outputted on the screen.
```

- **Note:**

- If `scan.nextint()` is written instead of **`scan.nextInt()`**
- `public static void main(string [] args);` is written instead of

```
public static void main(String [] args)
```

- `system.out.println("The square of a = " + b);` is written instead of

```
System.out.println("The square of a = " + b);
```

Then the **`compilation error`** will be displayed on the screen.

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("9876.54321056");
// moves the decimal point to 2 places left of current position
System.out.println(x.movePointLeft(2));
}
}

// Output: 98.7654321056
```

- **Program 2.3**

Java program to find the greatest of two numbers using **`if - else`** statement

```
public class MyClass {
public static void main(String [] args) {
int a, b;
a=2; b
=3;
if(a>b)
```

The main method's header

```
{
System.out.println("a is greater than b");
}
else
{
System.out.println("b is greater than a");
}
}
}
```

The output on the screen:

```
b is greater than a
```

In the above program:

If the condition `(a> b)` is true, then the **statement**:

```
System.out.println("a is greater than b");
```

is executed to print the output:

```
a is greater than b
```

`else` the statement:

```
System.out.println("b is greater than a");
```

is executed to print the output:

```
b is greater than a
```

```
import java.util.*;
public class MyClass {
public static void main(String... args) {
List<String> x = new ArrayList<String>();
x.add("Albert");
x.add("John");
x.add("Mary");
for(String i : x) {
System.out.println(i);
}
}
}
```

Output:

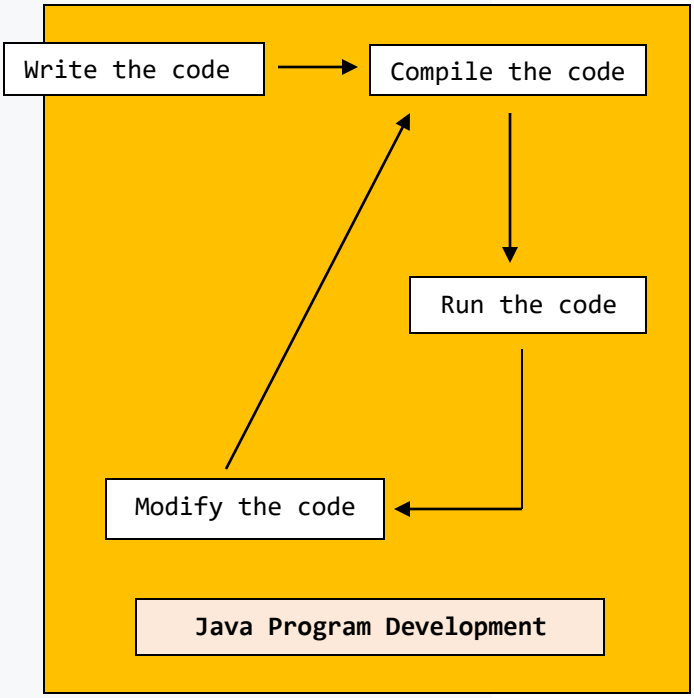
```
Albert
John
Mary
```

Hash table:

A data structure designed for effective object search, insertion, and deletion

If you want to supply the values for a and b through the **key board**, then the above program should be rewritten as:

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args){
int a, b;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
if(a>b) {
System.out.println("a is greater than b");
}
else {
System.out.println("b is greater than a");
}
}
}
```



The output on the screen:

```
Enter any two Numbers:
If you enter two numbers 2 and 3
b is greater than a
will be outputted on the screen.
```

```
public class MyClass {
static String x = "Elsa Einstein";
public static void main(String[] args) {
String x = "Albert Einstein";
System.out.println(x);
// Output: Albert Einstein
}
}
```

```
public class MyClass {
public static void main(String[] args) {
try {
System.out.print("Albert ");
} finally {
System.out.print("Einstein");
}
}
}
```

Output:

Albert Einstein

```
public class MyClass {
public static void main(String[] args) {
try {
System.out.print("Albert ");
System.exit(0);
} finally {
System.out.print("Einstein");
}
}
}
```

Output:

Albert

Ceases Java Virtual Machine's execution

```
import java.math.BigInteger;

public class MyClass {
public static void main(String[] args) {
BigInteger x= new BigInteger("5");
BigInteger y = new BigInteger("6");
BigInteger z = new BigInteger("7");
BigInteger c = BigInteger.ZERO;
c = c.add(x);
c = c.add(y);
c = c.add(z);
System.out.println(c);
// Output: 18 (5 + 6 + 7)
}
}
```

- **Program 2.4**

  Java program to find the greatest of three numbers using **`else if`** statement

```
public class MyClass {

public static void main(String [] args){
int a, b, c;
a=2;
b =3;
c=4;
if(a>b&&a>c) {

System.out.println("a is greater than b and c");

}

else if(b>a&&b>c) {

System.out.println("b is greater than a and c");

}

else {

System.out.println("c is greater than b and a");

}

}

}
```

The output on the screen:

```
c is greater than b and a
```

If the statements:

```
if(a>b&&a>c) {

System.out.println("a is greater than b and c");

}

else if(b>a&&b>c) {

System.out.println("b is greater than a and c");

}

else {

System.out.println("c is greater than b and a");

}
```

are replaced by the statements:

```
if(a>b&&a>c)
{
System.out.println(a + "is greater than" + b  + "and" + c);
}
else if(b>a&&b>c)
{
System.out.println(b + "is greater than" + a  + "and" + c);
}
else
{
System.out.println(c + "is greater than" + b  + "and" + a);
}
```

Then the output on the screen is:

```
4 is greater than 3 and 2
```

```
import java.math.BigDecimal;

public class MyClass {
public static void main(String[] args) {
BigDecimal x = new BigDecimal("9876.54321056");
// moves the decimal point to 3 places right of current position
System.out.println(x.movePointRight(3));
}
}

// Output: 9876543.21056
```

- **Precedence:** The sequence in which operations are evaluated
- **Scaffolding:** Code that is utilized while creating a software but isn't included in the final version

- **Program 2.5**

  Java program to find the average of 10 numbers

JavaPlan:

A Java programming language-based object-oriented design and diagramming tool

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, X;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any ten Numbers: ");

N1 = scan.nextInt();
N2  =  scan.nextInt();
N3  =  scan.nextInt();
N4  =  scan.nextInt();
N5  =  scan.nextInt();
N6  =  scan.nextInt();
N7  =  scan.nextInt();
N8  =  scan.nextInt();
N9  =  scan.nextInt();
N10 = scan.nextInt();
X = (N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9 + N10) /10;
System.out.println("The average of 10 numbers = " + X);
}
}
```

```
public class MyClass {
public static void main(String args[]) {
StringBuffer x = new
StringBuffer("Hilbert");
System.out.println(x.deleteCharAt(0));
// Output: ilbert
}
}
```

JavaOS:

An operating system based on Java technology that is designed to function on a variety of computing and consumer environments

The output on the screen:

```
Enter any ten Numbers:
If you enter ten numbers 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10
The average of 10 numbers = 5
will be outputted on the screen.
```

- **Note:** The average of **10** numbers is **5.5**, the output on the screen is **5** because **int** is used instead of **float**.

- **Program 2.6**

  Java program to find the simple interest

```
public class MyClass {
public static void main(String [] args) {
int P,T, R, SI;          // declaration statement

P = 1000;
T = 2;                   assignment statements
R = 3;
SI = P*T*R/100;
System.out.println("The simple interest = " + SI);
}
}
```

The output on the screen:

```
The simple interest = 60
```

If you want to supply the values for P, T and R through the **key board**, then the above program should take the form:

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int P,T, R, SI;
Scanner scan = new Scanner(System.in);
System.out.println("Enter principal amount: ");
P = scan.nextInt();
System.out.println("Enter time: ");
T = scan.nextInt();
System.out.println("Enter rate of interest: ");
R = scan.nextInt();
```

Linked data structure:

A group of data made up of several objects connected by pointers kept in the objects' instance variables

```
SI = P*T*R/100;

System.out.println("The simple interest = " + SI);

}

}
```

The output on the screen:

Enter principal amount:

If you enter the principal amount 1000

Enter time:

If you enter the time 2

Enter rate of interest:

If you enter the rate of interest 3

The simple interest = 60

will be outputted on the screen.

```
public class MyClass {
public static void main(String[] args) {
int x = 62; { // start of block
int y = 58;
System.out.println("x: " + x);
System.out.println("y: " + y);
} // end of block
}
}
```

Output:

```
x: 62
y: 58
```

- **Program 2.7**

  Java program to find the senior citizen

```
public class MyClass {
public static void main(String [] args) {
int age;
age=20;
if(age >= 60) {
System.out.println("Senior citizen");
}
else {
System.out.println("Not a senior citizen");
}
}
```

```
public class MyClass {
public static void main(String args[]) {
StringBuffer x = new StringBuffer("Alan Turing");
System.out.println(x.insert(4, " Mathison"));
// Output: Alan Mathison Turing
}
}
```

The output on the screen:

```
Not a senior citizen
```

- (age> = 60) implies **age** greater than or equal to **60**

If you want to supply the value for age through the **key board**, then the above program should be rewritten as:

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int age;
Scanner scan = new Scanner(System.in);
System.out.println("Enter the age: ");
age = scan.nextInt();
if(age >= 60)
{
System.out.println("Senior citizen");
}
else
{
System.out.println("Not a senior citizen");
}
}
}
```

The output on the screen:

```
Enter the age:

If you enter the age 60
```

Output:

```
Length of x is: 20
Length of y is: 4
```

```
public class MyClass {
public static void main(String args[]) {
int x[] = new int[20];
int y[] = {14, 13, 12, -1};
System.out.println("Length of x is: " + x.length);
System.out.println("Length of y is: " + y.length);
}
}
```

```
Senior citizen

will be outputted on the screen.

Suppose if you enter the age 28

Not a senior citizen

will be outputted on the screen.
```

```
public class MyClass {
public static void main(String[] args) {
Double a, b, c;
a = 15.0;
b = 12.0;
c = Math.sqrt(a*a + b*b);
// prints the hypotenuse of a right triangle
System.out.println("Hypotenuse is: " + c);
}
} // Output: Hypotenuse is: 19.209372712298546
```

- **Program 2.8**

  Java program to get marks for 3 subjects and declare the result:

  If the marks >= 35 in all the subjects the student passes else fails.

```
public class MyClass {

public static void main(String [] args) {
int M1, M2,M3;
M1 = 38;

M2= 45;

M3 = 67;

if(M1 >= 35 && M2 >= 35 && M3 >= 35)

{

System.out.println("candidate is passed");

}

else

{

System.out.println("candidate is failed");

}

}

}
```


The output on the screen:

```
candidate is passed
```

If you want to supply the values for marks M1, M2 and M3 through the **key board**, then theabove program should be rewritten as:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int age;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any three Numbers: ");
M1= scan.nextInt();
M2 = scan.nextInt();
M3 = scan.nextInt();
if(M1 >= 35 && M2 >= 35 && M3 >= 35)
{
System.out.println("candidate is passed");
}
else
{
System.out.println("candidate is failed");
}
}
}
```

Python code:

```
x = True
print(type(x) == bool)
# Output: True

print(isinstance(x, bool))
# Output: True

print(round(0.5435, 2))
# Output: 0.54
```

The output on the screen:

```
Enter any three Numbers:

If you enter three numbers 26, 28, 39

candidate is failed

will be outputted on the screen.
```

```
public class MyClass {
public static void main(String[] args) {
int x = 1;
int y = ++x;
System.out.println("y: " + y + ", x: " + x);
// Output: y: 2, x: 2
}
}
```

```
public class MyClass {
public static void main(String[] args) {
int x = 1;
int y = x++;
System.out.println("y: " + y + ", x: " + x);
// Output: y: 1, x: 2
}
}
```

- **Program 2.9**

  Java program to find profit or loss

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int CP, SP, loss, profit;
Scanner scan = new Scanner(System.in);
System.out.println("Enter cost  price:  ");
CP = scan.nextInt();
System.out.println("Enter selling price: ");
SP = scan.nextInt();
if(SP>CP)

{

System.out.println("profit= " + (SP-CP));

}

else

{

System.out.println("loss ="  +(CP-SP));

}

}

}
```

JavaSpaces:

A Java programming language technology that offers distributed durability and data communication protocols

The output on the screen:

```
Enter cost price:

If you enter the cost price 25

Enter selling price:

If you enter the selling price 26

profit = 1

will be outputted on the screen.
```

```
import java.util.*;

public class MyClass {
public static void main(String args[]) {
Formatter x = new Formatter();
x.format("|%f|%n|%12f|%n|%012f|",
15.98765, 15.98765, 15.98765);
System.out.println(x);
}
}
```

Output:

```
|15.987650|
|   15.987650|
|00015.987650|
```

- **Program 3.0**

  Java program to find the incremented and decremented values of two numbers

```
public class MyClass {
public static void main(String [] args) {
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;

System.out.print("The incremented value of a = "+ c);
System.out.print("The incremented value of b = "+ d);
System.out.print("The decremented value of a = "+ e);
System.out.print("The decremented value of b = "+ f);
}
}
```

JMAPI

(Java Management API)

A set of Java classes and interfaces that enable programmers to create system, network, and service management applications

The output on the screen:

```
The incremented value of a = 11 The incremented value of b = 13 The decremented value
                        of a = 9 The decremented value of b = 11
```

If the statements:

```
System.out.print("The incremented value of a = "+ c);
System.out.print("The incremented value of b = " + d);
System.out.print("The decremented value of a = " + e);
System.out.print("The decremented value of b = " + f);
```

are replaced by the statements:

Python code:

```
product = lambda x, y : x * y
print(product(6, 6))
# Output: 36
```

```
print("Al\"an")
# Output: Al"an
```

```
System.out.print("\n The incremented value of a = " + c);

System.out.print("\n The incremented value of b = " + d);

System.out.print("\n The decremented value of a = " + e);

System.out.print("\n The decremented value of b = " + f);
```

Then the **output on the screen** is:

```
The incremented value of a = 11

The incremented value of b = 13

The decremented value of a = 9

The decremented value of b = 11
```

i.e., `\n` make provision for the another result to print in the `new line`. If you want to supply the values for `a` and `b` through the **`key board`**, then the above program should take the form:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int a, b, c, d, e, f;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any Number: ");
a = scan.nextInt();
System.out.println("Enter any Number: ");
b = scan.nextInt();
c=a+1;
d=b+1;
e=a-1;
f=b-1;
System.out.print("\n The incremented value of a = " + c);

System.out.print("\n The incremented value of b = " + d);

System.out.print("\n The decremented value of a = " + e);

System.out.print("\n The decremented value of b = " + f);
}

}
```

Mosaic:

A tool that offers a simple Graphical User Interface to make it simple to access data stored online

The output on the screen:

```
Enter any Number:
If you enter the value 2
Enter any Number:
If you enter the value 3

The incremented value of a = 3
The incremented value of b = 4
The decremented value of a = 1
The decremented value of b = 2

will be outputted on the screen.
```

```
public class MyClass {
public static void main(String... args) {
int[] x = {21, 6, 3, 4, 7};
for(int i = 0; i < x.length; i++) {
if(x[i] == 4) {
System.out.println("Fount 4 at index: " + i);
// Output: Fount 4 at index: 3
}
}
}
}
```

- **What will be the output of the following programs?**

A)

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
float T1, T2, A;
Scanner scan = new Scanner(System.in);
System.out.println("Enter the temperature in degree Celcius: ");
T1 = scan.nextFloat();
System.out.println("Enter the temperature in degree Celcius: ");
T2 = scan.nextFloat();
A = (T1 + T2) / 2;
System.out.println("The average temperature of the day = " + A + " degree Celcius");
}
}
```

Answer:

```
Enter the temperature in degree Celcius:
26.3 # entered temperature
Enter the temperature in degree Celcius:
27.9 # entered temperature
The average temperature of the day = 27.099998 degree Celcius
```

B)

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int P;
Scanner scan = new Scanner(System.in);
System.out.println("Enter the percentage: ");
P = scan.nextInt();
if(P >= 60) {
System.out.println("First class");
}
else if(P>=50 && P<60) {
System.out.println("Second class");
}
else {
if(P<40) System.out.println("Fail");
else System.out.println("Pass class");
}
}
}
```

Secure Socket Layer (SSL):

A technology that enables encrypted communication between a Web browser and a server

Answer:

```
Enter the percentage:

45 # entered percentage

Pass class
```

Architecture-neutral

↓

Software that is created without taking the target platform into consideration

- **Program 3.1**

  Java program to calculate the discounted price and the total price after discount

  Given:

  - If purchase value is greater than 1000, 10% discount
  - If purchase value is greater than 5000, 20% discount
  - If purchase value is greater than 10000, 30% discount

- **Discounted price**

Mutual exclusion

↓

Prohibits two threads from using the same resource simultaneously

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int PV, dis;
Scanner scan = new Scanner(System.in);
System.out.println("Enter purchased value: ");
PV = scan.nextInt();
if(PV<1000) {
```

```
System.out.println("dis = " + PV* 0.1);
}
else if(PV>5000) {
System.out.println("dis = " + PV* 0.2);
}
else {
System.out.println("dis = " + PV* 0.3);
}
}
}
```

Boxing:

The method of encapsulating a value within an object

The output on the screen:

```
Enter purchased value:
If you enter the purchased value 6500
dis = 1300
will be outputted on the screen.
```

```
import java.util.*;
public class MyClass {
public static void main(String args[]) {
Formatter x = new Formatter();
x.format("Albert %s %d %s", "Einstein", 1915, "Papers");
System.out.println(x);
// Output: Albert Einstein 1915 Papers
}
}
```

- **Total price**

```
import java.util.Scanner;

public class MyClass {

public static void main(String [] args) {
int PV, total;
Scanner scan = new Scanner(System.in);
System.out.println("Enter purchased value: ");
PV = scan.nextInt();
if(PV<1000) {
System.out.println("total= " +  PV - PV* 0.1);
}
else if(PV>5000) {
```

```
System.out.println("total = " + PV- PV* 0.2);
}
else {
System.out.println("total= " + PV- PV* 0.3);
}
}
}
```

The output on the screen:

```
Enter purchased value:

If you enter the purchased value 650

total = 585

will be outputted on the screen.
```

```
public class MyClass {
public static void main(String[] args) {
Integer x = new Integer(125);
int i = x.intValue();
System.out.println(x); // displays 125
System.out.println(i); // displays 125
}
}
```

- **Combing both the programs (above), we can write:**

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int PV, dis, total;
Scanner scan = new Scanner(System.in);
System.out.println("Enter purchased value: ");
PV = scan.nextInt();

if(PV<1000) {
System.out.println("dis = " + PV* 0.1);
System.out.println("total= " + total - dis);
}
else if(PV>5000) {
System.out.println("dis = " + PV* 0.2);
System.out.println("total= " + total - dis);
```

World readable files:

Files on a file system – accessible to all users. For example, files on web servers can only be viewed by **Internet users** if their permissions are set to world readable

```
}
else {
System.out.println("dis = " + PV* 0.3);
System.out.println("total= " + total - dis);
}
}
}
```

The output on the screen:

```
Enter purchased value:
If you enter the purchased value 850
dis = 85
total = 765

will be outputted on the screen.
```

```
public class MyClass {
public static void main(String[] args) {
int x[] = {10, 13, 6, 4, 2};
for(int i = -1; i++ < x.length -1;) {
System.out.println("x[" + i + "] = " + x[i]);
}
}
}
```

Output:

```
x[0] = 10
x[1] = 13
x[2] = 6
x[3] = 4
x[4] = 2
```

- **Program 3.2**

  Java program to print the **first ten natural numbers** using **for loop** statement

```
public class MyClass {
public static void main(String [] args) {
int i;
for(i=1; i<=10; i++)
System.out.println("value of i = " + i);
}
}
```

The output on the screen is:

```
value of i = 1 value of i = 2 value of i= 3  value of i= 4  value of i= 5  value of
i= 6 value of i = 7 value of i= 8  value of i = 9  value of i = 10
```

If the **statement**:

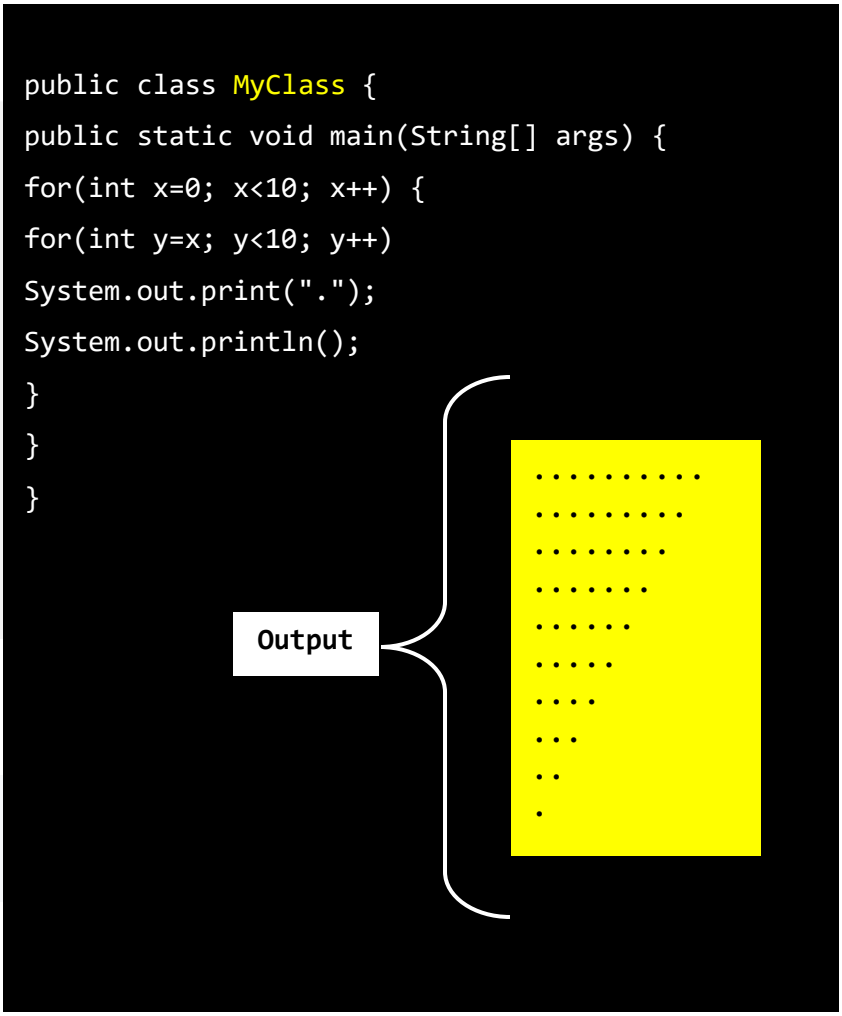
```
System.out.println("value of i = " + i);
```

is replaced by the **statement**:

```
System.out.println("\n value of i = " + i);
```

Then the **output on the screen** is:

```
value of i = 1
value of i = 2
value of i = 3
value of i = 4
value of i = 5
value of i = 6
value of i = 7
value of i = 8
value of i = 9
value of i = 10
```



If the **`for loop`** statement:

```
for(i=2; i<=10; i++)
```

is written instead of the statement:

`for(i=1; i<=10; i++)`, then the **`output on the screen`** is:

```
value of i = 2 value of i = 3 value of i= 4 value of i= 5 value of i= 6 value
of i = 7 value of i= 8 value of i = 9 value of i= 10
```

If the **`for loop`** statement:

```
for(i=1; i<10; i++)
```

is written instead of the statement: `for(i=1; i<=10; i++)`, then the **`output on the screen`** is:

```
value of i = 1 value of i = 2 value of i= 3  value of i= 4  value of i= 5  value of
i= 6 value of i = 7 value of i= 8  value of i = 9
```

- **Note:** the condition `i<=10` tells to print till value of `i=10` but the condition `i<10` tells to print till value of `i=9`

If the statement:

```
for(i=1; i=10; i++)
```

is written instead of the statement: `for(i=1; i<=10; i++)`, then the **`output on the screen`** is:

```
MyClass.java:12: error: incompatible types: int cannot be converted to boolean
for(i=1; i=10; i++)
          ^
       1 error
```

Variable-length argument:

A feature that permits a function to take any number of arguments

If the statement:

`System.out.println("\n value of i = " + i);` is replaced by the statement

```
System.out.println("\n " + i);
```

Then the **output on the screen** is:

```
1
2
3
4
5
6
7
8
9
10
```

```
public class MyClass {
public static void main(String[] args) {
int x = 24;
int y = 21;
boolean z = x < y;
System.out.println(z);
// Output: false
}
}
```

- **What is the mistake in the following program?**

```
public class MyClass {
public static void main(String []args) throws Exception {
System.out.println("Hello World");
}
}
```

Pragmatics: Guidelines for what makes quality program writing. The pragmatics of a programming language, for example, includes stylistic rules and recommendations for program organization

Answer:

There is `no mistake` in the above program. Addition of the statement `throws Exception` does not make any change in the output displayed on the screen or give rise to **any compilation error** onthe screen.

- **Program 3.3**

  What will be the output of the following program?

```
public class MyClass {
public static void main(String [] args) {
int i;
for(i =1; i<=5; i++)
System.out.println("\n Linux is not portable");
}
}
```

Answer:

```
Linux is not portable
Linux is not portable
Linux is not portable
Linux is not portable
Linux is not portable
```

Multithreaded: The ability of computer software to support numerous users simultaneously without requiring additional copies of the software to be running on the system. Multiple requests from the same user can be handled via multithreading as well

- **Java program to print the first ten natural numbers using `while loop` statement**

```
public class HelloWorld{
public static void main(String [] args)
{
int i = 1;
```

```
while (i<=10)
{
System.out.println("\n  " + i++);
}
}
}
```

The output on the screen is:

```
1
2
3
4
5
6
7
8
9
10
```

```
public class MyClass {
public static void main(String[] args) {
int a = 64;
double b = 64.98;
System.out.println("a mod 10 = " + a % 10);
// Output: a mod 10 = 4
System.out.println("b mod 10 = " + b % 10);
// Output: b mod 10 = 4.980000000000004
}
}
```

If the statement:

```
int i = 1;
```

is replaced by:

```
int i = 0;
```

Then the `output on the screen` is:

```
0
1
2
3
4
5
6
7
8
9
10
```

```
public class MyClass {
public static void main(String[] args) {
int x[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
System.out.println("February has " + x[1] + " days");
}
}

// Output: February has 28 days
```

Similarly if the statement `int i = 0;` is replaced by:

```
int i = 7;
```

Then the **output on the screen** is:

```
7

8

9

10
```

Java Concurrency Utilities:

A method of asynchronous programming that let users propose or plan activities for parallel execution, construct threads that inherit Java EE context, and provide Java EE context to interfaces like asynchronous callbacks when they are invoked

- **Java program to print first 10 numbers using `do while loop` statement**

```
public class MyClass {
public static void main(String [] args) {
int i =1;
do {
System.out.println("\n i= " + i++);
} while (i<=10);
}
}
```

The output on the screen is:

```
i= 1
i= 2
i= 3
i= 4
i= 5
i= 6
i= 7
i= 8
i= 9
i= 10
```

The statement:

```
System.out.println("\n i= " + i++);
```

is executed and then condition (i<=10) is checked. If condition (i<=10) is true then

The statement:

```
System.out.println("\n i= " + i++);
```

is executed again. This process repeats until the given condition (i<=10) becomes false.

```
import java.math.BigInteger;

// Subtraction: 50 - 41 = 9

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("50");
BigInteger y = new BigInteger("41");
System.out.println(x.subtract(y));
}
}

// Output: 9
```

- **Program 3.4**

  Java program to print the characters from A to Z using `for loop`, `do while loop` and `while loop` statement.

  - **Java program to print the characters from A to Z using `for loop` statement:**

```
public class MyClass {
public static void main(String [] args) {
char a;
for( a='A'; a<='Z'; a++)
System.out.println("\n   " + a);
}
}
```

The output on the screen:

```
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
```

```
public class MyClass {
public static void main(String args[]) {
int x = 200, y = 300;
while(++x < --y) ;
// prints the midpoint between 'x' and 'y'
System.out.println("Midpoint is: " + x);
// Output: Midpoint is: 250
}
}
```

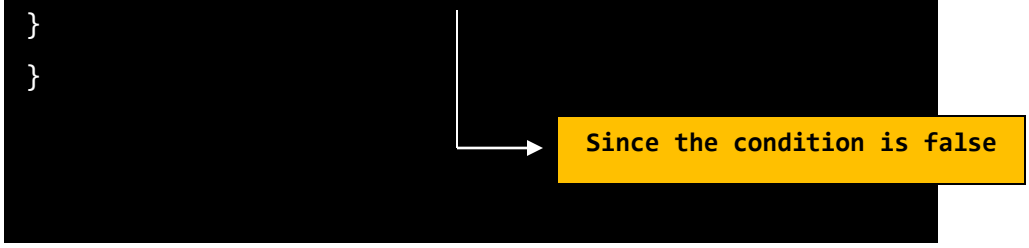
```
public class MyClass {
public static void main(String args[]) {
int x, y;
for(x=1, y=4; x<y; x++, y--) {
System.out.println("x : " + x);
System.out.println("y : " + y);
}
}
}
```

Output:

```
x : 1
y : 4
x : 2
y : 3
```

P
Q
R
S
T
U
V
W
X
Y
Z

```
public class MyClass {
public static void main(String args[]) {
int x = 20, y = 60;
while(x > y)
System.out.println("This won't appear on the screen");
}
}
```



- **Java program to print the characters from A to Z using `while loop` statement:**

```
public class MyClass {
public static void main(String [] args) {
char a = 'A';
while (a<='Z') {
System.out.println("\n " + a++);
}
}
}
```

- **Java program to print the characters from A to Z using `do while loop` statement:**

```
public class MyClass {
public static void main(String [] args) {
char a = 'A';
do {
System.out.println("\n "  + a++);
} while (a<='Z');
}
}
```

```
import java.math.BigInteger;

public class MyClass {
public static void main(String[] args) {
BigInteger x = BigInteger.valueOf(11);
BigInteger y = BigInteger.valueOf(11);
if(x.equals(y)) {
 System.out.println("Equal");
}
else {
 System.out.println("Not Equal");
}
}
}

// Output: Equal
```

```
import java.math.BigInteger;

// Addition: 50 + 50 = 100

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("50");
BigInteger y = new BigInteger("50");
System.out.println(x.add(y));
}
}

// Output: 100
```

- **Program 3.5**

  Java program to print the given number is even or odd.

```
import java.util.Scanner;

public class MyClass {

public static void main(String [] args) {

int a;

Scanner scan = new Scanner(System.in);

System.out.println("Enter a number: ");

a = scan.nextInt();
```

```
if(a%2 == 0) {
System.out.println("The number is even");
}
else {
System.out.println("The number is odd");
}
}
}
```

The output on the screen:

```
Enter a number:
If you enter the number 4
The number is even
will be outputted on the screen.
```

```
// Using break to exit a loop
public class MyClass {
public static void main(String args[]) {
for(int x=0; x<10; x++) {
if(x == 3) break; // terminate loop if x is 3
System.out.println(x);
}
System.out.println(".....");
}
}
```

Output:

```
0
1
2
.....
```

- **Program 3.6**

  Java program to print the remainder of two numbers

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
a = scan.nextInt();
System.out.println("Enter a number: ");
b = scan.nextInt();
c = a%b;
System.out.println("The remainder of a and b =  " + c);
}
}
```

The output on the screen:

```
Enter a number:
If you enter the number 3
Enter a number:
If you enter the number 2
The remainder of a and b = 1
will be outputted on the screen.
```

Since (a=3 and b=2). **Therefore:** `3 divided by 2` (i.e., `a divided by b`) yields the remainder equal to 1.

- **Program 3.7**

  Java program to check equivalence of two numbers.

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int x, y;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
x = scan.nextInt();
System.out.println("Enter a number: ");
y = scan.nextInt();
if(x-y==0) {
System.out.println("The 2 numbers are equivalent");
}
else {
System.out.println("The 2 numbers are not equivalent");
}
}
}
```

The output on the screen:

```
Enter a number:

2 # entered number

Enter a number:

2 # entered number

The 2 numbers are equivalent
```

- **Program 3.8**

  Java program to print the leap year or not

```
public class MyClass {
public static void main(String [] args) {
int year;
year =1996;
if(year%4==0) {
System.out.println("Leap year");
}
else {
System.out.println("Not a leap year");
}
}
}
```

The output on the screen:

```
Leap year
```

- **Encapsulation** enables us to change our implementations over time without affecting the code that relies on the **classes' public interface**
- **Polymorphism** enables us to create clear, sane, readable, resilient code

- **What will be the output on the screen?**

```
public class MyClass {int
a=5;
public static void main(String[] args) {
int a=2;
System.out.println("Value of a = " + a);
}
}
```

Answer:

```
Value of a = 2
```

```
public class MyClass {
public static void main(String[] args) {
String x;
x = "Einstein";
System.out.print("Albert ");
System.out.println(x);
}
} // Output: Albert Einstein
```

If the statement:

```
System.out.println("Value of a = " + a);
```

is replaced by the statement

```
System.out.println("Value of a = " + ::a);
```

(where **::** denote scope resolution operator)

i.e.,

```
public class MyClass {
int a=5;
public static void main(String[] args) {
int a=2 ;
System.out.println("Value of a = " + ::a);
}
}
```

Then the **compilation error** will be displayed on the screen because [like **C++**] JAVA does not hold **or** support the **resolution operator**.

- **Program 3.9**

  Java program to print whether the given number is positive or negative

```
public class MyClass {
public static void main(String [] args) {
int a;
a = -35;
if(a>0) {
System.out.println("Number is positive");
}
else {
System.out.println("Number is negative");
}
}
}
```

The output on the screen:

```
Number is negative
```

Since `a = -35`. Therefore:

```
a is less than 0 i.e., a<0
```

The statement

```
System.out.println("Number is negative");
```

is executed to print the output:

```
Number is negative
```

- **Program 4.0**

  Java program to print the sum of the first 10 digits using **`for loop`** statement:

```
public class MyClass {

public static void main(String [] args) {

int i,  sum = 0;

for(i=1; i<=10; i++)

sum = sum + i;

System.out.println("Sum of  the first 10 digits = " + sum);

}

}
```

The output on the screen:

```
Sum of the first 10 digits = 55
```

The statement:

```
System.out.println("Sum of  the first 10 digits = " + sum);
```

is executed to display the output:

```
Sum of the first 10 digits = 55
```

on the screen.

If the **statement**:

```
int i, sum = 0;
```

is replaced by:

```
int i, sum = 1;
```

Then the **output on the screen** is:

```
Sum of the first 10 digits = 56
```

- **What will be the output if the `for loop` statement `for(i=1; i<=10; i++)` is replaced by the statement `for(i=2; i<10; i++)`?**

  Answer: `Sum of 10 digits = 44`

Generalization:

Code that has been generalised is more adaptable, more likely to be reused, and occasionally even simpler to implement

If the **statement:**

`int i, sum, sum = 0;` is written instead of `int i, sum = 0;`

Then the **`compilation error message`** will be displayed on the screen (**stating that sum is twice declared**).

If the **`for loop`** is ended with a semicolon i.e.,

```
for(i=1; i<=10; i++);
```

Then the **`compilation error`** will be displayed on the console screen.

- **Program 4.1**

  Java program to print the average of the first 10 numbers using **`for loop`** statement

```
public class MyClass {
public static void main(String [] args) {
int i, avg, sum = 0;
for(i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
System.out.println("Sum of the first 10 numbers = " + sum);
System.out.println("Average of the first 10 numbers =  " + avg);
}
}
```

The output on the screen:

```
Sum of the first 10 numbers = 55
Average of the first 10 numbers = 5
```

If the **data type float** is used i.e.,

```
public class MyClass {
public static void main(String [] args) {
float i, avg, sum = 0;
for(i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
System.out.println("Sum of the first 10 numbers = " + sum);
System.out.println("Average of the first 10 numbers = " + avg);
}
}
```

The output on the screen:

```
Sum of the first 10 numbers = 55
Average of the first 10 numbers = 5.5
```

- **Program 4.2**

  Java program to print the product of the first 10 digits using **for loop** statement

```
public class MyClass {
public static void main(String [] args) {
int i, product = 1;
for(i=1; i<=10; i++)
product = product * i;
System.out.println("The product of the first 10 digits = " + product);
}
}
```

The output on the screen:

```
The product of the first 10 digits = 3628800
```

The statement:

```
System.out.println("The product of the first 10 digits = " + product);
```

is executed to display the output:

```
The product of the first 10 digits = 3628800
```

If the statement `int i, product = 1;` is replaced by `int i, product = 0;`

Then the output on the screen is:

```
The product of the first 10 digits = 0
```

If the statement `for(i=1; i<=10; i++)` is replaced by `for(i=5; i<=8; i++)`

Then the output on the screen is:

```
The product of the first 10 digits = 1680
```

- **Program 4.3**

  Java Program to print the table of a number using the **`for loop`** statement

```
import java.util.Scanner;

public class MyClass {
public static void main(String [] args) {
int x, i;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
x = scan.nextInt();
for(i=1; i<=5; i++)
System.out.println (\n x  +  "  *    " +  i + "  =  " +  x * i);
}
}
```

Output on the screen:

```
Enter any number:
If you enter the number 2 (i.e., x=2)

2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10

will be outputted on the screen.
```

If the symbol * is replaced by +

i.e.,

```
import java.util.Scanner;

public class MyClass {

public static void main(String [] args) {

int x, i;

Scanner scan = new Scanner(System.in);

System.out.println("Enter a number: ");

x = scan.nextInt();

for(i=1; i<=5; i++)

System.out.println ( \n x  +  "  +   " +  i + "  =  " +  x + i);

}

}
```

Then the output on the screen is:

```
Enter any number:
If you enter the number 2 (i.e., x=2)

2 + 1 = 3
2 + 2 = 4
2 + 3 = 5
2 + 4 = 6
2 + 5 = 7

will be outputted on the screen.
```

```
public class MyClass {
public static void main(String... args) {
int x[] = {16, 3, 6, 3, 2};
int i = 0;
for (; ;) {
if (i >= x.length) {
break;
}
System.out.println("x[" + i + "] = " + x[i]);
i++;
}
}
}
```

Output

```
x[0] = 16
x[1] = 3
x[2] = 6
x[3] = 3
x[4] = 2
```

- **Program 4.4**

Java program to print the first 10 numbers starting from one together with their squares

```
public class MyClass {
public static void main(String[] args) {
int i;
for(i=1; i<=10; i++)
System.out.println("\n number = " +  i +  " its square =  " + i*i);
}
}
```

Output:

```
number = 1 its square=1
number = 2 its square=4
number = 3 its square=9
number = 4 its square=16
number = 5 its square=25
number = 6 its square=36
number = 7 its square=49
number = 8 its square=64
number = 9 its square=81
number= 10 its square=100
```

Blocking method:

A method that halts the thread that is currently running until their operation is complete. The InputStream read() method, which stops until the entire contents of the InputStream have been read, is a well-known example of the blocking method

If the **statement:**

```
System.out.println(" \n number = " +  i +  " its square =  " + i*i);
```

is replaced by the statement:

```
System.out.println(" \n number = " +  i +  " \t its square =  " + i*i);
```

i.e.,

```
public class MyClass {
public static void main(String[] args) {
int i;
for(i=1; i<=10; i++)
System.out.println("\n number = " +  i +  " \t its square =  " + i*i);
}
}
```

Then the output on the screen is:

```
      number=1     its square=1
      number=2     its square=4
      number=3     its square=9
      number=4     its square=16
      number=5     its square=25
      number=6     its square=36
      number=7     its square=49
      number=8     its square=64
      number=9     its square=81
      number=10    its square=100
```

tab /t is included because to leave space between

number=1 and its square=1

```java
import java.math.BigInteger;

// Division: 20/2 = 10

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("20");
BigInteger y = new BigInteger("2");
System.out.println(x.divide(y));
}
}

// Output: 10
```

```java
import java.math.BigInteger;

// Multiplication: 50 * 2 = 100

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("50");
BigInteger y = new BigInteger("2");
System.out.println(x.multiply(y));
}
}


// Output: 100
```

```java
import java.util.*;

public class MyClass {
public static void main(String[] args) {
Set<Integer> x = new TreeSet<Integer>();
List<Integer> y = new ArrayList<Integer>();
for(int i = -3; i < 3; i++) {
x.add(i);
y.add(i);
}
for(int i = 0; i < 3; i++) {
x.remove(i);
y.remove(i);
}
System.out.println(x + " " + y);
}
} // Output: [-3, -2, -1] [-2, 0, 2]
```

This program removes the non-negative values from the set 'x' and the odd values from the list 'y'

Aliasing:

The condition when 2 or more variables refer to the same object

```java
import java.math.BigInteger;

// Power: 10 ^ 2 = 10 * 10 = 100

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("10");
System.out.println(x.pow(2));
}
}

// Output: 100
```

```java
import java.util.List;
public class MyClass {
public static void main(String... args) {
List<Integer> x = List.of(8, 6, 5, 4, 1);
for(int i = 0; i < x.size(); i++) {
System.out.println("x[" + i + "] = " + x.get(i));
}
}
}
```

Output:

```
x[0] = 8
x[1] = 6
x[2] = 5
x[3] = 4
x[4] = 1
```

```java
import java.math.BigInteger;

// Remainder: 20 % 8 = 4

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("20");
BigInteger y = new BigInteger("8");
System.out.println(x.remainder(y));
}
}

// Output: 4
```

Pseudorandom:

A sequence of integers that seem random but were actually generated by a deterministic computation

- **Write a program to print the first 10 numbers starting from one together with their squares and cubes:**

Answer:

```
public class MyClass {
public static void main(String[] args) throws Exception {
int i;
for(i=1; i<=10; i++)
System.out.println(" \n number = " + i + " its square = " + i*i + " its cube = " +i*i*i);
}
}
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println("Albert");
System.out.println("Albert");
} // Printing begins in the next line
}
```

```
Albert
Albert
```

Output

```
public class MyClass {
public static void main(String[] args) {
System.out.print("Albert");
System.out.print("Albert");
} // Printing begins in the same line
}

// Output:
                AlbertAlbert
```

- **Program 4.5**

  Java program to print the sum of two numbers using method

```
public class MyClass {
public static void main(String[] args) {
int a, b, c;
a = 11;
b = 6;
c = add (a, b);
System.out.println("Sum of two numbers = " + c);
}
public static int add (int a, int b) {
```

```
return (a+b) ;
}
}
```

The output on the screen:

```
Sum of two numbers = 17
```

There are 2 methods in the above program:

- `public static void main(String[] args)`
- `public static int add (int a, int b)`

`public static void main(String[] args)` `imply`: **main method** and

{

} imply the body of the **main method** with in which the program statements:

```
int a, b, c;
a = 11;
b = 6;
c = add (a, b);
System.out.println("Sum of two numbers = " + c);
```

are written.

`public static int add (int a, int b)` **imply: the** method to add two integers `'x'` and `'y'` and

```
{
return (a+b);
}
```

imply the body of the method **`public static int add (int a, int b)`**

main method:

```
public static void main(String[] args)
```

and the **method:**

```
public static int add (int a, int b)
```

should be written inside the body of the **`public class MyClass.`**

The statement

```
int a, b, c;
```

imply that we creating the integer variables `a`, `b` and `c`.

The **statements:**

```
a = 11;
b = 6;
c = add (a, b);
```

imply that we are assigning the values to the created variables.

The statement:

```
c = add (x, y);
```

imply **`method call`** (i.e., we are calling the method `public static int add (int a, int b)`

to add the values (i.e., 11 and 6) and return the result (i.e., 17) to the statement

```
System.out.println("Sum of two numbers = " + c);
```

to make provision to display the output of the sum of two entered numbers as 17 on the screen.

```
import java.math.BigInteger;

// GCD: Greatest Common Divisor (GCD) for 14 and 20 is 2

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("14");
BigInteger y = new BigInteger("20");
System.out.println(x.gcd(y));
}
}

// Output: 2
```

- **Java program to print the product of two numbers using method**

```
public class MyClass {
public static void main(String[] args) {
int a, b, c;
a = 2;
b = 3;
c = mult (a, b);
System.out.println("Product of two numbers  = " + c);
}
public static int mult (int a, int b) {
return (a*b) ;
}
}
```

The output on the screen:

```
Product of two numbers = 6
```

will be outputted on the screen.

```
if(x < 20) System.out.println("x is less than 20");

If 'x' contains a value that is less than 20, the conditional expression is true, then the println() will get executed. If 'x' contains a value greater than or equal to 20, the condition is not true, then the println() method is bypassed
```

- **Java program to print the greatest of two numbers using method**

```
import java.util.Scanner;

public class MyClass {
public static void main(String[] args) {
int a, b;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any two numbers: ");
a = scan.nextInt();
b = scan.nextInt();
System.out.println("Largest of two numbers  = " + max (a, b) );
}
public static int max (int a, int b) {
if(a>b)
return a;
else
return b;
}
}
```

The output on the screen:

```
Enter any two numbers:

If you enter two numbers 5 and 2

Largest of two numbers= 5
```

will be outputted on the screen.

- **Java program to print the greatest of three numbers using method**

```
import java.util.Scanner;

public class MyClass {
public static void main(String[] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any three numbers: ");
a = scan.nextInt();
b = scan.nextInt();
c= scan.nextInt();
System.out.println("Largest of two numbers  = " + max (a, b, c) );
}
public static int max (int a, int b, int c) {
if(a>b && a>c)
return a;
else if (b>c && b>a)
return b;
else
return c;
}
}
```

The output on the screen:

```
Enter any three numbers:
If you enter three numbers 3, 5 and 10
Largest of three numbers = 10
will be outputted on the screen.
```

Abstract parameter:

A group of parameters that act together as a single parameter

- **Java program to print the square of the number using method**

```
import java.util.Scanner;

public class MyClass {
public static void main(String[] args) {
int x;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any number: ");
x = scan.nextInt();
System.out.println("Square of the number = " + square (x));
}
public static int square (int x) {
return x*x;
}
}
```

The output on the screen is:

```
Enter any number:

If you enter the number 5

Square of the number = 25

will be outputted on the screen.
```

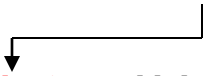
Object method

A method that is called on an object and performs operations on it. The keyword "static" does not exist in object methods

```
import java.math.BigInteger;

// Maximum of two BigIntegers: 21

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("20");
BigInteger y = new BigInteger("21");
System.out.println(x.max(y));
}
}

// Output: 21
```

Class method: A method that includes the keyword static.

Class methods have no **current object** and are not called on objects



The object on which an object method is invoked

- **Program 4.6**

  `Switch (case)` allows to make decision from the number of choices i.e., from the numberof cases

For example:

```
public class MyClass {
public static void main(String[] args)throws Exception {
char ch;
System.out.print("Enter a character: ");
ch = (char)System.in.read();
switch(ch) {
case 'R': System.out.print("Red");
break;
case 'W': System.out.print("White");
break;
case 'Y': System.out.print("Yellow");
break;
case 'G': System.out.print("Green");
break;
default: System.out.print("Error");
break;
}
}
}
```

The output on the screen is:

```
Enter a character:
If you enter a character R
Red
will be outputted on the screen.
```

```
// Char variables behave like integers
public class MyClass {
public static void main(String args[]) {
char i = 'B';
System.out.println(i); // Output: B
i++; // i is incremented
System.out.println(i); // Output: C
}
}
```

- **Program 4.7**

Java program to print the output

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
```

using arrays:

```
public class MyClass {
public static void main(String[] args) {
int i;
int [] num = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
System.out.println("Element [" +  i + " ] = " + num[i]);
}
}
```

The output on the screen:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
```

Ends because of the condition **`i<5`**.

```
public class MyClass {
public static void main(String args[]) {
boolean i = false;
int x = 0;
for( ; !i; ) {
System.out.println(x);
if(x == 3) i = true;
x++;
}
}
}
```

Output:

```
0
1
2
3
```

```
public class MyClass {
public static void main(String args[]) {
System.out.println("Albert Einstein");
if(true) return; // stop the execution of a method
System.out.println("This statement is not executed.");
}
}
```

Output:

```
Albert Einstein
```

Array declaration in C:

```
int num [5] = {16, 18, 20, 25, 36};
            or
int num [] = {16, 18, 20, 25, 36};
```

Array declaration in C++:

```
int num [5] = {16, 18, 20, 25, 36};
            or
int num [] = {16, 18, 20, 25, 36};
```

But array declaration in java:

```
int [] num = {16, 18, 20, 25, 36};
```

```
import java.math.BigInteger;

// Minimum of two BigIntegers: 20

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("20");
BigInteger y = new BigInteger("21");
System.out.println(x.min(y));
}
}


// Output: 20
```

Classpath:

A parameter in the Java Virtual Machine or Java Compiler that indicates where user-defined classes and packages are located

- **Java program to print the sum of the elements in array.**

```
public class MyClass {
public static void main(String[] args) {
int i, sum = 0;
int [] num = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
System.out.println("Sum of the Elements in the array  = " +  sum);
}
}
```

The output on the screen:

```
Sum of the Elements in the array = 115
i.e., 16 + 18 + 20 + 25 + 36 = 115
```

```
import java.math.BigInteger;

// Binary Or: 10 | 9 = 11

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("10");
BigInteger y = new BigInteger("9");
System.out.println(x.or(y));
}
}
// Output: 11
```

- **Java program to print the average of the elements in the array**

```
public class MyClass {

public static void main(String[] args) {
int i, avg, sum = 0;
int [] num = {16, 18, 20, 25, 36};

for(i=0; i<5; i++)
sum = sum + num[i];
avg = sum/5;
System.out.println("Sum of the Elements in the array = " + sum);
System.out.println("Average of the Elements in the array = " +  avg);
}

}
```

The output on the screen:

```
Sum of the Elements in the array = 115
Average of the elements in the array = 23
```

- **Write a program to print**

```
Einstein [0] = E
Einstein [1] = I
Einstein [2] = N
Einstein [3] = S
Einstein [4] = T
Einstein [5] = E
Einstein [6] = I
Einstein [7] = N
```

using arrays

```
public class MyClass {
public static void main(String[] args) {
System.out.println(98765 + 4321L);
// Output: 103086
}
}
```

Compilation unit:

The smallest unit of source code that can be compiled

Compositing:

Combining multiple images into one through the method of superimposing

Answer:

```
public class MyClass {
public static void main(String[] args) throws Exception {
int i;
char [] num = {'E' , 'I', 'N', 'S', 'T', 'E', 'I', 'N'};
for(i=0; i<8; i++)
System.out.println("Einstein [" + i + " ] = " + num[i]);
}
}
```

- **What will be the output of the following programs?**

i)

```
public class MyClass {
public static void main(String[] args) throws Exception {
int i;
int [] name = {'E' , 'I', 'N', 'S', 'T ', 'E', 'I', 'N'};
for(i=0; i<8; i++)
System.out.println("Einstein [" +  i + " ] = " + name[i]);
}
}
```

Answer:

```
Einstein [0] = 69
Einstein [1] = 73
Einstein [2] = 78
Einstein [3] = 83
Einstein [4] = 84
Einstein [5] = 69
```

```
Einstein [6] = 73
Einstein [7] = 78
```

ii)

```
public class MyClass {
public static void main(String[] args) throws Exception {
int i;
char [] body = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
System.out.println("body [" +  body [i] + " ] = " + body [i]);
}
}
```

Answer:

```
body [b] = b
body [o] = o
body [d] = d
body [y] = y
```

```
import java.util.Arrays;

public class MyClass {
public static final int x[] = {15, 6, 8, 9, 2};
public static void main(String... args) {
for(int i = 0; i < x.length; i++) {
if(i == 2) {
System.out.println(".........");
break;
}
System.out.println("x[" + i + "] = " + x[i]);
}
}
}
```

Output:

```
x[0] = 15
x[1] = 6
.........
```

```
import java.math.BigInteger;

// Binary And: 10 & 9 = 8

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("10");
BigInteger y = new BigInteger("9");
System.out.println(x.and(y));
}
}
// Output: 8
```

Constructor:

A pseudo-method that creates an object

- **What is the mistake in the following program?**

```
public class MyClass {
public static void main(String [] args) {
long float x;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");
x = scan.nextFloat();
System.out.println(" square root of x = " + Math.cbrt(x));
}
```

Answer:

`long float x;` should not be used – only `float x` should be used because **Java** do not support the data type such as long int, long float etc.

```
import java.math.BigInteger;

// Binary Xor: 10 ^ 9 = 3

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("10");
BigInteger y = new BigInteger("9");
System.out.println(x.xor(y));
}
}

// Output: 3
```

Critical section:

The term "critical section" refers to a code segment that is accessed by multiple processes

- **Program 4.8**

  Continue and break statements:

A)

```
public class MyClass {
public static void main(String []args) {
int i;
for (i=1; i<=5; i++){
if (i==3) {
continue;
}
System.out.println("" + i);
}
}
}
```

```
public class MyClass {
public static void main(String... args) {
// The size of the array 'x' is: 3
int[] x = new int[3];
for(int i = 0; i < x.length; i++) {
System.out.println("x["+ i +"]= " + x[i]);
}
}
}
```

Output:

```
x[0]= 0
x[1]= 0
x[2]= 0
```

Output on the screen:

```
1
2
4
5
```

B)

```
public class MyClass {

public static void main(String []args) {

int i;

for (i=1; i<=5; i++){

if (i==3){

break;
```

Logging:

Creating log files that can be utilized to track down code issues

```
}
System.out.println("" + i);
}
}
}
```

Output on the screen:

1

2

```
import java.util.List;

public class MyClass {
public static void main(String... args) {
List<Integer> x = List.of(25, 8, 3);
x.forEach(item -> System.out.println(item));
}
}
```

Output:

25

8

3

- **What will be the output of the following program?**

```
public class MyClass {
public static void main(String args[]) {
System.out.println(Math.max(1269, 1356));
}
}
```

Output on the screen:

1356

```
Abstraction ⟶ Hiding implementation details from the user by providing interface


Encapsulation ⟶ Hiding data
```

In the statement:

"1 + 2"

"1" and "2" imply the **operands** and the **plus symbol** imply the **operator**.

- **Polymorphism**

Suppose if you are in **class room** that time you behave like a student, when you are in **shopping mall** at that time you behave like a customer, when you at your **home** at that time you behave like a son or daughter. Your ability to present in different-different behaviors is known as polymorphism.

In the example:

```
public class MyClass {
public static void main(String [] args) {
int a, b, sum;
a=1;
b=2;
sum = a + b;
System.out.println("The sum of a and b = " + sum);
}
}
```

Plus symbol ("+") act as an arithmetic operator in the statement:

```
sum = a+b;
```

and it acts as the **concatenation operator** in the statement:

```
System.out.println("The sum of a and b = " + sum);
```

The ability of **`plus symbol`** to behave both as **arithmetic operator** and concatenation operator is known as polymorphism.

- **Inheritance**

```
public class game {
}

public class player extends game {
}
```

```
import java.util.Arrays;
public class MyClass {
public static final int x[] = {12, 8, 9, 6, 1};
public static void main(String... args) {
for (int i = 0; i < x.length; ++i) {
if (i % 2 != 0) {
continue;
}
System.out.println("x[" + i + "] = " + x[i]);
}
}
}
```

Output →

```
x[0] = 12
x[2] = 9
x[4] = 1
```

Here public class player extends game implies: class player is public and it is the sub class of the **class game**. Since class player is the subclass of class game – class player automatically takes on all the behavior and attributes of its parent class "game" i.e., methods or fields within the class game will be automatically be included in the class player.

The statements:

```
public class player extends game

public class game extends ball
```

implies: that **class player** is not only a subclass of class game but also it is a subclass of class ball.

- **Encapsulation**

```
public class Account {
           private decimal accountBalance = 500.00;

           public decimal CheckBalance() {
                    return accountBalance;
           }
  }

/* accountBalance can be checked via public "CheckBalance" method provided by the "Account" class but its value cannot be manipulated because data variable accountBalance is declared private */
```

Encapsulation is the technique of bringing the data variables and methods in single frame and declaring data variable private (so it cannot be accessed by anyone outside the class, thereby **hiding** or **encapsulating** the data variable (String name) within the public class Student) and providing indirect access to the data variable via public methods.

```
import java.math.BigInteger;

// RightShift: 10 >> 1 = 5
// LeftShift: 10 << 1 = 20

public class MyClass {
public static void main(String[] args) {
BigInteger x = new BigInteger("10");
System.out.println(x.shiftRight(1)); // Output: 5
System.out.println(x.shiftLeft(1)); // Output: 20
}
}
```

```
public class MyClass {
public static void newLine() {
System.out.print("\n");
}
public static void main(String[] args) {
System.out.print("Albert");
newLine();
System.out.print("John");
}
}

// Output:
                  Albert
                  John
```

```
import java.util.Arrays;

public class MyClass {
public static void main(String... args) {
int x[] = {6, 7, 9, 8, 2};
Arrays.stream(x).forEach(System.out::println);
}
}
```

Output:

6
7
9
8
2

```
public class MyClass {
public static void main(String[] args) {
int i = 10;
boolean x = (i%2 == 0);
if(x) {
System.out.println(i + " is even number");
}
}
}

// Output: 10 is even number
```

Deprecation:

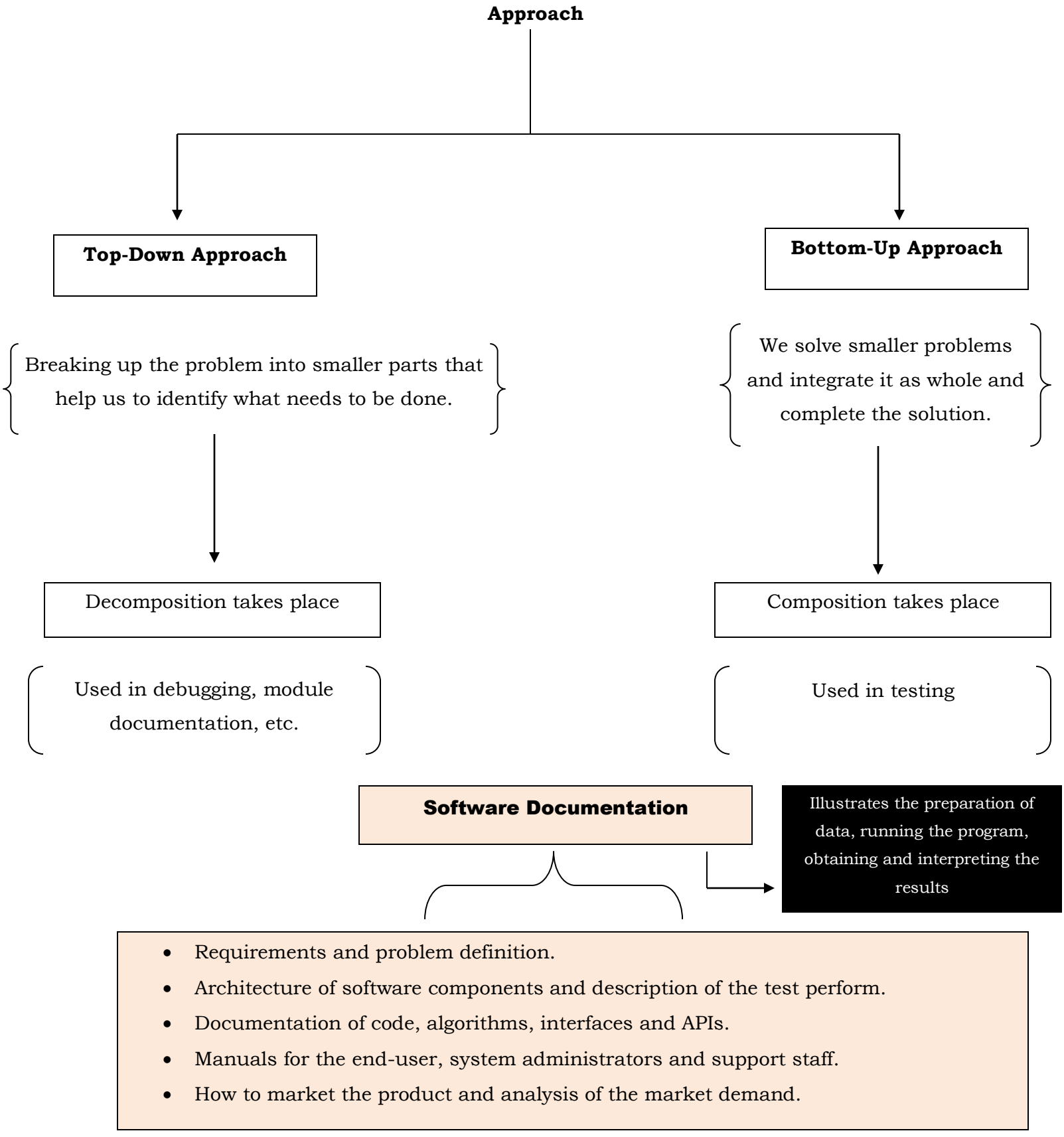
Refers to a class, interface, constructor, method, or field that is no longer suggested and might disappear in a future update

```
public class MyClass {
public static void main(String[] args) {
System.out.println(new Character('K').charValue());
// Output: K
}
}
```

```
import java.util.regex.Pattern;

public class MyClass {
public static void main(String[] args) {
System.out.println(MyClass.class.getName().replaceAll("\\.", "/") + ".class");
// Output: MyClass.class
System.out.println(MyClass.class.getName().replaceAll(Pattern.quote("."), "/") + ".class");
// Output: MyClass.class
}
}
```

```
public class MyClass {
public static void main(String[] args) {
double x = 0.0 / 0.0;
System.out.println(x - x == 0);
// Output: false
}
}
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println(new Integer(5) == new Integer(5));
// Output: false
System.out.println(new Integer(5) == 5);
// Output: true
}
}
```


Approach
Top-Down Approach
Bottom-Up Approach
Breaking up the problem into smaller parts that help us to identify what needs to be done.
We solve smaller problems and integrate it as whole and complete the solution.
Decomposition takes place
Composition takes place
Used in debugging, module documentation, etc.
Used in testing
Software Documentation
Illustrates the preparation of data, running the program, obtaining and interpreting the results
Requirements and problem definition.
Architecture of software components and description of the test perform.
Documentation of code, algorithms, interfaces and APIs.
Manuals for the end-user, system administrators and support staff.
How to market the product and analysis of the market demand.

```
print("Jean-Luc Picard".lower())
```

Output

jean-luc picard

Program maintenance

- Find, eliminate and correct program errors
- Modify the program and Improve its performance
- Add new features, functionalities and a better user interface
- Update the documentation

The best programs are written so that computing machines can perform them quickly and so that human beings can understand them clearly. A programmer is ideally an essayist who works with traditional aesthetic and literary forms as well as mathematical concepts, to communicate the way that an algorithm works and to convince a reader that the results will be correct.

— Donald E. Knuth, Selected Papers on Computer Science

Classification

Binary Classification

A given data is being classified into two classes

2 emails are sent to us, one is sent by a mobile advertising company that keeps sending their ads, and the other is from our bank regarding our balance. The email service provider will classify the 2 emails, the first one will be sent to the spam folder and the second one will be kept in the primary folder.

Multiclass Classification

A given data is being classified into more than two classes

Classifying books according to their Genre

Ensemble Learning

A machine learning model that combines the predictions from two or more models

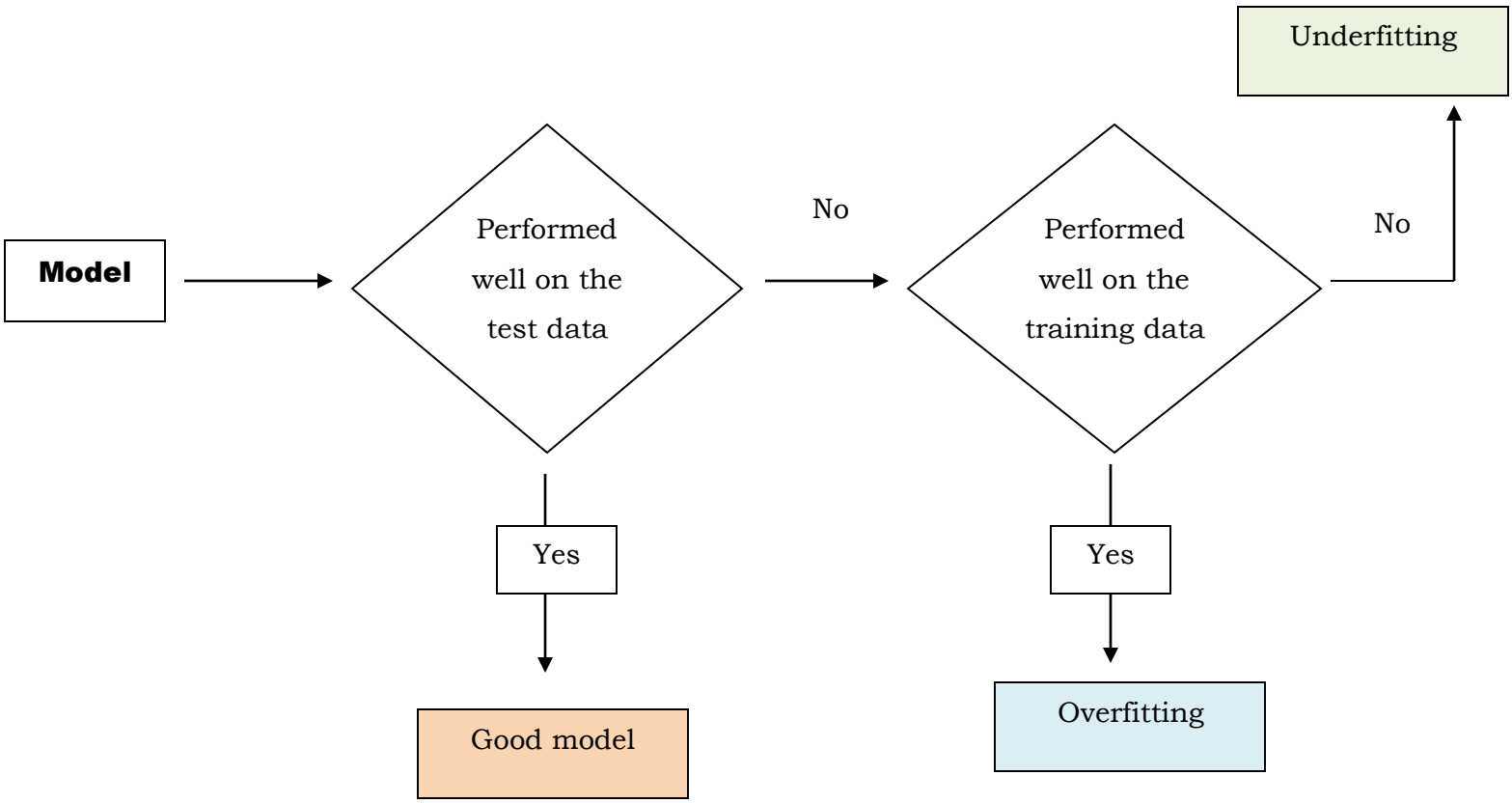
Parametric method	**Non-Parametric method**
Information is completely known and specific assumptions are made	No Information is available and No assumptions are made

Dimensionality reduction

A Technique that reduce the number of input variables in training data

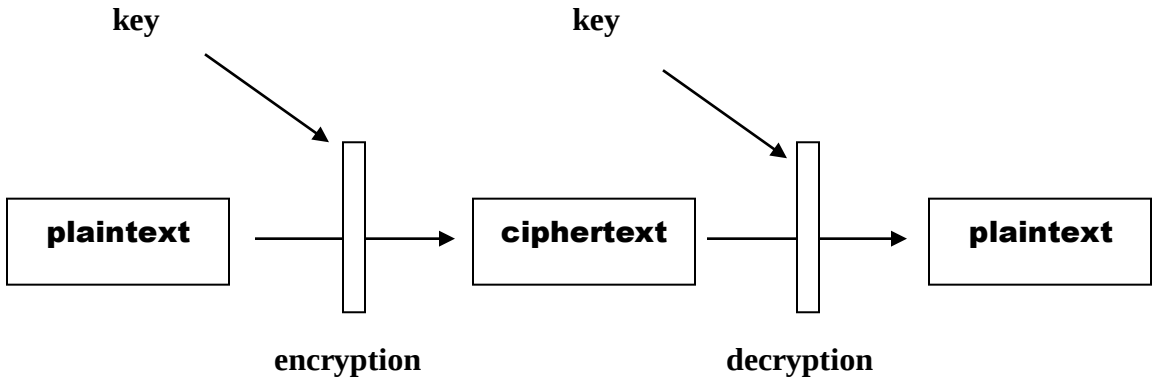
- It helps in data compression and hence reduced storage space.
- It reduces computation time.

Training set	Testing set
The portion of the dataset used to train the model	The portion of the dataset used to test the trained model

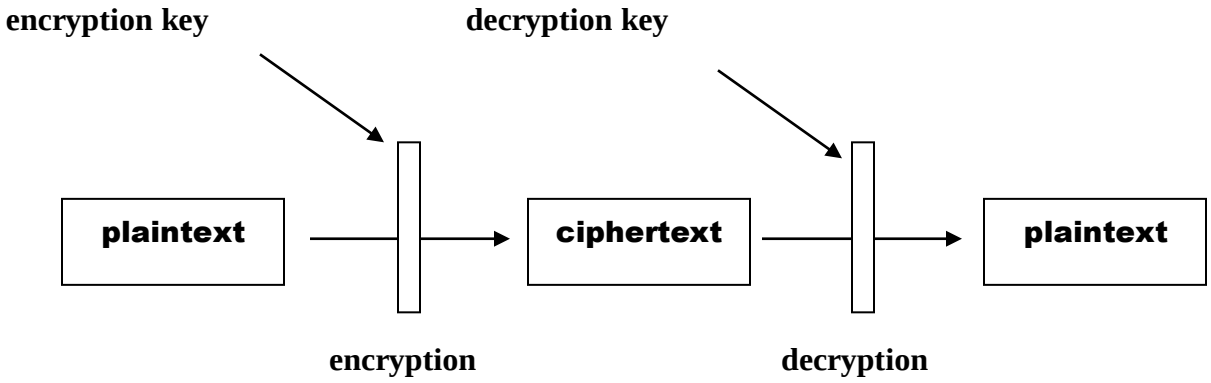


Testing	Debugging
Finding and locating of a error	Fixing that error
Done by testing team	Done by development team
Find as many error as possible	Remove those errors
can be automated	cannot be automated

Symmetric Algorithms: use the same key for encryption and decryption



Asymmetric Algorithms: use a different key for encryption and decryption



Verification ↓ Are you building it right?	Validation ↓ Are you building the right thing?
Check whether the software meets all the functionality	Check whether the functionality meets the intended behavior
Done by developers	Done by testers

Scripted testing	Unscripted testing
Test execution carried out by following a previously documented sequence of tests	Test execution carried out without previously documented sequence of tests

Black-box testing method

Software testing method in which the internal structure of code or program being tested is not known to the tester

White-box testing method

Software testing method in which the internal structure of code or program being tested is known to the tester

Gray-box testing method = Black-box testing method + White-box testing method

- **Error:** a mistake made by developer that leads to discrepancy between the actual and the expected result.
- **Defect:** a problem in the functioning of software during testing.

- **Fault:** an incorrect step, process or data definition in software.
- **Bug:** a flaw in software that causes the software to behave in an unintended manner.
- **Failure:** inability of software to perform its operations within the specified performance benchmark.

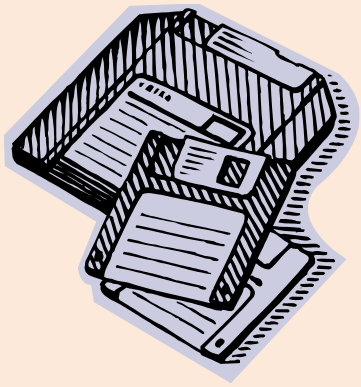
System Integration Testing	User Acceptance Testing
Testing the functionality of a system as a whole after integrating all the system components	Testing the system from the user's perspective

Proactive testing	Reactive testing
Focusing on eliminating problems before they have a chance to appear	Reacting to problems when they occur instead of doing something to prevent them
Testing is initiated as early as possible in order to find and fix the errors before the build is created	Testing is not started until design and coding are completed

Exploratory testing	Adhoc testing
↓ Exploring the application while learning (Simultaneous learning, test design and test execution) **(Documentation is mandatory)**	↓ Learning application first and then work with actual testing process **(Documentation is not a basic need)**
Aim: to get information to design new and better tests	**Aim:** to find errors
Result: • errors are found and registered • new tests are designed and documented for further usage	**Result:** • errors are found and registered

Structured query language

SQL: Structured query language – a standard computer language developed by American computer scientists **Donald D. Chamberlin** and **Raymond F. Boyce** at IBM in 1974 to create database, store, update, manipulate, delete and retrieve data stored in database. It became a standard of the American National Standards Institute (ANSI) in 1986 and of the International Organization for Standardization (ISO) in 1987. SQL is **NOT case sensitive**: select is the same as SELECT.



How to create database in MySQL

First you have to open `MySQL terminal` and then you have to enter the command:

```
create database data;
                                        or
CREATE DATABASE data;
```

And press enter. Then

```
Query OK, 1 row affected (0.01 sec)
```

will be displayed on the console screen indicating that database named data is created. And if you enter the command:

```
show databases;
```

And press enter. Then

```
+--------------------
+

| Database           |


+--------------------
+

| CODINGGROUND       |


| data               |


| information_schema |


| mysql              |


| performance_schema |


| test
```

will be displayed on the console screen. And if you want to create a table in the database "data", then you have to enter the command:

```
use data;
```

And press enter. Then

```
Database changed
```

will be displayed on the console screen stating that your active database is now "data". And if you want to create a table named "states" with three fields: id, state, and population:

id	state	population

in your active database named "data", then you have to enter the command:

```
CREATE TABLE states (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, state CHAR(25),
population INT(9));
```

And press enter. Then

```
Query OK, 0 rows affected (0.07 sec)
```

will be displayed on the console screen stating that the above table is created.

Note:

- The INT command will make the id field contain only numbers (i.e., integers).
- The NOT NULL command makes sure that the id field cannot be left blank or empty.
- The PRIMARY KEY designates the id field as the key field in the table.

- The AUTO_INCREMENT command will automatically assign increasing values into the id field, essentially automatically numbering each entry.
- The CHAR(**characters**) and INT(**integers**) commands designate the types of data allowed in those fields. The number next to the commands CHAR and INT indicate how many characters or integers can fit in the field.

Now it's time to start entering your information. Use the following **command**:

```
INSERT INTO states (id, state, population) VALUES (NULL, 'Karnataka', 256666); INSERT
INTO states (id, state, population) VALUES (NULL, 'Assam', 2568585); INSERT INTO states
(id, state, population) VALUES (NULL, 'Kashmir', 2569);
```

to input your entry. Then

```
Query OK, 1 row affected (0.03 sec)



Query OK, 1 row affected (0.01 sec)



Query OK, 1 row affected (0.00 sec)
```

will be displayed on the **console screen** stating that you have inputted your entry. And if you enter the following command:

```
select*from states;
```

Then, your created table named "states" will be displayed on the screen as follows:

```
+----+-----------+------------
+

| id | state     | population |


+----+-----------+------------
+

|  1 | Karnataka |     256666 |


|  2 | Assam     |    2568585 |


|  3 | Kashmir   |       2569 |


+----+-----------+------------+
```

And if you wish to create the following table

```
+----+-----------+------------+----------
+

| id | state     | population | language |


+----+-----------+------------+----------
+

|  1 | Karnataka |     256666 | Kannada  |
```

```
|  2 | Assam     |       2569 | Assami   |
```

You have to use the following command:

```
CREATE TABLE states (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, state CHAR (25),
population INT (9), language CHAR (25));
```

And press enter and

```
Query OK, 0 rows affected (0.03 sec)
```

will be displayed on the console screen and then you should enter the following command:

```
INSERT INTO states (id, state, population, language) VALUES (NULL, 'Karnataka', 256666,
'Kannada'); INSERT INTO states (id, state, population, language) VALUES
(NULL,'Assam',2569,'Assami');
```

And press enter and

```
Query OK, 1 row affected (0.01 sec)



Query OK, 1 row affected (0.00 sec)
```

will be displayed on the console screen and if you enter the command:

```
select*from states;
```

Then the above table will be displayed on the screen.

If you enter the command:

```
select state, population from states;
```

Then

```
state           | population |


+----------------+------------
+

| Karnataka      |       2562 |


| assam          |      25695 |
```

will be displayed on the console screen. And if you enter the command:

```
select state from states;
```

Then

```
| state          |


+----------------
+

| Karnataka      |


| assam
```

will be displayed on the console screen.

If you enter the command:

```
select*from states where language ='kannada';
```

Then

```
+----+-----------+------------+----------+-----------
+

| id | state     | population | language |

+----+-----------+------------+----------+-----------
+

|  1 | Karnataka |       2562 | kannada
```

will be displayed on the console screen. Similarly, if you enter the command:

```
select*from states where id =2;
```

Then

```
+----+-------+------------+----------+---------
+

| id | state | population | language |

+----+-------+------------+----------+---------
+
```

```
|  2 | assam |      25695 | assami
```

will be displayed on the console screen.

- **SQL and & or command**

If you enter the command:

```
select*from states where population =2566 and language ='kannada';

or

select*from states where population = 22666 or language = 'kannada';
```

Then

```
+----+-----------+------------+----------
+

| id | state     | population | language |


+----+-----------+------------+----------
+

|  1 | Karnataka |       2566 | Kannada  |


+----+-----------+------------+----------+
```

will be displayed on the console screen.

If you enter the command:

```
select *from states where population = 2566 or language ='assami';
```

```
+----+-----------+------------+----------
+

| id | state     | population | language |


+----+-----------+------------+----------
+

|  1 | Karnataka |       2566 | Kannada  |


|  2 | assam     |      22666 | assami   |
```

- **HOW to insert information into the table**

If you enter the command:

```
INSERT INTO states (id, state, population, language) VALUES (NULL, 'tamil nadu',
288,'tamil');
```

Then

```
+----+------------+------------+----------
+

| id | state      | population | language |


+----+------------+------------+----------
+

|  1 | Karnataka  |       2566 | Kannada  |


|  2 | assam      |      22666 | assami   |


|  3 | tamil nadu |        288 | tamil    |


+----+------------+------------+----------+
```

will be displayed on the console screen.

- **UPDATE INFORMATION**

If you enter the command:

```
update states set language =' telagu', population = 1 where state ='Karnataka';
```

Then

```
Query OK, 1 row affected (0.01 sec)


Rows matched: 1  Changed: 1  Warnings: 0
```

will be displayed on the console screen. And if you enter the command:

```
select*from states;
```

Then

```
+----+------------+------------+----------
+

| id | state      | population | language |


+----+------------+------------+----------
+

|  1 | Karnataka  |          1 |  telagu  |


|  2 | assam      |      22666 | assami   |


|  3 | tamil nadu |        288 | tamil    |


+----+------------+------------+----------
+

3 rows in set (0.00 sec)
```

will be displayed on the console screen.

- **DELETE information**

If you enter the command:

```
delete from states where language ='assami' and state ='assam';
```

Then

```
Query OK, 1 row affected (0.00 sec)
```

will be displayed on the console screen. And if you enter the command:

```
select*from states;
```

Then

```
+----+------------+------------+----------
+

| id | state      | population | language |


+----+------------+------------+----------
+

|  1 | Karnataka  |          1 |  telagu  |


|  3 | tamil nadu |        288 | tamil    |


+----+------------+------------+----------
+

2 rows in set (0.00 sec)
```

will be displayed on the console screen.

- **How to delete database in MYSQL**

If want to delete database "**dbtest**" from MySQL. Then you have to enter the command:

```
drop database dbtest;
```

Then

```
Query OK, 1 row affected (0.00 sec)
```

will be displayed on the console screen stating that database "**dbtest**" is deleted from MySQL.

If want to delete table "states" from database "**dbtest**". Then you have to enter the command:

```
drop table states;
```

Then

```
Query OK, 1 row affected (0.00 sec)
```

will be displayed on the console screen stating that table "states" is deleted from database "**dbtest**".

- **Limit Data Selection From MySQL Database**

If enter the command:

```
select*from states limit 1;
```

Then

```
id | state      | population | language |
```

```
+----+------------+------------+----------
+

|  1 | Karnataka  |       2566 | Kannada  |
```

will be displayed on the console screen.

If enter the command:

```
select*from states limit 2;
```

Then

```
id | state      | population | language |


+----+------------+------------+----------
+

|  1 | Karnataka  |       2566 | Kannada  |


|  2 | assam      |      22666 | assami   |
```

will be displayed on the console screen.

- **TRUNCATE**

If you enter the command:

```
truncate states;
```

Then

```
Query OK, 0 rows affected (0.06 sec)
```

will be displayed on the **console screen** stating that all the rows are removed from the table "states". And you can confirm it by entering the command:

```
select*from states;
```

Then:

```
Empty set (0.01 sec)
```

will be displayed on the console screen.

Some of The Most Important SQL Commands

- SELECT - extracts data from a database
- UPDATE - updates data in a database
- DELETE - deletes data from a database
- INSERT INTO - inserts new data into a database
- CREATE DATABASE - creates a new database
- ALTER DATABASE - modifies a database
- CREATE TABLE - creates a new table
- ALTER TABLE - modifies a table
- DROP TABLE - deletes a table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

Artificial Intelligence	Machine Learning	Deep Learning
A technology that enables human thinking capability and behavior in machine to perform **human-like tasks** on its own	Method of teaching machines to learn from the past data and past experiences to **identify patterns** and make predictions with minimal human intervention	The process of using **Artificial Neural Networks** (algorithms modeled to work like the human brain) to solve complex problems

Deep neural network

Consists of **3 Layers**

- **Input Layer:** receives the input
- **Hidden Layer:** carry out the analysis
- **Output Layer:** Display the desired result

Data masking

A data security technique that replaces **confidential data** with structurally similar bogus data

Synthetic data: Artificially generated data that is not based on any real-world observations and is used to train **ML models** when actual data is unavailable

Data perturbation: A data security technique that protects **sensitive information** from unauthorized users by adding unwanted information to a database

Python

Advantages over other Programming Languages

- Less Code
- Prebuilt Libraries
- Simple syntax and Open source
- Platform Independent
- Huge community of users

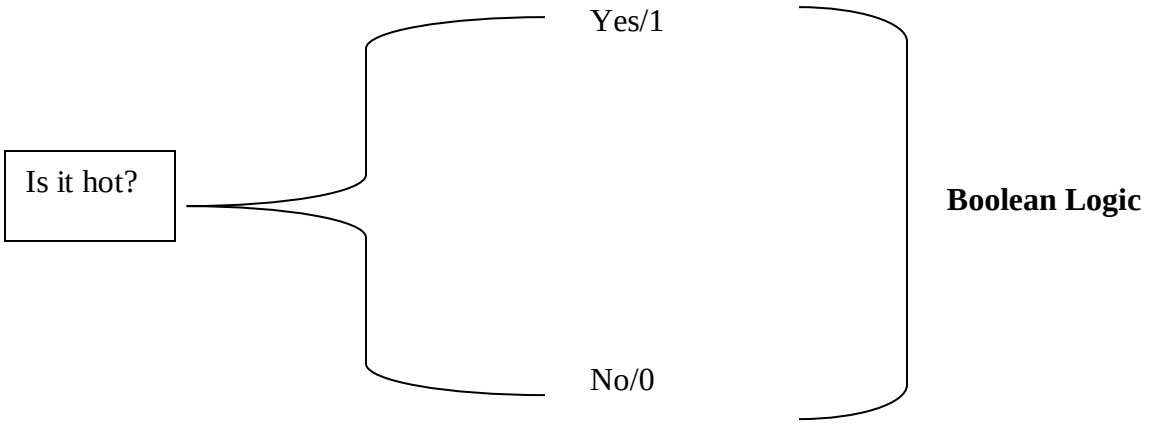
Data Loss Prevention

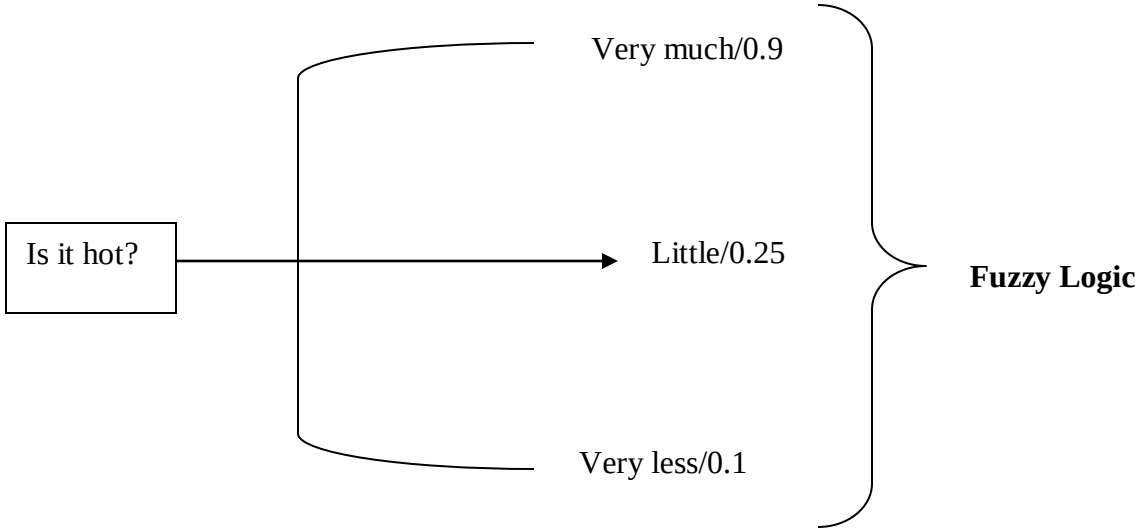
The process of detecting and preventing the misusage, unauthorized usage and **unwanted destruction** of sensitive data

Python Library	Used for
Tensorflow	performing heavy computations that involve Neural Networks
Scikit-Learn	working with complex data
NumPy	computing mathematical data
Theano	computing mathematical expressions involving multi-dimensional arrays
Keras	computing models, evaluating data-sets and visualizing graphs
Natural Language ToolKit	Natural Language Processing, text analysis, and text mining

ML Library	Purpose
Numpy	Mathematical Computation
Pandas	Tabular Data handling
Scikit Learn	Data Modeling and Data Preprocessing
Statsmodels	Time-Series Analysis
Regular Expressions	Text processing
Natural Language ToolKit	
Tensorflow	Deep Learning
Pytorch	

Cognitive Computing → the use of computational models to simulate human reasoning and human behavior





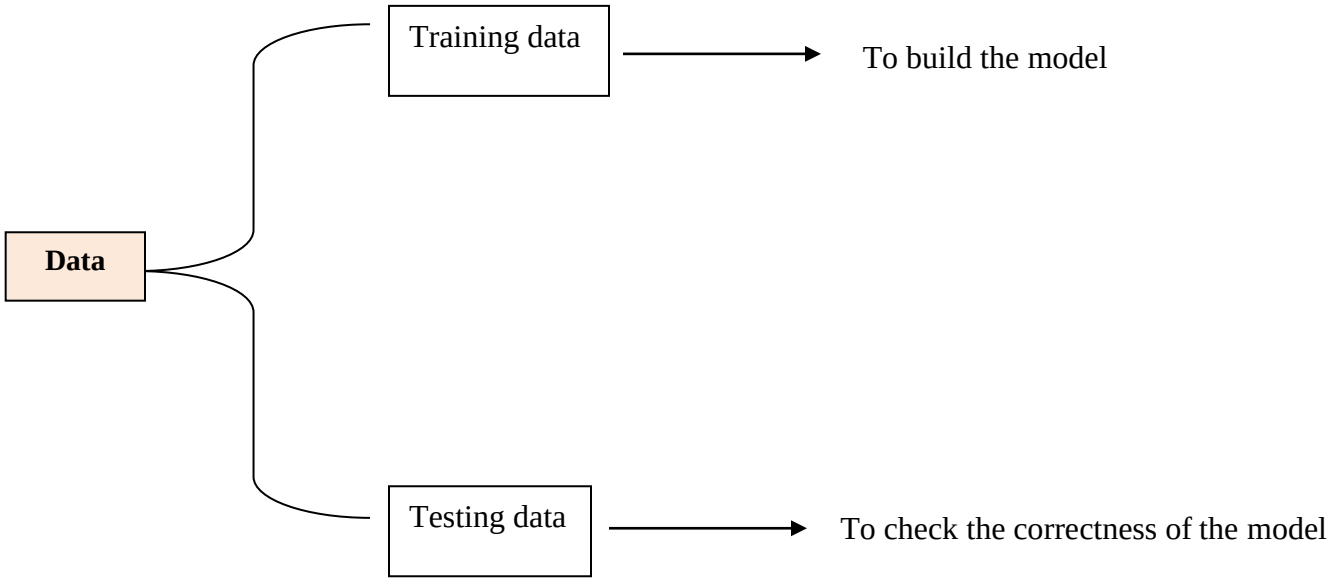
3 limitations of ML that led to the emergence of Deep Learning:

- Unable to process high dimensional data
- Manual Feature engineering
- Unable to process image data sets

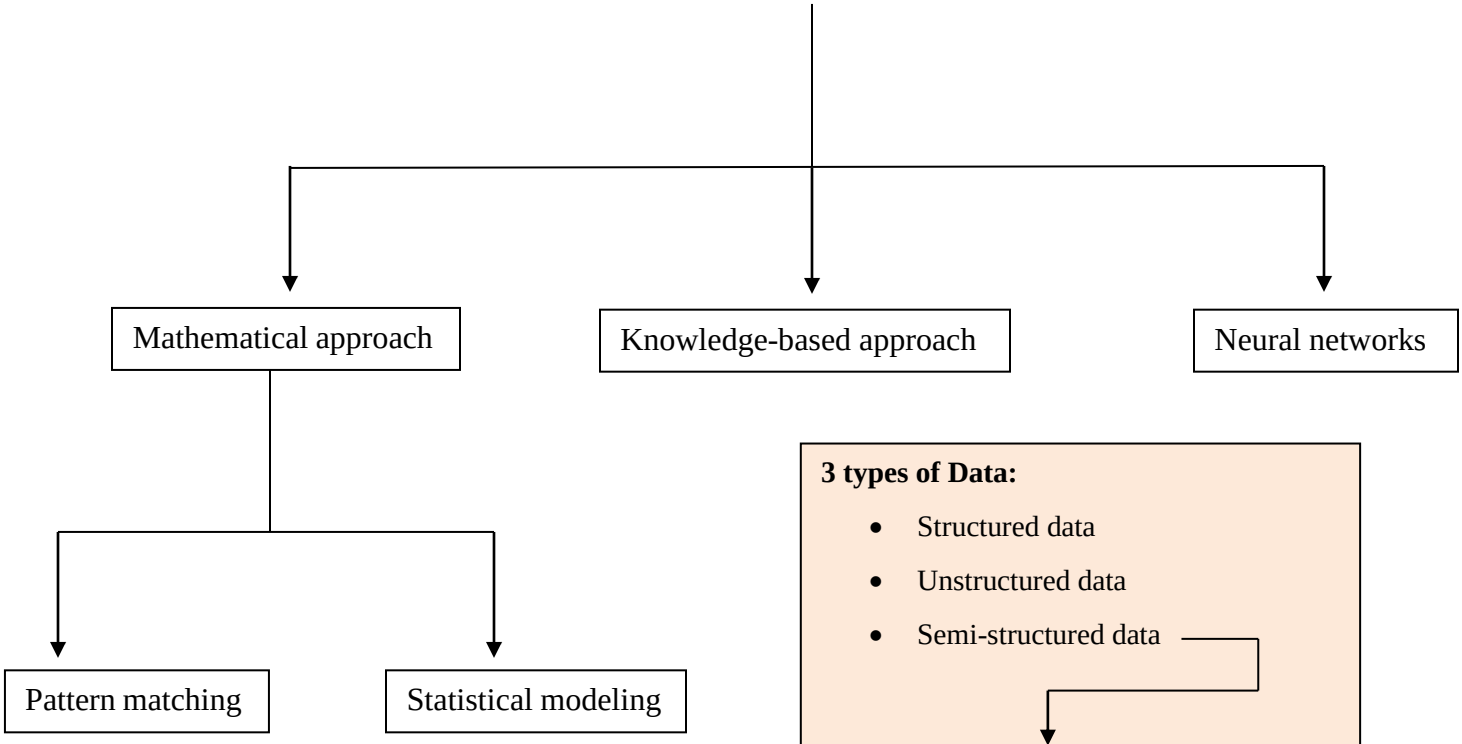
Pattern Recognition

- Learn from the data
- Recognize patterns from different angles and shapes

Applications of patterns recognition are voice recognition, weather forecast and object detection in images.



Pattern Classification Models



3 types of Data:

- Structured data
- Unstructured data
- Semi-structured data

(Structured data + Unstructured data)

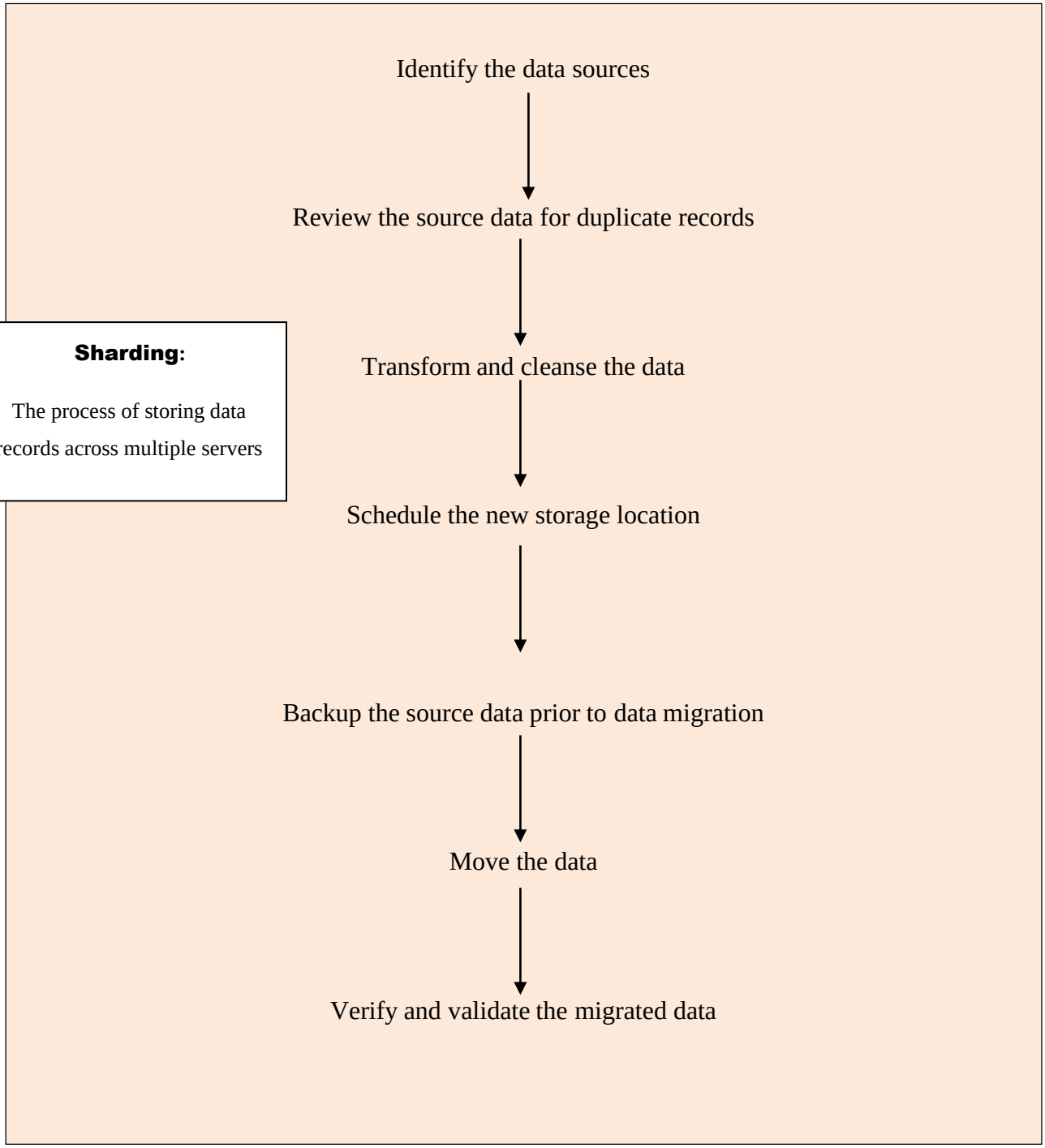
Feature Vector:

A set of numerical characteristics of observable phenomena that are arranged and provided to the Machine Learning model that makes a prediction

Real-time machine learning applications:

Applications that work with real-time data feeds to produce decisions that must be made quickly, including those needed by consumer applications, fraudulent prediction, recommendation systems, and proactive maintenance, to mention a few.

Data migration



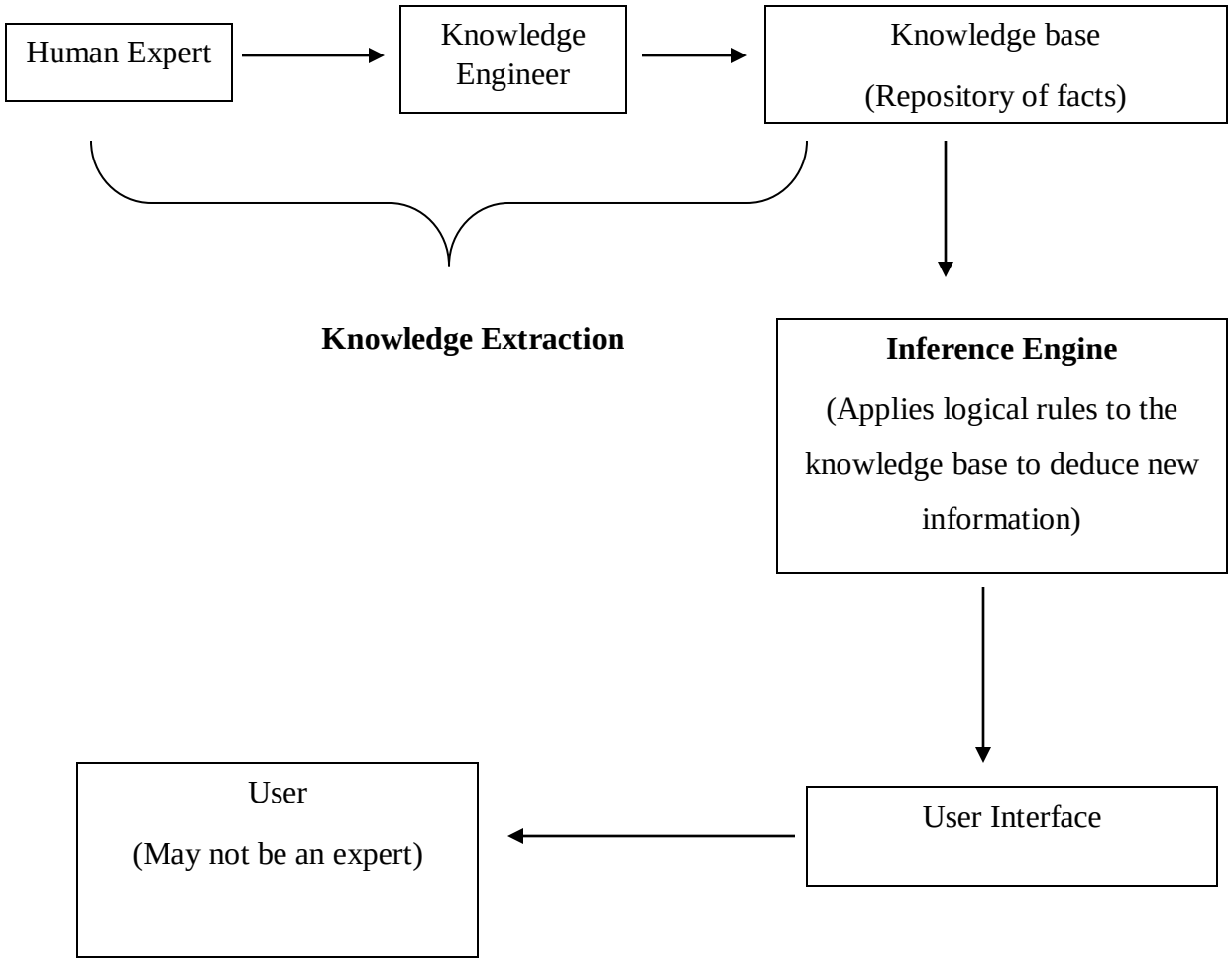
Sharding:

The process of storing data records across multiple servers

Data Visualization:

Collect data → prepare data → visualize → generate report

Expert System → computer-based decision making system that is designed to solve complex problems using both Knowledge base facts and heuristics.



Knowledge engineer → a technical person who integrates knowledge into computer systems.

Perl Program:

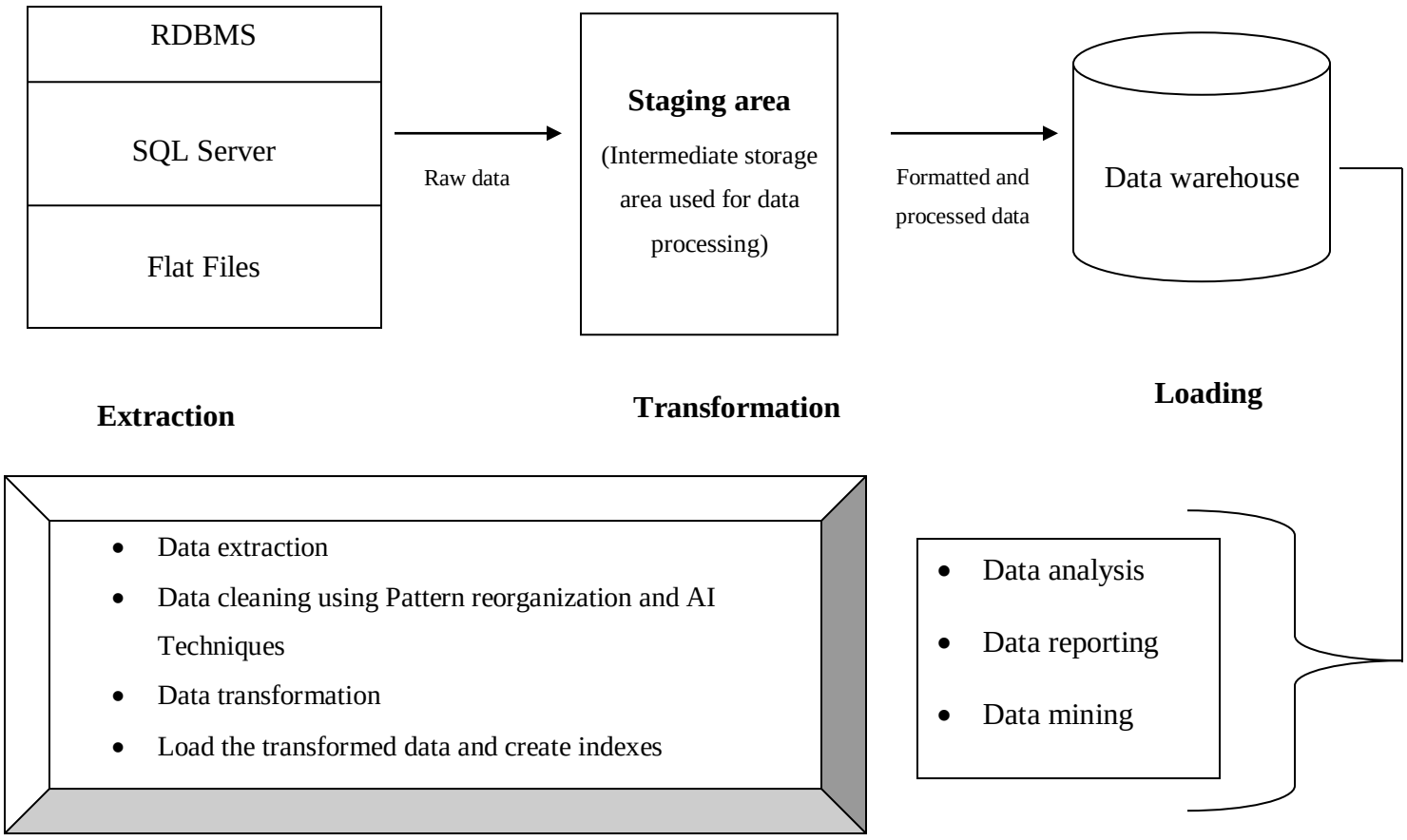
```
$x = 2;
until( $x > 10 ) {
   printf "$x\n";
   $x = $x + 1;
}
```

Output

```
2
3
4
5
6
7
8
9
10
```

ETL in Data warehousing

DATA SOURCES



Python code:

```
x = "Albert"

print(type(x) == str)

# Output: True

print(isinstance(x, str))

# Output: True
```

```
x = 1

print(type(x) == int)

# Output: True

print(isinstance(x, int))

# Output: True
```

```
x = 1.1

print(type(x) == float)

# Output: True

print(isinstance(x, float))

# Output: True
```

Falsy value = False, 0, '', []

```
print(0 or 1)

# Output: 1

print(False or 'Albert')

# Output: Albert

print('Albert' or 'Alan')

# Output: Albert

print([] or False)

# Output:  False

print(False or [])

# Output: []
```

If the value of the first operand = **Falsy value**: **"or"** used in an expression returns the value of second operand

```
print(0 and 1)

# Output: 0

print(False and 'Albert')

# Output: False

print('Albert' and 'Alan')

# Output: Alan

print([] and False)

# Output: []

print(False and [])

# Output: False
```

If the value of the first operand = **Falsy value**: **"and"** used in an expression returns the value of first operand

Web Data integration

- Identify the URL where the data is located
- Extract the data
- Transform and cleanse the extracted data
- Integrate the prepared data with a library of APIs to support seamless integrations with other systems
- Analyze and visualize the data with graphs and charts to find answers and obtain insights.

3 types of Data Models:

- **Conceptual Data Model** (the needs and requirements of the database)
- **Representational Data Model** (logical part of the database)
- **Physical Data Model** (physical structure of the database)

Cassandra-Cqlsh is a Python-based query language that enables users to:

- Define a schema
- Insert a data
- Execute a query

Basic MongoDB Commands

`db.help()`	get a list of commands
`show dbs`	print a list of all databases on the server
`use myTestDB`	create new database "**myTestDB**"
`db`	know your current selected database
`db.dropDatabase()`	drop the current selected database

`db.createCollection("Employee")`	create new collection "**Employee**"
`show collections`	print a list of all collections created
`db.Employee.drop()`	drop the collection "**Employee**"
`db.Employee.insert({name: 'Raj', address: 'Bangalore'})`	insert document in collection "**Employee**"
`db.Employee.find()`	list the documents in collection "**Employee**"
`{ "_id" : ObjectId("60658a0dbe02cfa1d386ab52"), "name" : "Raj", "address" : "Bangalore" }`	
`db.Employee.update({'name' : 'Raj'}, {$set: {'name' : 'Albert'}})`	update the document in collection "**Employee**"
`db.Employee.find()`	list the documents in collection "**Employee**"
`{ "_id" : ObjectId("60658a0dbe02cfa1d386ab52"), "name" : "Albert", "address" : "Bangalore" }`	
`db.Employee.save({"_id": new ObjectId("60658a0dbe02cfa1d386ab53"), name: "Newton", address: "Delhi"});`	save document in collection "**Employee**"
`db.Employee.find()`	list the documents in collection "**Employee**"
`{ "_id" : ObjectId("60658a0dbe02cfa1d386ab52"), "name" : "Albert", "address" : "Bangalore" }` `{ "_id" : ObjectId("60658a0dbe02cfa1d386ab53"), "name" : "Newton", "address" : "Delhi" }`	
`db.Employee.remove({'name': 'Albert'})`	delete document in collection "**Employee**"
`db.Employee.find()`	list the documents in collection "**Employee**"
`{ "_id" : ObjectId("60658a0dbe02cfa1d386ab53"), "name" : "Newton", "address" : "Delhi" }`	
`db.getUsers();`	list down all the users of current database
`show roles`	list down all the roles
`db.Employee.dataSize()`	get the size of the collection "**Employee**"
`db.Employee.storageSize()`	get the total size of document stored in the collection "**Employee**"
`db.Employee.totalSize()`	get the total size in bytes for both collection data and indexes
`db.Employee.totalIndexSize()`	get the total size of all indexes in the collection "**Employee**"

Python

Paradigm	Multi-paradigm: functional, imperative, object-oriented, structured, reflective
Designed by	Guido van Rossum
Developer	Python Software Foundation
First appeared	1990; 30 years ago
Stable release	3.8.3 / 13 May 2020; 32 days ago
Preview release	3.9.0b3 / 9 June 2020; 5 days ago
Typing discipline	Duck, dynamic, gradual (since 3.5)
OS	Linux, macOS, Windows Vista (and newer) and more
License	Python Software Foundation License
Filename extensions	.py, .pyi, .pyc, .pyd, .pyo (prior to 3.5), .pyw, .pyz (since 3.5)
Website	www.python.org

Guido van Rossum

(Creator of Python Language)

Major implementations
CPython, PyPy, Stackless Python, MicroPython, CircuitPython, IronPython, Jython, RustPython
Dialects
Cython, RPython, Starlark
Influenced by
ABC, Ada, ALGOL 68, APL, C, C++, CLU, Dylan, Haskell, Icon, Java, Lisp, Modula-

3, Perl, Standard ML
Influenced
Apache Groovy, Boo, Cobra, CoffeeScript, D, F#, Genie, Go, JavaScript, Julia, Nim, Ring, Ruby, Swift

Python is a popular, very powerful high-level language (like **C, C++, Perl, and Java** – and its name is derived from "**Monty Python's Flying Circus**" – **a British television series**), object- oriented programming scripting language ideally designed for rapid prototyping of **complex applications** by Dutch programmer "**Guido van Rossum**" in the early 1990s (often referred to as a "**glue**" language, meaning that it is capable to work in mixed-language environment) – which has simple syntax similar to the English language and is easy to understand, easy to use, write, modify and debug and flexible and easy to implement and run onopen source operating systems like Linux, **Windows**, Macintosh, Solaris, **FreeBSD**, OS/2, Amiga, AROS, AS/400 and is employed to perform automated testing of applications (i.e., to execute tests of applications, report outcomes and compare results with earlier test runs) and to increase the effectiveness and speed of software testing and its other commercial uses include financial applications, educational software, games, production of special effects for suchmovies as The Phantom Menace and The Mummy Returns, and business software. And python might not be the **best choice** for building the following types of applications and systems: Graphics-intensive applications, such as action games – where performance is important (because it is true that **CODE WRITTEN IN** Python, C# and visual basic etc. is far slower thanthe same code write in **C++.** Hence, **C++** is necessarily preferred). You should be very careful while working on python code. Indentation in python takes the center-stage. You cannot write a loop or a conditional statement without using indentation. There is a popular saying that code inpython should be as **Guido** indented it not how he intended it.

Python Comments:

```
# This is a single line comment

print("Albert Einstein")
```

Output:

Albert Einstein

```
''' This a multiple
Line
Comment'''

print("Albert Einstein")
```

```
"""This is a
Multiple line
Comment"""

print("Albert Einstein")
```

Python Variables:

```
p = 15
q = "Turing"
print(p)
print(q)
```

Output:

15

Turing

```
p, q, r = "Albert", "Alan", "John"
print(p)
print(q)
print(r)
```

Output:

Albert

Alan

John

```
p = q = r = "Albert"
print(p)
print(q)
print(r)
```

Output:

Albert

Albert

Albert

```
p = "Einstein"
print("Albert " + p)
```

Output:

Albert Einstein

```
x = "Turing"
print(f"Alan {x}")

x = "Turing"
print("Alan {}".format(x))
```

Output:

Alan Turing

```
p = "Alan "

q = "Turing"

r = p + q

print(r)
```

Output:

Alan Turing

```
p = 15

q = 20

print(p + q)


# Output:

                                35
```

```
print(bool(6<8)*30)

# Output: 30
```

```
p = 25

q = "Albert"

print(p + q)
```

```
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

```
print("Albert", end=" ")

print("Einstein")

print("1905", end="-")

print("papers")
```

Output:

Albert Einstein

1905-papers

```
p = "Hawking"

def myfunc():

   print("Stephen " + p)

myfunc()


# Output:

                Stephen Hawking
```

```
p,q,*r=[21,22,23,24,25]

print(p)

print(q)

print(r)


# Output:
```

```
21

22

[23, 24, 25]
```

```
try:
    raise ValueError('Albert', 'Einstein')
except ValueError as x:
    print(x.args)


# Output: ('Albert', 'Einstein')
```

```
# print the data type of the variable x

p = 25

print(type(p))
```

Output:

<class 'int'>

```
Jan = Mar = May = Jul = Aug = Oct = Dec = 31

Apr = Jun = Sep = Nov = 30

Feb = 28

x = Jan + Feb + Mar + Apr + May + Jun + Jul + Aug + Sep + Oct + Nov + Dec

print(x)
```

Output:

365

Value of y	Data Type
y = "Albert Einstein"	str
y = 50	int
y = 50.5	float
y = 5j	complex
y = ["albert", "alan", "hawking"]	list
y = ("albert", "alan", "mary")	tuple
y = range(16)	range
y = {"name" : "Mary", "age" : 106}	dict
y = {"albert", "alan", "john"}	set

y = frozenset({"alan", "john", "jerry"})	frozenset
y = True	bool
y = b"Albert"	bytes
y = bytearray(5)	bytearray

`y = memoryview(bytes(5))`	`memoryview`

Python Numbers

```
# Display a random number between 1 and 8:

import random

print(random.randrange(1, 9))


# Output:

                                        5
```

```
print(4**3**2**1)

# Output: 262144
```

Python Casting

```
p = int(6)

q = float(5.8)

r = str("albert")

print(p)

print(q)

print(r)
```

Output:

6

5.8

albert

Python Strings

```
x = "Mary"

print(x)

# Output: Mary
```

```
import math

print(math.pow(20,2))
```

→ 400.0

```
print(pow(20,2))
```

→ 400

```
x = """Mary"""

print(x)

# Output: Mary


x = '''Mary'''

print(x)

# Output: Mary
```

```
# The ASCII value of 'A' is 65

print(ord("A"))

# Output: 65

print(chr(65))

# Output: A
```

Python Booleans

```
print(20 > 19)

print(20 == 19)

print(20 < 19)
```

Output:

True

False

False

Python Operators

- Python Arithmetic Operators

a = 20

b = 2

Operator	Name	Example	Output
+	Addition	print(a + b)	22
-	Subtraction	print(a - b)	18

*	Multiplication	print(a * b)	40
/	Division	print(a / b)	10.0
%	Modulus	print(a % b)	0
**	Exponentiation	print(a ** b)	400
//	Floor division	print(a // b)	10

```
# 19/ 2 = 9.5

# the floor division // rounds 9.5 to 9

a = 19

b = 2

print(a // b)
```

```
# prints the numpy version

import numpy

print(numpy.__version__)


# Output: 1.16.2
```

- Python Assignment Operators

Operator	Example	Same As	Output
=	a = 15 **print(a)**	a =15 **print(a)**	15
+=	a+= 3 **print(a)**	a= a + 3 **print(a)**	18

-=	a -= 3 **print(a)**	a = a - 3 **print(a)**	12
*=	a *= 3 **print(a)**	a = a * 3 **print(a)**	45
/=	a /= 3 **print(a)**	a = a / 3 **print(a)**	5.0
%=	a %= 3 **print(a)**	a = a % 3 **print(a)**	0
//=	a //= 3 **print(a)**	a = a // 3 **print(a)**	5
**=	a **= 3 **print(a)**	a = a ** 3 **print(a)**	3375
&=	a &= 3 **print(a)**	a = a & 3 **print(a)**	3
|=	a |= 3 **print(a)**	a = a | 3 **print(a)**	15
^=	a ^= 3	a = a ^ 3	12

	print(a)	print(a)	
>>=	a >>= 3 **print(a)**	a = a >> 3 **print(a)**	1
<<=	a <<= 3 **print(a)**	a= a << 3 **print(a)**	120

- Python Comparison Operators

```
a = 13
b = 12
```

Operator	Name	Example	Output
==	Equal	print(a == b)	False
!=	Not equal	print(a != b)	True
>	Greater than	print(a > b)	True
<	Less than	print(a < b)	False
>=	Greater than or equal to	print(a >= b)	True
<=	Less than or equal to	print(a <= b)	False

- Python Logical Operators

```
a = 2
```

Operator	Description	Example	Output
and	Returns True if both statements are true	print(a < 5 and a < 10)	True
or	Returns True if one of the statements is true	print(a < 5 or a < 1)	True
not	Returns False if both statements are true	print(not(a < 5 and a < 10))	False

- Python Identity Operators

```
a = 3
b = 4
```

Operator	Description	Example	Output
is	Returns True if both variables are the same	print(a is b)	False
is not	Returns True if both variables are not the same	print(a is not b)	True

```
print("{:.2f}".format(99.452))
# Output: 99.45


print(pow(2,2))
# Output: 4
```

```
x = pow(20,2,3)
print(x)


x = pow(20,2)%3
print(x)
```

Output:

1

- Python Membership Operators

```
a = ["albert", "alan"]
```

Operator	Description	Example	Output
in	Returns True if **b** is present in **a**	b ="john" print(b in a)	False
not in	Returns True if **b** is not present in **a**	b ="john" print(b not in a)	True

- Python Bitwise Operators

```
a = 10
b = 2
```

Operator	Description	Example	Output
&	Bitwise AND	print(a & b)	2
|	Bitwise OR	print(a | b)	10
~	Bitwise NOT	print(~a)	-11
^	Bitwise XOR	print(a ^ b)	8
>>	Bitwise right shift	print(a>>2)	2
<<	Bitwise left shift	print(a<<2)	40

Bitwise AND

a	b	a & b
0	0	0
0	1	0
1	0	0
1	1	1

```
a = 10 = 1010 (in Binary)
b = 2 =  0010 (in Binary)
       1010
    &  0010
    ---------
       0010    = 2 (in Decimal)
    ---------
```

Bitwise OR

a	b	a | b
0	0	0
0	1	1
1	0	1
1	1	1

```
a = 10 = 1010 (in Binary)
b = 2 =  0010 (in Binary)
       1010
    |  0010
    ---------
       1010    = 10 (in Decimal)
    ---------
```

Bitwise XOR

a	b	a ^ b
0	0	0
0	1	1
1	0	1
1	1	0

```
a = 10 = 1010 (in Binary)
b = 2 =  0010 (in Binary)
       1010
    ^  0010
    ---------
       1000    = 8 (in Decimal)
    ---------
```

Bitwise NOT

a	b	~a	~b
0	0	1	1
0	1	1	0
1	0	0	1
1	1	0	0

```
a = 10 = 1010 (in Binary)
~a = ~1010
   = -(1010 + 1)
   = -(1011)
   = -11 (Decimal)
```

Bitwise left shift

```
a = 10 = 1010 (in Binary)
a<<1 = 1010<<1
     = 10100
     = 20 (in Decimal)
```

The binary number 1010 is appended with one zero at the end

```
a<<2 = 1010<<2
     = 101000
     = 40 (in Decimal)
```

The binary number 1010 is appended with two zeros at the end

Bitwise right shift

```
a = 10 = 1010 (in Binary)

a>>1 = 1010>>1
     = 101
     = 5 (in Decimal)

a>>2 = 1010>>2
     = 10
     = 2 (in Decimal)
```

The last one digit is removed from binary number 1010

The last two digits are removed from binary number 1010

```
x=6+4j
y=3+2j
print(x*y)
# Output: (10+24j)

x=6+4j
y=3+2j
print(x+y)
# Output: (9+6j)
```

```
x=6+4j
y=3+2j
print(x/y)
# Output: (2+0j)
```

```
x=6+4j
y=3+2j
print(x-y)
# Output: (3+2j)
```

```
iseven=lambda i:i%2==0
x=[50,42,43,80,70]
print(all(iseven(i) for i in x))
print(any(iseven(i) for i in x))
# Output:
False
True
```

```
isodd=lambda i:i%2!=0
x=[50,42,43,80,70]
print(all(isodd(i) for i in x))
print(any(isodd(i) for i in x))
```

Output:

False

True

```
iseven=lambda i:i%2==0
print(iseven(102))
# Output: True

iseven=lambda i:i%2==0
print(iseven(101))
# Output: False
```

```
isodd=lambda i:i%2!=0
print(isodd(101))
# Output: True

isodd=lambda i:i%2!=0
print(isodd(102))
# Output: False
```

Python Lists

```
x = ["alan", "albert", "john"]

print(x)

# Output: ['alan', 'albert', 'john']


x = ["alan", "albert", "john"]

print(x[1])

# Output: albert


x = ["alan", "albert", "john"]

print(x[-1])

# Output: john


x = ["alan", "albert", "john", "mary", "james", "joseph"]

print(x[2:5])

# Output: ['john', 'mary', 'james']


x = ["alan", "albert", "john", "mary", "james", "joseph"]

print(x[:4])

# Output: ['alan', 'albert', 'john', 'mary']


x = ["alan", "albert", "john", "mary", "james", "joseph"]

print(x[2:])

# Output: ['john', 'mary', 'james', 'joseph']
```

```
x=(6+4j)*(3-2j)/(3+2j)*(3-2j)

print(x)

# Output: (10-24j)


x=(6+4j)/(3+2j)

print(x)

# Output: (2+0j)
```

```
x = ["alan", "albert", "john", "mary", "james", "joseph"]

print(x[-4:-1])

# Output: ['john', 'mary', 'james']
```

```
x = ["alan", "albert", "john", "mary", "james", "joseph"]

for i in x:

    print(i)

# Output:

                          alan

                          albert

                          john

                          mary

                          james

                          joseph
```

```
x=zip('alan')

print(list(x))

# Output: [('a',), ('l',), ('a',), ('n',)]
```

Python code to get the home directory:

```
import os

print(os.path.expanduser('~'))

# Output: C:\Users\Manju
```

```
x = ["alan", "albert", "john", "mary", "james", "joseph"]

if "alan" in x:

  print("Yes, 'alan' is in this list")

# Output: Yes, 'alan' is in this list
```

```
x = ["alan", "albert", "john", "mary", "james", "joseph"]

x[1] = "david"

print(x)

# Output: ['alan', 'david', 'john', 'mary', 'james', 'joseph']
```

```
# prints the length of the List
x = ["alan", "albert", "john", "mary", "james", "joseph"]
print(len(x))
# Output: 6


x = ["alan", "albert", "john", "mary", "james", "joseph"]
x.append("david") # add an item to the end of the list
print(x)
# Output: ['alan', 'albert', 'john', 'mary', 'james', 'joseph', 'david']


x = ["alan", "albert", "john", "mary", "james", "joseph"]
x.insert(1, "david") # insert an item as the second position
print(x)
# Output: ['alan', 'david', 'albert', 'john', 'mary', 'james', 'joseph']


x = ["alan", "albert", "john", "mary", "james", "joseph"]
x.remove("alan") # remove an item from the list
print(x)
# Output: ['albert', 'john', 'mary', 'james', 'joseph']


x = ["alan", "albert", "john", "mary", "james", "joseph"]
x.pop()
print(x)
# Output: ['alan', 'albert', 'john', 'mary', 'james']


x = ["alan", "albert", "john", "mary", "james", "joseph"]
del x[0]
print(x)
```

Output: ['albert', 'john', 'mary', 'james', 'joseph']

```
x = ["alan", "albert", "john", "mary", "james", "joseph"]

del x

print(x)

# This will cause an error [NameError: name 'x' is not defined]

# because we have successfully deleted "x".



x = ["alan", "albert", "john", "mary", "james", "joseph"]

x.clear()

print(x)

# Output: []
```

```
import math

print(math.ceil(4.34)) # Output: 5

print(math.ceil(4.66)) # Output: 5
```

```
x = ["alan", "albert", "john", "mary", "james", "joseph"]

x = x.copy() # copy of a list with the copy() method

print(x)

# Output: ['alan', 'albert', 'john', 'mary', 'james', 'joseph']



x = ["alan", "albert", "john", "mary", "james", "joseph"]

x = list(x) # copy of a list with the list() method

print(x)

# Output: ['alan', 'albert', 'john', 'mary', 'james', 'joseph']



x = ["A", "B", "C"]

y = [11, 12, 13]

z = x + y

print(z)
```

```
Output: ['A', 'B', 'C', 11, 12, 13]
```

```
x = ["A", "B", "C"]
y = [11, 12, 13]
for i in y:   # append y into x
  x.append(i)
print(x)
# Output: ['A', 'B', 'C', 11, 12, 13]
```

```
import math
print(math.floor(4.34)) # Output: 4
print(math.floor(4.66)) # Output: 4
```

```
x = ["A", "B", "C"]
y = [11, 12, 13]
x.extend(y) # add y at the end of x using extend() method
print(x)
# Output: ['A', 'B', 'C', 11, 12, 13]
```

```
x = list(("A", "B", "C"))
print(x)
# Output: ['A', 'B', 'C']
```

```
x = list(("A", "B", "C"))
x.reverse()
print(x)
# Output: ['C', 'B', 'A']
```

```
x="james"
y=list(range(6))
z=(12,22,32,42,52)
b=zip(x, y, z)
print(list(b))
# Output:
[('j', 0, 12), ('a', 1, 22), ('m', 2, 32), ('e', 3, 42), ('s', 4, 52)]
```

```
x=['albert', 'einstein']
y=[19,15]
z=zip(x,y)
print(list(z))
# Output: [('albert', 19), ('einstein', 15)]
```

```
print([x for x in range(10,50) if x% 10 == 0])
# Output: [10, 20, 30, 40]
```

```
x=['A', 'B', 'C']
y=[21,22,23]
z=zip(x,y)
b=list(z)
p,q=zip(*b)
print(p,q)
# Output: ('A', 'B', 'C') (21, 22, 23)
```

```
print(round(4.34))
print(round(4.66))
# Output:
4
5
```

```
print((lambda x, y, z: (x+y)**z) (3,2,4))
# Output: 625
```

Python Tuples

```
x = ("albert", "alan", "mary")
print(x)
# Output: ('albert', 'alan', 'mary')
```

```
x = ("albert", "alan", "mary")
print(x[1])
# Output: alan
```

```
x = ("albert", "alan", "mary")
print(x[-1])
# Output: mary
```

```
x = ("A", "B", "C", "D", "E", "F", "G")
print(x[2:5])
# Output: ('C', 'D', 'E')
```

```
x = ("A", "B", "C", "D", "E", "F", "G")
print(x[-4:-1])
# Output: ('D', 'E', 'F')
```

```
x = ("A", "B", "C", "D", "E", "F", "G")
a = list(x)
a[1] = "Z"
x = tuple(a)
print(x)
# Output: ('A', 'Z', 'C', 'D', 'E', 'F', 'G')
```

```
x = ("A", "B", "C")

for i in x:

  print(i)

# Output:

             A

             B

             C

x = ("albert", "alan", "james")

if "james" in x:

   print("Yes, 'james' is in this tuple")

# Output: Yes, 'james' is in this tuple


# prints the length of the Tuple

x = ("alan", "albert", "john", "mary")

print(len(x))

# Output: 4
```

```
from os.path import exists as file_exists

print(file_exists('1.txt'))
```

OR

```
import pathlib

file=pathlib.Path('1.txt')

print(file.exists())
```

If the 1.txt file exists:

Output: True

Otherwise:

Output: False

```
# You cannot add items to a tuple

x = ("albert", "alan", "mary")

x[3] = "james"

print(x)
```

Error

```
x = ("alan",)

print(type(x)) # Output: <class 'tuple'>

y = ("alan")

print(type(y)) # Output: <class 'str'>
```

```
x = ("albert", "alan", "joseph")

del x

print(x)
```

```
# this will raise an error [NameError: name 'x' is not defined]
# because the tuple 'x' no longer exists
```

```
x = ("A", "B", "C")

y = (12, 22, 33)

z = x + y

print(z)

# Output: ('A', 'B', 'C', 12, 22, 33)
```

```
import time

print("Albert")

time.sleep(5)

print("Einstein") # this statement is executed after 5 seconds
```

```
x = tuple(("alan", "albert", "jenny"))

print(x)

# Output: ('alan', 'albert', 'jenny')
```

```
# prints the number of times a specified value occurs in a tuple

x = tuple(("alan", "albert", "jenny", "albert"))

print(x.count("albert"))

# Output: 2


# Searches the tuple for a specified value and prints the position of where it was found

x = tuple(("alan", "albert", "jenny", "james"))

print(x.index("albert"))

# Output: 1
```

Python Sets

```
x = {"alan", "john", "ram"}

print(x)

# Output: {'alan', 'john', 'ram'}
```

```
x = {"alan", "john", "ram"}

for i in x:

  print(i)

# Output:

                    alan

                    ram

                    john
```

```
x = {"alan", "john", "ram"}

print("ram" in x)

# Output: True
```

```
x = {"alan", "john", "ram"}

x.add("albert")

print(x)

# Output: {'alan', 'ram', 'john', 'albert'}
```

```
# returns a set that contains all items from both sets

# except items that are present in both sets


x = {"albert", "alan", "david"}

y = {"mary", "david", "albert"}

z = x.symmetric_difference(y)

print(z)

# Output: {'alan', 'mary'}
```

```
# returns True if all items set y are present in set x


x = {"F", "G", "H", "C", "B", "A"}

y = {"A", "B", "C"}

z = x.issuperset(y)

print(z)

# Output: True
```

```
x = {"alan", "john", "ram"}

x.update(["david", "albert", "james"])

print(x)

# Output: {'alan', 'david', 'james', 'ram', 'john', 'albert'}
```

```
# prints the length of the set

x = {"alan", "john", "albert"}

print(len(x))

# Output: 3
```

```
x = set(("alan", "albert", "john"))

y = x.copy()

print(y)

# Output: {'alan', 'john', 'albert'}
```

```
x = {"alan", "john", "albert"}

x.remove("john")

print(x)

# Output: {'albert', 'alan'}
```

```
# prints a set that contains the items that only exist

# in set x and not in set y


x = {"albert", "john", "david"}

y = {"james", "david", "albert"}

z = x.difference(y)

print(z)

# Output: {'john'}
```

```
x = {"alan", "john", "albert"}

x.discard("john")

print(x)

# Output: {'alan', 'albert'}
```

```
x = {"alan", "john", "albert"}

i = x.pop()

print(i) # prints removed item

print(x) # prints the set after removal
```

Output:

john

{'albert', 'alan'}

```
x = {"alan", "john", "albert"}

x.clear()

print(x)

# Output: set()
```

Constituency parsing

A technique for breaking down sentences into constituent parts

```
x = {"alan", "john", "albert"}

del x

print(x)
```

```
# this will raise an error [NameError: name 'x' is not defined]
# because the set 'x' no longer exists
```

```
x = {"A", "B", "C"}

y = {11, 12, 13}

z = x.union(y)

print(z)

# Output: {13, 'B', 'C', 11, 12, 'A'}
```

```
x = {"A", "B", "C"}

y = {11, 12, 13}

x.update(y)

print(x)

# Output: {13, 'B', 'C', 11, 12, 'A'}
```

```
x = set(("alan", "albert", "john"))

print(x)

# Output: {'alan', 'john', 'albert'}
```

```
# returns True if no items in set x is present in set y


x = {"albert", "john", "david"}

y = {"mary", "joseph", "alan"}

z = x.isdisjoint(y)

print(z)

# Output: True
```

```
# returns True if all items in set x are present in set y


x = {"A", "B", "C"}

y = {"G", "H", "I", "C", "B", "A"}

z = x.issubset(y)

print(z)

# Output: True
```

```
x = 6

y = 7.4

z = "Turing"

print(x, y, z)

# Output: 6 7.4 Turing


x = 6

y = 7.4

z = "Turing"

print(x, y, z, sep=",")

# Output: 6,7.4,Turing
```

```
name = "Albert Einstein"

year = "1905"

print("{1} papers of {0}".format(name, year))


# Output: 1905 papers of Albert Einstein


x = '{:^20}'.format('Albert Einstein')

print(x)


# Output: Albert Einstein
```

```
print('{:+d}'.format(1905))

# Output: +1905
```

```
x = '{:{width}.{prec}f}'.format(2.71828, width=5, prec=2)

print(x)

# Output: 2.72
```

```
x = {'n1': 'Stephen', 'n2': 'William', 'n3': 'Hawking'}

print('{n1} {n3}'.format(**x))

# Output: Stephen Hawking


from datetime import datetime

print('{:%Y-%m-%d %H:%M}'.format(datetime(1879, 3, 14, 11, 18)))

# Output: 1879-03-14 11:18
```

Python Dictionaries

```
x = {

"name": "John",

"gender": "Male",

"year": 1965

}

print(x)

# Output: {'name': 'John', 'gender': 'Male', 'year': 1965}
```

```
x = {

"name": "John",

"gender": "Male",

"year": 1965

}

y = x["gender"]

print(y)

# Output: Male
```

```
x = {

"name": "John",

"gender": "Male",

"year": 1965

}

x["year"] = 2019

print(x)

# Output: {'name': 'John', 'gender': 'Male', 'year': 2019}
```

```
x = {

"name": "John",

"gender": "Male",

"year": 1965

}

y = x.get("gender")

print(y)
```

Output:

Male

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
for i in x:
  print(i)

# Output:
                    name
                    gender
                    year
```

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
for i in x:
  print(x[i])

# Output:
                    John
                    Male
                    1965
```

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
for i in x.values():
    print(i)

# Output:
                    John
                    Male
                    1965
```

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
for a, b in x.items():
    print(a, b)

# Output:
                    name John
                   gender Male
                    year 1965
```

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
if "gender" in x:
   print("Yes, 'gender' is one of the keys in the 'x' dictionary")
# Output: Yes, 'gender' is one of the keys in the 'x' dictionary


# prints the length of the dictionary
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
print(len(x))
# Output: 3


x = {
"name": "John",
"gender": "Male",
"year": 1965
}
x["job"] = "Doctor"
print(x)
# Output: {'name': 'John', 'gender': 'Male', 'year': 1965, 'job': 'Doctor'}
```

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
x.pop("year")
print(x)
# Output: {'name': 'John', 'gender': 'Male'}
```

```
x = {

"name": "John",

"gender": "Male",

"year": 1965

}

x.popitem()

print(x)

# Output: {'name': 'John', 'gender': 'Male'}
```

```
x = {

"name": "John",

"gender": "Male",

"year": 1965

}

del x["gender"]

print(x)
```

Output:

```
{'name': 'John', 'year': 1965}
```

```
x = {

"name": "John",

"gender": "Male",

"year": 1965

}

x.clear()

print(x)

# Output: {}
```

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
y = x.copy()
print(y)
# Output: {'name': 'John', 'gender': 'Male', 'year': 1965}
```

```
x = "Einstein"
i = x[6:]
print(f"Albert'{i}'")
# Output: Albert'in'
```

```
x = {
"name": "John",
"gender": "Male",
"year": 1965
}
y = dict(x)
print(y)
# Output: {'name': 'John', 'gender': 'Male', 'year': 1965}
```

```
a = "    albert"
print(a)
print(a.lstrip())

# Output:
    albert
albert
```

```
x = "Einstein"
i = x[:3]
print(f"Albert'{i}'")
# Output: Albert'Ein'
```

```
mycolleagues = {

"colleague1" : {

"name" : "John",

"gender" : "Male"

},

"colleague2" : {

"name" : "Mary",

"gender" : "Female"

},

"colleague3" : {

"name" : "Jessey",

"gender" : "Female"

}

}

print(mycolleagues)
```

Output:

```
{'colleague1': {'name': 'John', 'gender': 'Male'}, 'colleague2': {'name': 'Mary',
    'gender': 'Female'}, 'colleague3': {'name': 'Jessey', 'gender': 'Female'}}
```

```
for x, y in enumerate(['Alan','John','Albert'], 100):

    print(x,y)

# Output:

                        100 Alan

                        101 John

                       102 Albert
```

```
x = "Einstein"

i = x[6:12]

print(f"Albert'{i}'")

# Output: Albert'in'
```

```
x = dict( name="David", gender="Male", year=1965)

print(x)

# Output: {'name': 'David', 'gender': 'Male', 'year': 1965}



x = {'a', 'e', 'i', 'o', 'u' }

y = 'vowel'

a = dict.fromkeys(x, y)

print(a)

# Output: {'u': 'vowel', 'o': 'vowel', 'i': 'vowel', 'a': 'vowel', 'e': 'vowel'}
```

Python Lambda

```
y = lambda i: i + 15

print(y(2))

# Output: 17


y = lambda i, u: i * u

print(y(2, 4))

# Output: 8


y = lambda i, u, v: i + u + v

print(y(2, 8, 4))

# Output: 14
```

```
def myfunc(i):

    return lambda x : x * i

y = myfunc(4)

print(y(22))
```

Output:

88

Python Arrays

```
x = ["Albert", "John", "Mary"]

print(x)

# Output: ['Albert', 'John', 'Mary']
```

```
x = ["Albert", "John", "Mary"]

i = x[0]

print(i)

# Output: Albert
```

```
x = ["Albert", "John", "Mary"]

x[0] = "David"

print(x)

# Output: ['David', 'John', 'Mary']
```

```
# prints the length of the array

x = ["Albert", "John", "Mary"]

print(len(x))

# Output: 3
```

```
x = ["Albert", "John", "Mary"]

for i in x:

   print(i)
```

Output:

Albert

John

Mary

```
x = ["Albert", "John", "Mary"]

x.append("Henry")

print(x)

# Output: ['Albert', 'John', 'Mary', 'Henry']
```

```
x = ["Albert", "John", "Mary"]

x.remove("Mary")

print(x)

# Output: ['Albert', 'John']
```

```
x = ["Albert", "John", "Mary"]

x.pop(2)

print(x)

# Output: ['Albert', 'John']
```

```
x = ["Albert", "John", "Mary"]

x.clear()

print(x)

# Output: []
```

```
a = "ALBERT EINSTEIN"

print(a.swapcase())

# Output: albert einstein


b = "albert einstein"

print(b.swapcase())

# Output: ALBERT EINSTEIN
```

```
x = ["Albert", "John", "Mary"]

x.reverse()

print(x)

# Output: ['Mary', 'John', 'Albert']


# prints the number of times the value "Albert" appears in 'x' list

x = ["Albert", "John", "Mary", "Albert"]

print(x.count("Albert"))

# Output: 2


# prints the position of the value "Mary" in 'x' list

x = ["Albert", "John", "Mary", "David"]

print(x.index("Mary"))

# Output: 2


x = ["Albert", "John", "Mary", "David"]

x.insert(1, "Joseph")

print(x)
```

Output:

['Albert', 'Joseph', 'John', 'Mary', 'David']

Python Classes

```
class MyClass:

    a=16

i = MyClass()

print(i.a)

# Output: 16
```

Python Iterators

```
x = ("Albert", "Alan", "Joseph")

i = iter(x)

print(next(i))

print(next(i))

print(next(i))
```

```
# Output:

Albert

Alan

Joseph
```

```
x = "ALAN"

i = iter(x)

print(next(i))

print(next(i))

print(next(i))

print(next(i))
```

```
# Output:

A

L

A

N
```

```
x = "ALAN"

for i in x:

  print(i)
```

```
# Output:

A

L

A

N
```

Python Math

```
a = min(15, 20, 35)

b = max(15, 20, 35)

print(a)

print(b)

# Output:

      15

      35


import math

a= math.sqrt(4)

print(a)

# Output: 2.0
```

```
import math

# prints the arc cosine value of 0.44

print(math.acos(0.44)) # Output: 1.1151976533990733

# prints the inverse hyperbolic cosine value of 5

print (math.acosh(5)) # Output: 2.2924316695611777

# prints the arc sine value of 0.44

print(math.asin(0.44)) # Output: 0.4555986733958234

# prints the inverse hyperbolic sine value of 5

print (math.asinh(5)) # Output: 2.3124383412727525
```

```
import math

# convert angle from a radian value to degree

print (math.degrees(6.702)) # Output: 383.9963142966777

# convert angle from a degree value to radian

print (math.radians(383.9963142966777)) # Output: 6.702
```

```
import math

# prints the arc tangent value of 0.44

print (math.atan(0.44)) # Output: 0.41450687458478597

# prints the inverse hyperbolic tangent value of 0.5

print (math.atanh(0.5)) # Output: 0.5493061443340549

# prints the arc tangent value of 8/2 in radian

print(math.atan2(8, 2)) # Output: 1.3258176636680326

# prints the cosine value of 0.00

print (math.cos(0.00)) # Output: 1.0

# prints the hyperbolic cosine value of 1

print (math.cosh(1)) # Output: 1.5430806348152437

# prints the error function value of 0.59

print (math.erf(0.59)) # Output: 0.5959364971979085

# prints the complementary error function value of 0.59

print (math.erfc(0.59)) # Output: 0.4040635028020914

# prints the exponential value of 56

print(math.exp(56)) # Output: 2.091659496012996e+24

# prints the absolute value of -55.68

print(math.fabs(-55.68)) # Output: 55.68

# prints the factorial of 4

print(math.factorial(4)) # Output: 24

# prints the remainder of 11/2

print(math.fmod(11, 2)) # Output: 1.0
```

```
import math

# prints the natural logarithm of 3.8278

print(math.log(3.8278)) # Output: 1.3422902256218356

# prints the base-10 logarithm of 3.8278

print(math.log10(3.8278)) # Output: 0.5829492381002341

# prints the base-2 logarithm of 3.8278

print(math.log2(3.8278)) # Output: 1.9365154519383503


# prints the greatest common divisor of 2 and 4

print(math.gcd(2, 4)) # Output: 2

# prints the tangent value of 90

print(math.tan(90)) # Output: -1.995200412208242

# prints the sine value of 10

print(math.sin(10)) # Output: -0.5440211108893698


# prints the log gamma value of 6

print(math.lgamma(6)) # Output: 4.787491742782047
```

```
import math


# check whether the 2 values are close to each other or not

print(math.isclose(1.533, 1.3566)) # Output: False

print(math.isclose(1.533, 1.533)) # Output: True
```

```
import math

# prints the natural logarithm of (1 + 3.5456)

print(math.log1p(3.5456)) # Output: 1.5141597321177864
```

```
x=9

i=3

print(x*(2**i))
```

```
import math

print(math.ldexp(9, 3))
```

Output:

72.0

```
import math

print(math.trunc(5.55))

print(math.trunc(6.36))

print(math.trunc(-100.78))
```

Output:

5

6

-100

```
import math

# prints the value of (11 + 12 + 13 + 14 + 15)

print(math.fsum([11, 12, 13, 14, 15])) # Output: 65.0
```

```
import math

Perpendicular = 20

Base = 10

# prints the hypotenuse of a right-angled triangle

print(math.hypot(Perpendicular, Base)) # Output: 22.360679774997898
```

```
import math

# check whether a value is 'Not a Number' or not

print (math.isnan(16)) # Output: False

print (math.isnan(math.inf)) # Output: False

print (math.isnan(float("nan"))) # Output: True

print (math.isnan(float("inf"))) # Output: False

print (math.isnan(float("-inf"))) # Output: False

print (math.isnan(math.nan)) # Output: True
```

```
import math

# Check whether a value is finite or not

print(math.isfinite(252))  # Output: True

print(math.isfinite(-15.15)) # Output: True

print(math.isfinite(+15.15)) # Output: True

print(math.isfinite(math.inf)) # Output: False

print(math.isfinite(float("nan"))) # Output: False

print(math.isfinite(float("inf"))) # Output: False

print(math.isfinite(float("-inf"))) # Output: False

print(math.isfinite(-math.inf)) # Output: False

print(math.isfinite(0.0)) # Output: True
```

Math Constants

```
import math


# prints the value of Euler's number

print (math.e)


# prints the value of positive infinity

print (math.inf)


# prints the value of negative infinity

print (-math.inf)


# prints the value of Pi

print (math.pi)


# tau = (the circumference of a circle / the radius of a circle) = 2Pi

# prints the value of tau

print (math.tau)
```

Output:

```
2.718281828459045

inf

-inf

3.141592653589793

6.283185307179586
```

Python JSON

```
import json

a = '{"name":"David", "age":30, "gender":"Male"}'

b = json.loads(a)

print(b["name"])

# Output: David
```

```
import json

a = '{"name":"David", "age":30, "gender":"Male"}'

print(json.dumps(a))

# Output: "{\"name\":\"David\", \"age\":30, \"gender\":\"Male\"}"
```

```
import json

a = {"name":"David", "age":30, "gender":"Male"}

print(json.dumps(a))

# Output: {"name": "David", "age": 30, "gender": "Male"}
```

Python RegEx

```
import re

# checks if the string 'x' starts with "Albert" and ends with "German"

x = "Albert Einstein was a German"

i = re.search("^Albert.*German$", x)

if i:

   print("Wow! We've found a match!")

else:

    print("No match")
```

Output:

Wow! We've found a match!

```
import re

# checks if the string 'x' starts with "Albert" and ends with "German"

x = "Albert Einstein was a German"

i = re.findall("^Albert.*German$", x)

if i:

   print("Wow! We've found a match!")

else:

    print("No match")
```

Output:

Wow! We've found a match!

```
import re

# splits the string 'x' at every white-space character

x = "Albert Einstein was a German"

print(re.split("\s", x))

# Output: ['Albert', 'Einstein', 'was', 'a', 'German']
```

```
import re

# replaces the word 'German' with the word "theoretical physicist"

x = "Albert Einstein was a German"

print(re.sub("German", "theoretical physicist", x))
```

Output:

```
Albert Einstein was a theoretical physicist
```

```
import re

x = "Albert Einstein grew up in Munich"

# prints all lower case characters alphabetically between "a" and "e"

print(re.findall("[a-e]", x))

# Output: ['b', 'e', 'e', 'e', 'c']
```

```
import re

x = "Albert Einstein grew up in Munich"

# prints all upper case characters alphabetically between "a" and "e"

print(re.findall("[A-E]", x))

# Output: ['A', 'E']
```

```
import re

x = "1905 Papers"

# prints all digit characters

print(re.findall("\d", x))

# Output: ['1', '9', '0', '5']
```

```
import re

x = "albert einstein"

print(re.findall("al...t", x)) # Output: ['albert']

print(re.findall("ein....n", x)) # Output: ['einstein']
```

```
import re

x = "albert einstein"

print(re.findall("al.*n", x)) # Output: ['albert einstein']

print(re.findall("ein.*n", x)) # Output: ['einstein']
```

```
import re

x = "albert einstein"

print(re.findall("al.+n", x)) # Output: ['albert einstein']

print(re.findall("ein.+n", x)) # Output: ['einstein']
```

```
import re

x = "alan"

y = "james"

print(re.findall("al.{1}n", x)) # Output: ['alan']

print(re.findall("ja.{2}s", y)) # Output: ['james']
```

```
import re

x = "Alan Turing was an English mathematician"

# Check if the string 'x' contains either "was" or "albert"

y = re.findall("was|albert", x)

print(y)

if y:

  print("Wow! We've found a match!")

else:

  print("No match")
```

Output:

['was']

Wow! We've found a match!

```
import re

x = "Stephen Hawking was a British scientist"

# Check if the string 'x' has any a, g or s characters

y = re.findall("[ags]", x)

print(y)

if y:

  print("Wow! We've found a match!")

else:

  print("No match")
```

Output:

['a', 'g', 'a', 's', 'a', 's', 's', 's']

Wow! We've found a match!

```
import re

x = "Stephen Hawking was a British scientist"

# Check if the string 'x' has any characters between a and c

y = re.findall("[a-c]", x)

print(y)

if y:

  print("Wow! We've found a match!")

else:

  print("No match")
```

Output:

['a', 'a', 'a', 'c']

Wow! We've found a match!

```
import re

x = "1905 Papers"

# Check if the string 'x' has any digits

y = re.findall("[0-9]", x)

print(y)

if y:

  print("Wow! We've found a match!")

else:

  print("No match")
```

Output:

['1', '9', '0', '5']

Wow! We've found a match!

```
import re

x = "a+b+c"

# Check if the string 'x' has any + characters

y = re.findall("[+]", x)

print(y)

if y:

  print("Wow! We've found a match!")

else:

  print("No match")
```

Output:

['+', '+']

Wow! We've found a match!

Python If ... Else

```
x = 25

y= 50

if y > x:

  print("x < y")

# Output: x < y
```

```
x = 25

y = 25

if x > y:

  print("x > y")

elif x == y:

  print("x=y")

# Output: x=y
```

```
x = 55

y = 25

if y > x:

  print("y>x")

elif x == y:

  print("x=y")

else:

  print("x>y")

# Output: x>y
```

```
x = 55

y = 25

if y > x:

  print("y>x")

else:

  print("x≠y")

# Output: x≠y
```

```
x = 55

y = 25

if y < x: print("y<x")

# Output: y<x
```

```
x = 55

y = 25

print("X") if y > x else print("Y")

# Output: Y
```

```
x = 25

y = 25

print("X") if x > y else print("=") if x == y else print("Y")

# Output: =
```

```
x = 50

y = 25

z = 100

if x > y and z > x:

  print("Both conditions are met")

# Output: Both conditions are met
```

```
x = 100

y = 50

z = 200

if x > y or x > z:

  print("At least one of the conditions is fulfilled")

# Output: At least one of the conditions is fulfilled
```

Python Loops

```
x = ["C++", "Java", "Python"]

for i in x:

  print(i)

  if i == "Java":

     break
```

Output:

C++

Java

```
x = ["C++", "Java", "Python"]
for i in x:
    if i == "Java":
      break
    print(i)
```

Output:

C++

```
x = ["C++", "Java", "Python"]
for i in x:
    if i == "Java":
      continue
    print(i)
```

Output:

C++

Python

```
for i in range(4):
   print(i)
# Output:
            0
            1
            2
            3
```

```
for i in range(3, 5):
  print(i)
Output:
                3
                4
```

```
for i in range(2, 8, 5):
   print(i)
# Output:
             2
             7
```

```
for i in range(3):
  print(i)
else:
  print("Completed successfully!")
```

Output:

0

1

2

Completed successfully!

```
i = ["Albert", "Alan"]

u = ["Engineer", "Doctor"]

for a in i:

 for b in u:

   print(a, b)
```

Output:

```
Albert Engineer

Albert Doctor

Alan Engineer

Alan Doctor
```

```
for i in [11, 12, 13]:

   pass

# would raise an error without the pass statement
```

```
x = 1

while x < 3:

  print(x)

  x += 1

Output:

             1

             2
```

```
x = 1

while x < 3:

 print(x)

 x += 1

else:

 print("x is no longer less than 3")
```

Output:

```
1

2

x is no longer less than 3
```

```
x = 0

while x < 4:

  x += 1

  if x == 2:

    continue

  print(x)
```

Output:

```
1

3

4
```

```
x = 0

while x < 4:

  x += 1

  if x == 2:

    break

  print(x)
```

Output:

```
1
```

Python User Input

```
x = input("Enter username:")

print("Username is: " + x)
```

```
Enter username:

If you enter the word "ram"

Username is: ram

will be outputted on the console screen
```

```
print "Hello World" # is acceptable in Python 2
print ("Hello World") # in Python 3, print must be followed by ()
```

Python Try Except

```
print(i) # would raise an error because i is not defined


try:  # would generate an exception because i is not defined
  print(i)
except:
  print("An exception occurred") # Output: An exception occurred
```

```
try: # would raise an NameError because i is not defined
  print(i)
except NameError:
  print("i is not defined")
except:
  print("Something else went incorrect") # Output: i is not defined
```

```
try: # would not generate any error
  print("Albert")
except:
  print("Something went incorrect")
else:
  print("Nothing went incorrect")
```

Output:

Albert

Nothing went incorrect

```
i = -6 # would raise an error and stop because i < 0
if i < 0:
  raise Exception("No numbers below zero acceptable.")


i = "Alan" # would raise a TypeError because i is not an integer
if not type(i) is int:
  raise TypeError("Only integer values are acceptable.")
```

Python String Formatting

```
year = 1905
name = "Einstein's"
x = "Albert {1} {0} Papers."
print(x.format(year, name))
# Output: Albert Einstein's 1905 Papers.
```

```
x = "{name} Einstein's {year} Papers."
print(x.format(name = "Albert", year = "1905"))
# Output: Albert Einstein's 1905 Papers.
```

```
i = 15
x = "The cost of the pen is: {:.4f} Rupees"
print(x.format(i)) # Output: The cost of the pen is: 15.0000 Rupees
```

```
x = True

print(("Albert", "Einstein")[x]) # Output: Einstein

x = False

print(("Albert", "Einstein")[x]) # Output: Albert
```

```
def myfunc():

    print("Albert Einstein")
if __name__ == '__main__':

    myfunc()

# Output: Albert Einstein
```

```
import sys

sys.stdout = open('1.txt','wt')

print("Albert Einstein")

print("Alan Turing")
```

A new file is created with name '1.txt' with following contents:

Albert Einstein

Alan Turing

```
print("Albert Einstein", file=open('1.txt','a'))

print("Alan Turing", file=open('1.txt','a'))
```

A new file is created with name '1.txt' with following contents:

Albert Einstein

Alan Turing

```
n = [2, 5, 7, 17]

a = map(lambda i: i%2, n)

for x in a:

   print(x)
```

Output:

0

1

1

1

```
months = ['jan', 'feb', 'mar', 'april', 'may', 'june']

x = filter(lambda x: x if len(x)==3 else '', months)

for i in x:

   print(i)

# Output:

                    jan

                    feb

                    mar

                    may
```

```
print((8-3)*(2-(1+1))) # Output: 0


A = 1,2,3,4

print(A) # Output: (1, 2, 3, 4)


B = 'a', 'b', 'c', 'd'

print(B) # Output: ('a', 'b', 'c', 'd')


x = [1,2,3,4]

print(x[3]) # Output: 4

print(x[5]) # would raise a IndexError: list index out of range


a = [11,12]

x = '+'.join(str(i) for i in a)

b = [21,22]

y = '*'.join(str(u) for u in b)

print(x,y) # Output: 11+12 21*22
```

```
print ("xyz" > "") # Output: True

print(" ".isspace() # Output: True


x = "Albert Einstein"

print(x.startswith("Ala")) # Output: False

print(x.endswith("ein")) # Output: True
```

```
x = ['albert', 'alan', 'james', 'mary', 'peter', 'john']

# prints a list of names that start with the letter 'a'

print([i for i in x if i[0] == 'a']) # Output: ['albert', 'alan']
```

```
print(sorted([250, 230, 240, 220, 210]))

# Output: [210, 220, 230, 240, 250]
```

```
f = open("1.txt","a+") # opens the 1.txt file in append mode

f.write("Albert Einstein") # appends content to the file

f.close()  # closes the file
```

```
import platform

print(platform.system()) # Output: Windows

print(platform.processor()) # Output: Intel64 Family 6 Model 78 Stepping 3, GenuineIntel

print(platform.architecture()) # Output: ('64bit', 'WindowsPE')

print(platform.machine()) # Output: AMD64

print(platform.node()) # Output: DESKTOP-68GI94H

print(platform.platform()) # Output: Windows-10-10.0.10586-SP0
```

Python Datetime

```
import time;

x = time.asctime(time.localtime(time.time()))

print ("Local current time:", x)

# Output: Local current time: Thu Jun 23 19:38:21 2022
```

```
import datetime

i = datetime.datetime.now()

# prints the year

print(i.year) # Output: 2022

# prints the full name of weekday

print(i.strftime("%A")) # Output: Thursday

# prints the short name of weekday

print(i.strftime("%a")) # Output: Thu

# prints the weekday as a number 0-6, 4 is Thursday

print(i.strftime("%w")) # Output: 4

# prints the day of month 01-31

print(i.strftime("%d")) # Output: 23

# prints the full name of month

print(i.strftime("%B")) # Output: June

# prints the short name of month

print(i.strftime("%b")) # Output: Jun

# prints the month as a number 01-12, 06 is June

print(i.strftime("%m")) # Output: 06

# prints the day number of year: 001-365

print(i.strftime("%j")) # Output: 174

# prints the local version of date and time

print(i.strftime("%c")) # Output: Thu Jun 23 20:20:39 2022
```

```
import datetime

i = datetime.datetime.now()

# prints the local version of date

print(i.strftime("%x")) # Output: 06/23/22

# prints the local version of time

print(i.strftime("%X")) # Output: 20:20:39

# prints AM/PM

print(i.strftime("%p")) # Output: PM

# prints the minute 00-59

print(i.strftime("%M")) # Output: 37

# prints the second 00-59

print(i.strftime("%S")) # Output: 10
```

```
# check the documentation of the print function

help(print)
```

```
class Science:

  def __init__(self, x = "Albert", y = 1905, z = "Relativity"):

    self.x = x

    self.y = y

    self.z = z

print(vars(Science()))


# Output: {'x': 'Albert', 'y': 1905, 'z': 'Relativity'}
```

```
import statistics

# prints the harmonic mean of the given data

print(statistics.harmonic_mean([40, 60, 80])) # Output: 55.38461538461538

# prints the average of the given data

print(statistics.mean([11, 12, 15, 17, 19, 14])) # Output: 14.666666666666666

# prints the median of the given data

print(statistics.median([11, 12, 15, 17, 19, 14])) # Output: 14.5

# prints the high median of the given data

print(statistics.median_high([11, 12, 15, 17, 19, 14])) # Output: 15

# prints the low median of the given data

print(statistics.median_low([11, 12, 15, 17, 19, 14])) # Output: 14

# prints the mode of the given data

print(statistics.mode([11, 12, 15, 11, 19, 14])) # Output: 11

# prints the standard deviation of the given data

print(statistics.stdev([11, 12, 15, 17, 19, 14])) # Output: 3.011090610836324

# prints the variance of the given data

print(statistics.variance([11, 12, 15, 17, 19, 14])) # Output: 9.066666666666666
```

```
import random

x = ["A", "B", "C"]

# prints a random element from a list 'x'

print(random.choice(x)) # Output: C
```

```
import random

x = ["A", "B", "C"]

# prints a list that contains any 2 of the items from the list 'x'

print(random.sample(x, k=2)) # Output: ['C', 'A']
```

```
import random

# prints a random number between and included 10 and 30

print(random.uniform(10, 30)) # Output: 23.223429438350152
```

```
import random

# prints a random number between and included 10 and 30 but most likely closer to 15

print(random.triangular(10, 30, 15)) # Output: 16.011782141402527
```

```
# import cmath for complex number operations

import cmath


# prints the arc cosine value of 5+6j

print (cmath.acos(5+6j)) # Output: (0.8800800926246654-2.7493465969740996j)

# prints the arc sine value of 5+6j

print (cmath.asin(5+6j)) # Output: (0.6907162341702312+2.7493465969740996j)

# prints the arc tangent value of 5+6j

print (cmath.atan(5+6j)) # Output: (1.488221988087583+0.09801052194400592j)

# prints the cosine value of 5+6j

print (cmath.cos(5+6j)) # Output: (57.21909818460074+193.4276431213065j)

# prints the sine value of 5+6j

print (cmath.sin(5+6j)) # Output: (-193.43002005693958+57.21839505634109j)

# prints the tangent value of 5+6j

print (cmath.tan(5+6j)) # Output: (-6.6852313902770184e-06+1.00001031089811 98j)
```

```
import numpy

x = numpy.array([50, 100, 150, 200])

print(x) # Output: [ 50 100 150 200]
```

```
import numpy as np

x = np.array([13, 9, 11, 15])

print(np.sort(x))

# Output: [ 9 11 13 15]
```

```
import numpy as np

x = np.array([50, 100, 150, 200])

print(x) # Output: [ 50 100 150 200]
```

```
import numpy as np

x = np.array([50, 100, 150, 200])

print(x[1]) # Output: 100

print(x[2] + x[3]) # Output: 350
```

```
import numpy as np

# splits the array 'x' into 3 parts

x = np.array([51, 52, 53, 54, 55, 56])

print(np.array_split(x, 3))

# Output: [array([51, 52]), array([53, 54]), array([55, 56])]
```

```
import numpy as np

x = np.array([500, 400, 300, 200])

print(x[[False, True, False, True]]) # Output: [400 200]
```

```
import numpy as np

# sort the array alphabetically

x = np.array(['james', 'mary', 'alan'])

print(np.sort(x)) # Output: ['alan' 'james' 'mary']
```

```
import numpy as np

x = np.array([[23, 22, 24], [25, 20, 21]])

print(np.sort(x))

# Output:

             [[22 23 24]

              [20 21 25]]
```

```
import numpy as np

x = np.array([31, 30, 29, 28, 27])

i = x.view()

x[0] = 32

print(x) # Output: [32 30 29 28 27]
```

```
import numpy as np

x = np.array([31, 32, 33, 34, 35, 36, 37])

print(x[1:5]) # Output: [32 33 34 35]

print(x[4:])  # Output: [35 36 37]

print(x[:4]) # Output: [31 32 33 34]

print(x[-3:-1]) # Output: [35 36]

print(x[1:5:2]) # Output: [32 34]

print(x[::2]) # Output: [31 33 35 37]
```

```
import numpy as np

x = np.array(['albert', 'james', 'mary'])

y = np.array([41, 42, 43, 44])

# prints the data type of arrays 'x' and 'y'

print(x.dtype) # Output: <U6

print(y.dtype) # Output: int64
```

```
import numpy as np

x = np.array([[31, 32, 33, 34, 35], [36, 37, 38, 39, 40]])

print(x[0:2, 1:4])

# Output:

             [[32 33 34]

              [37 38 39]]
```

```
import numpy as np

x = np.array([31, 30, 29, 28, 27])

i = x.copy()

x[0] = 32

print(x) # Output: [32 30 29 28 27]
```

```
import numpy as np

x = np.array([21, 22, 23])

y = np.array([24, 25, 26])

print(np.concatenate((x, y)))
```

Output:

```
[21 22 23 24 25 26]
```

Python code to print the current working directory:

```
import os

# prints the current working directory

print(os.getcwd()) # Output: C:\Users\Manju
```

Python code to create a directory:

```
import os

# get the current working directory
x = os.getcwd()
# create a directory named 'myfiles' in the current working directory
y = x + "/myfiles"


os.mkdir(path = y)

print("myfiles directory created successfully.")

# Output: myfiles directory created successfully.
```

```
import os

# get the current working directory

x = os.getcwd()

# create a directory named 'myfiles' in the current working directory

y = x + "/myfiles"

os.mkdir(path = y)

# prints the current working directory

print("Current working directory: ", os.getcwd())

# change our current working directory to the newly created directory 'myfiles'

os.chdir(y)

# verify that our working directory has been changed using getcwd()

print("New working directory: ", os.getcwd())
```

Output:

```
Current working directory:  C:\Users\Manju

New working directory:  C:\Users\Manju\myfiles
```

```
import os

# get the current working directory

x = os.getcwd()

# list all the files and directories present in the current working directory

print(os.listdir(path = x))
```

```
import os

# get the current working directory

x = os.getcwd()

y = "123.txt"

i = x + "/" + y

# list all the files and directories present in the current working directory

# before deletion of 123.txt

print("Files before deletion: ", os.listdir(path = x))

# delete the file '123.txt'

os.remove(path = i)

# list all the files and directories present in the current working directory

# after deletion of 123.txt

print("Files after deletion: ", os.listdir(path = x))
```

Set the path of the 123.txt file that we are going to delete

Pandas Series

```
import pandas as pd


x = pd.DataFrame([[25, 33], [47, 89]],
                 columns=['No. Rejected', 'No. Accepted'],
                 index=['1905 Papers', '1915 Papers'])
print(x)
# Output:
                           No. Rejected  No. Accepted
             1905 Papers            25            33
             1915 Papers            47            89
```

```
import pandas as pd

x= [12, 72, 22]

print(pd.Series(x))

# Output:
             0    12
             1    72
             2    22
             dtype: int64
```

```
import pandas as pd

x = [12, 72, 22]

i = pd.Series(x)

print(i[0])

# Output: 12
```

```
import pandas as pd

x= [12, 72, 22]

y = pd.Series(x, index = ["a", "b", "c"])

print(y["b"]) # Output: 72
```

```
import pandas as pd

x= [12, 72, 22]

y = pd.Series(x, index = ["a", "b", "c"])

print(y)

# Output:

              a    12

              b    72

              c    22

              dtype: int64
```

```
import pandas as pd
months = {"jan": 31, "feb": 28, "mar": 31}
print(pd.Series(months))

Output:

              jan    31

              feb    28

              mar    31

              dtype: int64
```

```
import pandas as pd

x = {

  "months": [1, 2, 3],

  "days": [31, 28, 31]

}

print(pd.DataFrame(x))
```

	months	days
0	1	31
1	2	28
2	3	31

```
x = ["12", "11", "15", "13", "14"]

x = [int(a) for a in x]

x.sort()

print(x)

# Output: [11, 12, 13, 14, 15]
```

```
print(4**0.5!= 2)

# Output: False
```

```
# creates a list of the first letters of every word in the given string

x = "Albert Einstein profoundly changed physics"

i = [a[0] for a in x.split()]

print(i) # Output: ['A', 'E', 'p', 'c', 'p']
```

```
# check if the length of a particular word in the given string is odd or even

x = "Albert Einstein"

for i in x.split():

    if len(i)%2 == 0:

        print("The length of the word " + i + " is even" + " because it has "+ str(len(i)) + " characters")

    elif len(i)%2 == 1:

        print("The length of the word " + i + " is odd" + " because it has "+ str(len(i)) + " characters")
```

Output:

```
The length of the word Albert is even because it has 6 characters

The length of the word Einstein is even because it has 8 characters
```

```
x = [11,12,[13,14,'Alan']]

x[2][2] = 'Albert'

print(x) # Output: [11, 12, [13, 14, 'Albert']]


print(5 ** 2) # Output: 25

print(25 ** 0.5) ) # Output: 5.0
```

```
print(divmod(10,2)) # Output: (5, 0)

print(divmod(15,5)) # Output: (3, 0)
```

```
# display the size of each item in the array

import numpy as np

x = np.array([[11,12,13]])

print(x.itemsize, "bytes") # Output: 4 bytes



# display the data type of each item in the array

import numpy as np

x = np.array([[11,12,13]])

print(x.dtype) # Output: int32
```

```
x = "Galileo Galilei"

for i in x:

    if i == "G":

        pass

        print(i)

    else:

        pass

# Output:

            G

            G
```

```
import numpy as np

# prints 6 values which are evenly spaced over the given interval 6-10

print(np.linspace(6, 10, 6))

# Output: [ 6.   6.8  7.6  8.4  9.2 10. ]
```

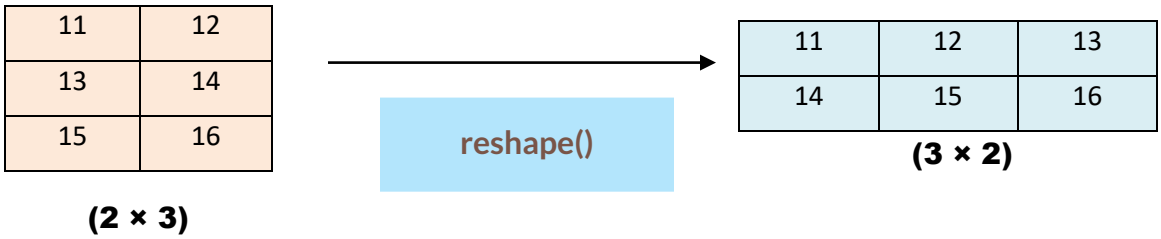
```
import numpy as np

x = np.array([[11,12,50],[20,25,8]])

y = np.array([[15,16,1],[32, 39, 35]])

print(np.vstack((x, y)))


print(np.hstack((x, y)))
```

Output:

```
[[11 12 50]
 [20 25  8]
 [15 16  1]
 [32 39 35]]
```

Output:

```
[[11 12 50 15 16  1]
 [20 25  8 32 39 35]]
```




```
import numpy as np

x = np.array([[11,12],[13,14],[15,16]])

print(x) # prints (2×3) array

"""

# Output:


[[11 12]

 [13 14]

 [15 16]]


"""

print(x.reshape(2,3)) # prints (3×2) array


"""

# Output:


[[11 12 13]

 [14 15 16]]


"""
```

```
import numpy as np

print(np.dot(2,4)) # Output: 8
```

```
print(2 * (6 + 10)) # Output: 32

print(2 * 6 + 10)  # Output: 22
```

```
print('Albert %s' %'Einstein') # Output: Albert Einstein

print('Albert {}'.format('Einstein'))  # Output: Albert Einstein
```

```
a=3

for x in range(a):

  print('+'*(a-x-1) + '*'*(2*x+1)+'+'*(a-x-1))

# Output:

             ++*++

             +***+

             *****
```

```
x=21

if sum(map(int, str(x))) % 2:

  print("odd")

else:

  print("even")

# Output: odd
```

```
a=2

x=a-1

while x>-1:

  print('+'*(a-x-1) + '*'*(2*x+1)+'@'*(a-x-1))

  x=x-1

# Output:

             ***

             +*@
```

```
x=5

for i in [i for i in range(2, x+1) if all(i % a!= 0 for a in range(2, i))]:

   print(i)

# Output:

             2

             3

             5
```

```
x="Einstein Albert"
print(" ".join(str.split(x, " ")[-1::-1]))
# Output: Albert Einstein


x = ((11, 14, 12), (19, 0, 13), (15, 16, 18))
print(x[2][1])
# Output: 16


x=3
for a in range(0, x, int(x**0.5)):
   print(a)
# Output:
             0
             1
             2
```

```
x = "15"
if x.isdigit():
  print("Digit")
else:
  print("String")
# Output: Digit
```

```
x = {1:['Albert', 1905], 2:['Newton', 1875], 3:['Alan', 1922]}
for i in x:
  print(i)
# Output:
             1
             2
             3
```

```
x="NUN"
# convert the string 'NUN' to lower case
# check whether the string is a palindrome or not
# if it is palindrome print 1 else 0
print(int(x.lower()==x.lower()[::-1])) # Output: 1
```

```
x = {11, 12, 13, 14}
y = {13, 14, 15, 16}
print(x ^ y) # Output: {11, 12, 15, 16}
```

```
x = [11, 13, 12, 12, 13, 14, 15, 12]

i = []

for a in x:

   if a not in i:

    i.append(a)

    print(i)
```

Output:

[11]

[11, 13]

[11, 13, 12]

[11, 13, 12, 14]

[11, 13, 12, 14, 15]

```
x = "15"

if x.isnumeric():

  print("Integer")

else:

  print("String")

# Output: Integer
```

```
x = "Albert"

if x.isnumeric():

  print("Integer")

else:

  print("String")

# Output: String
```

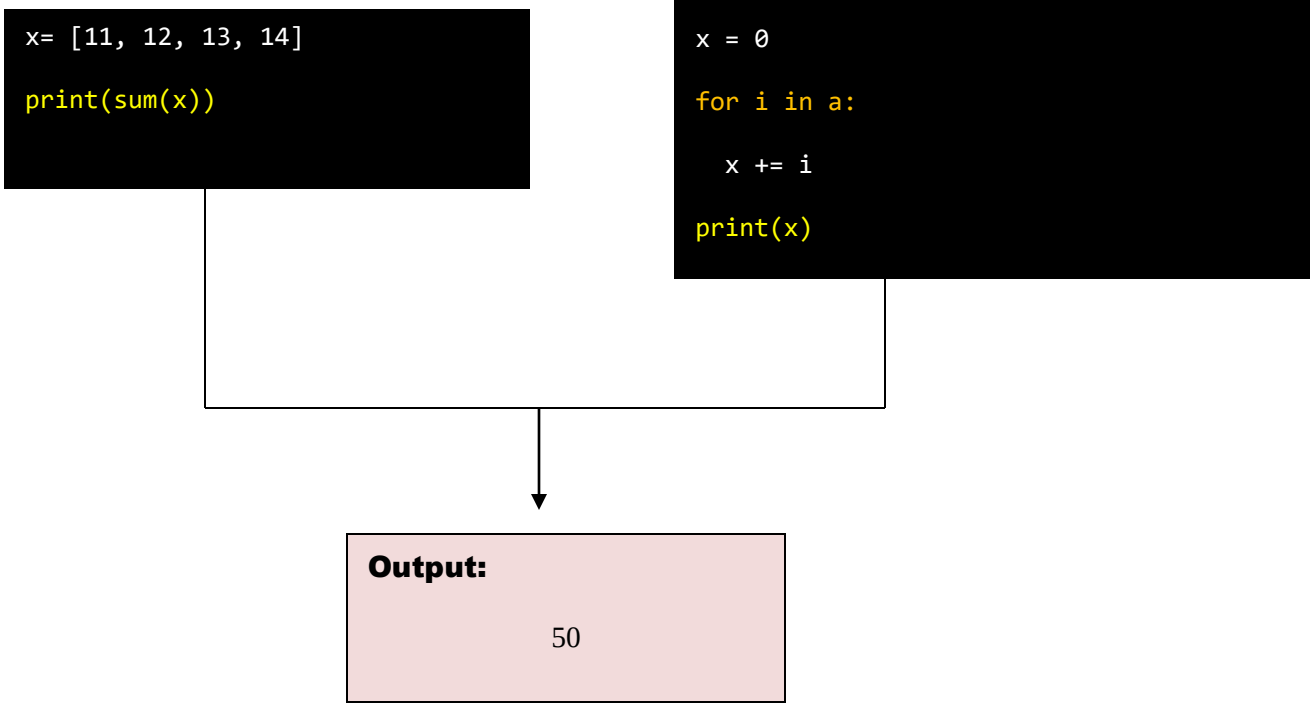
```
a = "john"

i = ""

for x in a:

  i = x + i

  print(i)

# Output:

             j

             oj

             hoj

             nhoj
```

```
x = "michaeljohn"

print(x[0].upper() + x[1:7] + x[7].upper() + x[8:])

# Output: MichaelJohn
```

```
exec('x = 6\ny=20\nprint("Sum", "of", x, "and", y, "=", x+y)')

# Output: Sum of 6 and 20 = 26
```

```
x= [11, 12, 13, 14]

print(sum(x))
```

```
a = [11, 12, 13, 14]

x = 0

for i in a:

  x += i

print(x)
```



Output:

50

```
x = [18,12,13,15,14,17]

# sort the list in ascending order

x.sort()

# prints second maximum element from a given list

print(x[-2]) # Output: 17

# prints second minimum element from a given list

print(x[1])  # Output: 13
```

```
x = [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

print(x[1 : : 2])  # Output: [12, 14, 16, 18, 20]
```

```
x = [12, 11, 12, 14, 15, 13, 16]

print(14 in x) # Output: True

print(20 in x) # Output: False
```

```
x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

print(x[1:-1:2]) # Output: [2, 4, 6, 8]
```

```
x = [19, 15, 'Papers']

for a in x:

   print(a, end=' ') # Output: 19 15 Papers


def myfunc(x):

   x.append(24)

x = [21, 22, 23]

print(x) # Output: [21, 22, 23]

myfunc(x)

print(x) # Output: [21, 22, 23, 24]
```

```
# prints the sum of digits from 1 to 5

print(sum(range(1,5))) # Output: 10
```

```
import pandas as pd

print(pd.DataFrame())
```

Output:

```
Empty DataFrame

Columns: []

Index: []
```

```
import pandas as pd

if pd.DataFrame().empty:

    print('Empty DataFrame')

else:

    print('DataFrame is not empty')

# Output: Empty DataFrame
```

```
import pandas as pd

if pd.DataFrame({'name': ["albert", "alan"], 'year': [1915, 1925]}).empty:

    print('Empty DataFrame')

else:

    print('DataFrame is not empty')
```

Output:

```
DataFrame is not empty
```

```
class Relativity:

  year = 1905

  name = 'Albert Einstein'


physicist = Relativity()


name = getattr(physicist, 'name')

print(name)


year = getattr(physicist, 'year')

print(year)
```

Output:

Albert Einstein

1905

```
x = [14, 10, 28, 37]

y = (14, 10, 28, 37)

print(isinstance(x, list)) # Output: True

print(isinstance(x, tuple)) # Output: False

print(isinstance(y, list)) # Output: False

print(isinstance(y, tuple)) # Output: True
```

```
x = set()

x.add(12)

x.add(14)

x.add(16)

print(x) # Output: {16, 12, 14}
```

```
i = 'print("Albert Einstein")\nprint("Newton")'

exec(compile(i, '', 'exec'))

# Output:

              Albert Einstein

              Newton
```

```
import numpy as np

# prints the Lowest Common Multiple of 14 and 16

print(np.lcm(14, 16)) # Output: 112
```

```
class Science:

    Relativity = 'E=mcsquared'

subject = 'Relativity'

object = Science

result = hasattr(object, subject)

print(f'Does the object {object} has attribute named "{subject}"? \n Ans. {result}')
```

Output:

Does the object <class '__main__.Science'> has attribute named "Relativity"?

Ans. True

```
class Science:

    Relativity = 'E=mcsquared'

subject = 'Gravitation'

object = Science

result = hasattr(object, subject)

print(f'Does the object {object} has attribute named "{subject}"? \n Ans. {result}')
```

Output:

Does the object <class '__main__.Science'> has attribute named "Gravitation"?

Ans. False

```
x = __import__('math', globals(), locals(), [], 0)

print(x.pow(2,3)) # Output: 8.0
```

```
import numpy as np

x = np.array([50, 25, 15, 10])

print(np.diff(x)) # Output: [-25 -10  -5]

# [-25 -10 -5] because (25-50) = -25, (15-25) = -10, and (10-15) = -5
```

```
import numpy as np

x = np.array([50, 25, 15, 10])

print(np.diff(x, n=2)) # Output: [15  5]



# [15  5] because:

# (25-50) = -25, (15-25) = -10, and (10-15) = -5

# -10 -(-25) = 15 and -5 -(-10) = 5
```

```
import numpy as np

# prints the Greatest Common Denominator of 6 and 9

print(np.gcd(12, 18)) # Output: 6
```

```
import numpy as np

x = np.array([11, 12, 13, 14])

y = np.array([15, 16, 17, 18])

print(np.prod([x, y])) # Output:1764322560

# 11 × 12 × 13 × 14 × 15 × 16 × 17 × 18 = 1764322560
```

```
import numpy as np

x = np.array([11, 12, 13, 14])

print(np.cumprod(x)) # Output: [ 11   132  1716 24024]

# [11, 11 × 12, 11 × 12 × 13, 11 × 12 × 13 × 14] = [ 11   132  1716 24024]
```

```
import pandas as pd

names = ['Einstein', 'Hawking', 'Planck', 'Darwin', 'Mendel']

is_Biologist = [False, False, False, True, True]

print(pd.DataFrame(zip(names, is_Biologist), columns=['names', 'is_Biologist']))

# Output:

      names    is_Biologist

0  Einstein         False

1   Hawking         False

2    Planck         False

3    Darwin          True

4    Mendel          True
```

```
import numpy as np

print(np.sin(np.pi/2)) # Output: 1.0

print(np.deg2rad(180)) # Output: 3.141592653589793
```

```
x = [11, 0, -12, 14, -13]

print([i**2 for i in x if i > 0]) # Output: [121, 196]
```

```
print(input("Albert Einstein")) # Output: Albert Einstein
```

```
def myfunc():

    yield "Albert Einstein"

print(next(myfunc())) # Output: Albert Einstein
```

```
def myFunc(*argv):
    for i in argv:
        print(i)
myFunc('Albert', 'Einstein')
# Output:
Albert
Einstein
```

```
# Fibonacci sequence of numbers: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34 . . .
# Python program to get the first 5 Fibonacci numbers
x, y = 0, 1
for i in range(0, 5):
    print(x)
    x, y = y, x + y
```

Output:

```
0
1
1
2
3
```

```
import numpy as np
x = np.array([2,6,5])
y = np.array([4,12,6])
z = np.sum((x,y), axis=0)
a = 0
for i in z:
    print("" + str(a) + ":", i)
    a = a + 1
```

Output:

```
0: 6
1: 18
2: 11
```

```
import numpy as np
x = np.array([2,6,5])
y = np.array([4,12,6])
z = np.sum((x,y), axis=1)
a = 0
for i in z:
    print("" + str(a) + ":", i)
    a = a + 1
```

Output:

```
0: 13
1: 22
```

```
def myFunc(**kwargs):

    for x, y in kwargs.items():

        print ("%s = %s" %(x, y))

myFunc(Name = 'Albert Einstein', Born = '14 March 1879', Died = '18 April 1955')

Output:

             Name = Albert Einstein

             Born = 14 March 1879

             Died = 18 April 1955
```

```
a = [[]]*6

a[0].append(3)

print(a) # Output: [[3], [3], [3], [3], [3], [3]]
```

```
def myfunc(i):

   months = ['jan','feb','mar','apr','may','jun','jul']

   yield months[i]

   yield months[i+1]

x = myfunc(0)

print(next(x))

print(next(x))
```

Output:

jan

feb

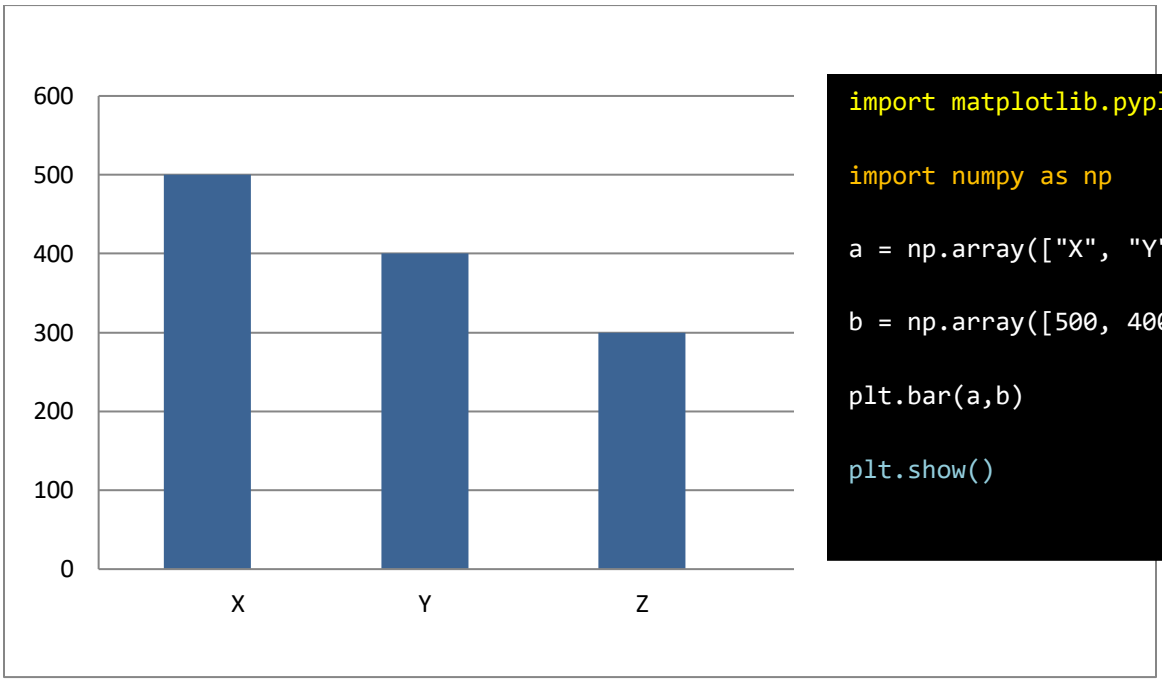
```
def myfunc(i):

   months = ['jan','feb','mar','apr','may','jun','jul']

   yield months[i]

   yield months[i+2]

x = myfunc(0)

print(next(x))

print(next(x))
```

Output:

jan

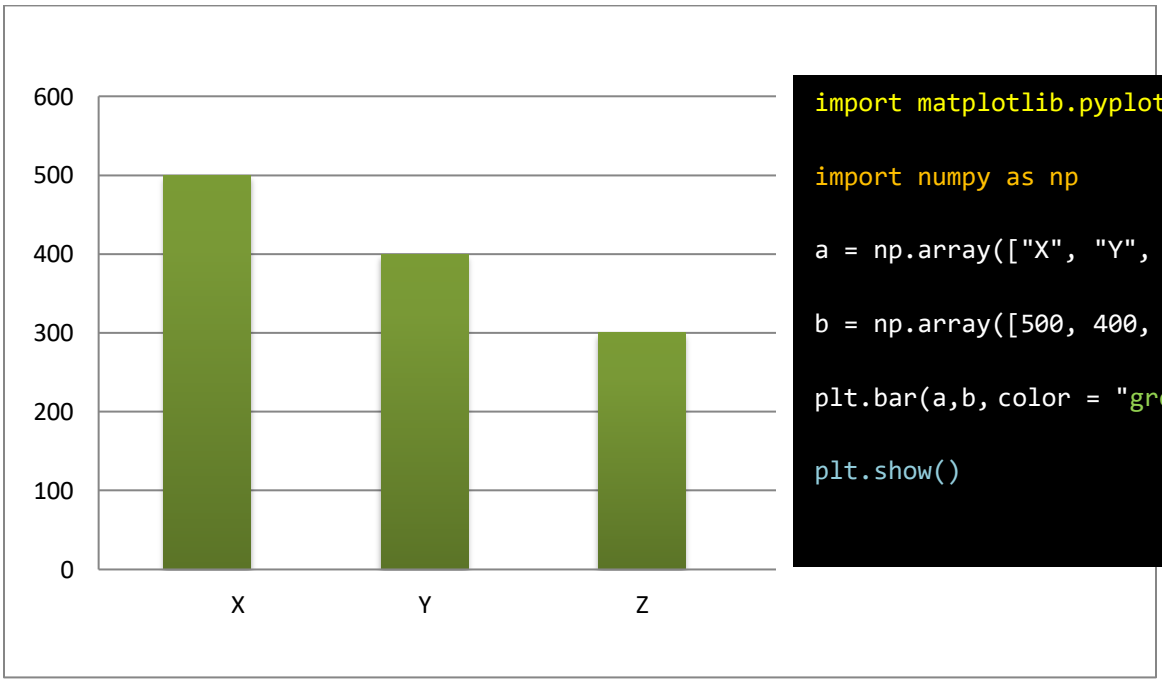
mar

Output:



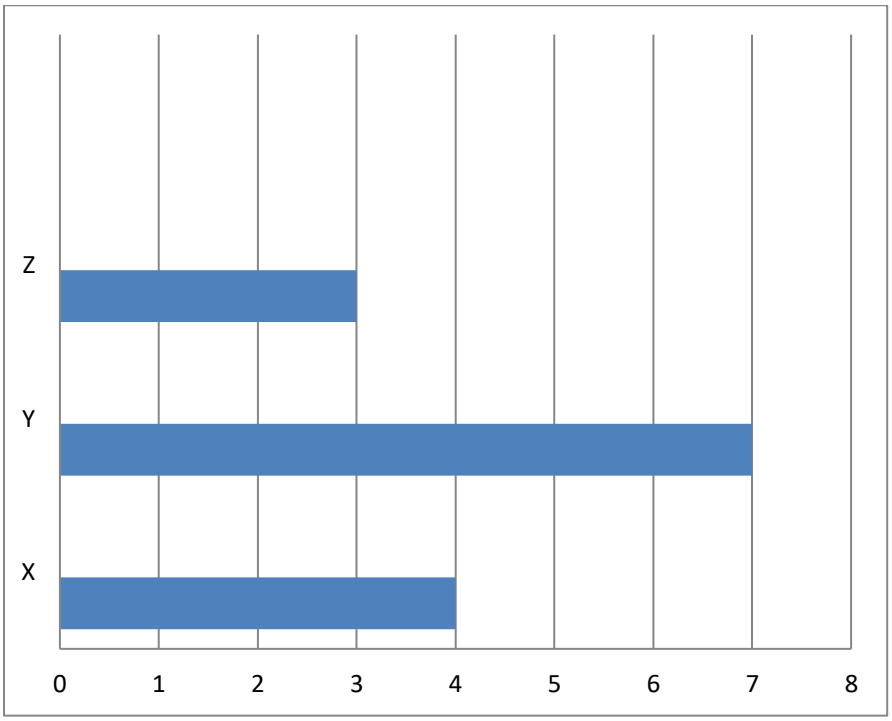

```
import matplotlib.pyplot as plt

import numpy as np

a = np.array(["X", "Y", "Z"])

b = np.array([500, 400, 300])

plt.bar(a,b)

plt.show()
```

Output:




```
import matplotlib.pyplot as plt

import numpy as np

a = np.array(["X", "Y", "Z"])

b = np.array([500, 400, 300])

plt.bar(a,b, color = "green")

plt.show()
```

Output:




```
import matplotlib.pyplot as plt

import numpy as np

a = np.array(["X", "Y", "Z"])

b = np.array([4, 7, 3])

plt.barh(a, b)

plt.show()
```

```
from scipy import constants

print(constants.pi) # Output: 3.141592653589793

print(constants.speed_of_light) # Output: 299792458.0

print(constants.golden_ratio) # Output: 1.618033988749895

print(constants.gravitational_constant) # Output: 6.67408e-11

print(constants.elementary_charge) # Output: 1.6021766208e-19

print(constants.electron_mass) # Output: 9.10938356e-31

print(constants.proton_mass) # Output: 1.672621898e-27

print(constants.neutron_mass) # Output: 1.674927471e-27

print(constants.Planck) # Output: 6.62607004e-34

print(constants.hbar) # Output: 1.0545718001391127e-34

print(constants.Avogadro) # Output: 6.022140857e+23
```

Machine Learning	Data Science
Prove mathematical properties of models	Understand empirical properties of models

Benford's Law:

The probability of a first digit being ' x ' is: $P(x) = \log_{10}(1 + 1/x)$

For example:

The probability of a first digit being ' 2 ' is:

$$P(2) = \log_{10}(1 + 1/2) = \log_{10}(1.5) = 0.17609125905$$

Data Processing

ETL (extract, transform, load)

Lotka's law

(Inverse square law of scientific productivity)

$a^x \times b = constant$

- a: number of publications
- b: number of authors credited with 'a' publications
- x is often nearly 2

Web mining

Web content mining

Mines content of Web pages

Web usage mining

Mines user interaction data recorded by Web server

Web structure mining

Analyzes links to and from Web page

The SQL commands are mainly categorized into 4 categories:

- **Data Definition Language:** commands used to define the database.

  Example: CREATE, DROP, ALTER, TRUNCATE, COMMENT, RENAME
- **Data Manipulation Language:** commands deal with the manipulation of data present in the database.

  Example: SELECT, INSERT, UPDATE, DELETE
- **Data Control Language:** commands deal with the rights, permissions and other controls of the database system.

  Example: GRANT, INVOKE
- **Transaction Control Language:** commands deal with the transaction of the database.
  Example: COMMIT, ROLLBACK, SAVEPOINT, SET TRANSACTION

A Schema in SQL is a collection of database objects associated with a database.

SQL Operators

```
SELECT 60 + 40;
SELECT 60 - 40;
SELECT 60 * 40;
SELECT 60 / 40;
SELECT 60 % 40;
```

Output

```
60 + 40
100
60 - 40
20
60 * 40
2400
60 / 40
1.5000
60 % 40
20
```

Agile methodology

- Requirements Gathering
- Software Design
- Implementation
- Testing
- Deployment
- Maintenance

Three phases are often taken to build a business strategy:

- Analysis
- Decision
- Implementation

Lean development:

- **Eliminate unnecessary steps**
- Create quality in development
- **Increase Feedback and amplify learning**
- Delay decisions until you have learned enough to make a good decision
- **Document the whole infrastructure**
- Deliver software as fast as possible
- **Empower the programming team**
- Build a focus on system integrity and the development sequence has to be perfected enough to be able to delete errors in the code − in order to create a flow of reliability and validity

Changeability:

The software product's ability to allow for the implementation of specific customizations

Language Theory	How do computations get expressed?
Automata Theory	How do computations get done?
Computability Theory	What are the computational system's underlying bounds?
Complexity Theory	What tools are needed to carry out particular computations?

- **Accuracy testing:** Testing a software application to evaluate its accuracy
- **Attractiveness:** The ability of the software application to appeal to users

Design thinking + Lean Development + Agile Development = Project success

The most popular DevOps tools:

- **Git :** Version Control System tool

Assist software teams to track changes in source code over time.

- **Jenkins :** Continuous Integration tool

Assist the software development process where multiple developers make small, frequent changes to code. The software is tested immediately after each change is made. With each change, code is built and tested. If the test is passed, the build is tested for deployment. If the deployment is successful, the code is pushed to production. This code change, build, test, and deploy is a continuous process and hence the name continuous integration continuous deployment.

- **Selenium :** Continuous Testing tool

Assist the testing of the software at every stage of the software development life cycle.

- **Puppet, Chef, Ansible :** Configuration Management and Deployment tools

Automate administration tasks, configure multiple servers at a time − without any user intervention and gives software team the power to deploy multi-tier applications as fast as possible.

- **Nagios :** Continuous Monitoring tool

Assist the software team to detect application performance issues, identify their cause and implement a solution before the issue leads to unplanned application downtime and lost revenue.

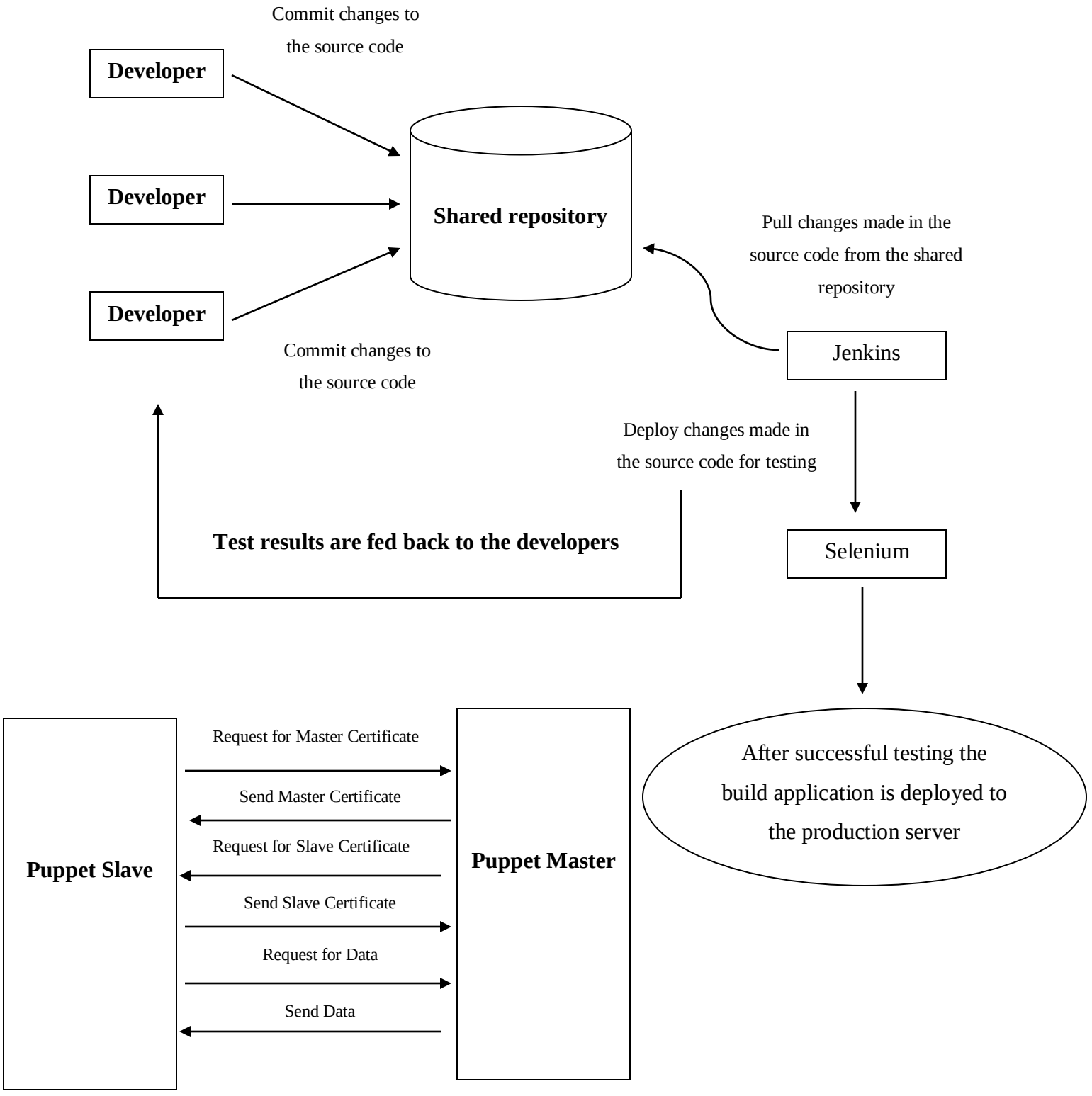
- **Docker :** Containerization tool

Assist the process of packaging a software application along with its required libraries, frameworks and configuration files together so that it runs quickly and reliably in various computing environments.

Selenium supports 2 types of testing:

- **Functional testing** → performed to ensure all functionalities of an application is working as expected.

- **Regression Testing** → performed to ensure that new product version still works after the new changes have been made.



AWS Services:

- Compute
- Storage
- Networking
- Monitoring
- Databases
- Security

AWS EC2

A part of the cloud computing infrastructure from Amazon.com that offers secure, scalable compute capability

AWS Lambda

A serverless compute service that makes it easier to implement the code without worrying about setting up or managing servers

AWS Elastic Beanstalk

An AWS service that is simple to use and creates the foundation for developing and scaling web applications

AWS Lightsail

An AWS service that gives application developers access to a customizable virtual private server (VPS) and a number of simple-to-use tools for data storage, code execution and the creation of websites or web-based applications

Python program:

```
x = ['Albert Einstein', 1879, 1955, 'physicist']

y = ['Charles Darwin', 1809, 1882, 'biologist']

print(x[0], y[2]) # Output: Albert Einstein 1882

print(y[0].split()[-1]) # Output: Darwin

print(y[2] * 1.25) # Output: 2352.5
```

```
print(("%d" % 1905), '{:d}'.format(1915))

# Output: 1905 1915
```

```
print("19" + str(15), int("1900") + 5)

# Output: 1915 1905
```

```
import os

os.system('1.txt') # opens the file 1.txt
```

```
x= {'albert': 1905, 'einstein': 'papers'}

print('%(albert)s => %(einstein)s' % x) # Output: 1905 => papers
```

```
import sys

print(sys.platform) # Output: win32

if sys.platform[:1] == 'w': print('Microsoft Windows') # Output: Microsoft Windows
```

```
import os

path = 'C:/Users/Manju'

# prints True if the path exists otherwise prints False

print(os.path.exists(path)) # Output: True
```

```
import os

path = 'C:/Users/Manju'

# prints True if the path is an existing directory otherwise prints False

print(os.path.isdir(path)) # Output: True
```

```
import os

# prints the size of path 'C:/Users/Manju' in bytes

print(os.path.getsize('C:/Users/Manju')) # Output: 40960
```

```
import os

print(os.path.split('C:/Users/Manju')) # Output: ('C:/Users', 'Manju')

print(os.path.join("C:", "/Users/Manju")) # Output: C:/Users/Manju
```

```
import os

os.startfile("1.txt") # open file in Notepad

os.startfile("1.docx") # open file in Microsoft Word
```

```
import sys

sys.stdout.write('Albert Einstein' + '\n') # Output: Albert Einstein
```

```
input('Albert Einstein') # Output: Albert Einstein


import glob

# prints all the .txt files in the current directory

print(glob.glob('*.txt'))

# Output: ['1.txt', '2.txt', 'names.txt', 'output.txt', 'spam.txt']


import os

# prints all the files and directories in the current directory

print(os.listdir(os.curdir))
```

```
import time

print('Albert') # Output: Albert

time.sleep(3) # wait for 3 seconds

print('Einsteinian %d and [%s] papers' % (1905, 1915)) # Output: Einsteinian 1905 and [1915] papers
```

```
# prints the content of 1.txt file

print(open('1.txt').read())


x = open('output.txt', 'r').read()

y = open(r'C:\Users\Manju\output.txt', 'r').read()

print(x == y) # Output: True
```

```
import html.parser

print(html.escape("4<9 <b>Albert</b>")) # Output: 4&lt;9 &lt;b&gt;Albert&lt;/b&gt;

print(html.unescape("4&lt;9 &lt;b&gt;Albert&lt;/b&gt;")) # Output: 4<9 <b>Albert</b>
```

```
print('Stephen %s %s' % ('William', 'Hawking')) # Output: Stephen William Hawking

print('Alan {} {}'.format('Mathison', 'Turing')) # Output: Alan Mathison Turing

print('Albert "%-5s", %06d' % ('Einstein', 1)) # Output: Albert "Einstein", 000001

print('Albert "{0:<5}", {1:06}'.format('Einstein', 1)) # Output: Albert "Einstein", 000001
```

```
print('x*y*z'.split('*')) # Output: ['x', 'y', 'z']

print('*'.join(['x', 'y', 'z'])) # Output: x*y*z
```

```
import re

print(re.split('--|==', 'xx--yy==zz')) # Output: ['xx', 'yy', 'zz']

print(re.split('[-=]', 'xx-yy=zz')) # Output: ['xx', 'yy', 'zz']

print(re.split('(--)|(==)', 'xx--yy==zz')) # Output: ['xx', '--', None, 'yy', None, '==', 'zz']

print(re.split('(?:--)|(?:==)', 'xx--yy==zz')) # Output: ['xx', 'yy', 'zz']
```

```
x = {}

x['b'] = 21

exec('a = 100', x, x)

exec('c = a + b', x, x)

print(x['c']) # Output: 121
```

```
import math

x = 1

for i in range(3):

    print(math.pi*x*(1+x))
```

Output:

```
6.283185307179586

6.283185307179586

6.283185307179586
```

```
x="Albert"

y=5

print(x[y-1]) # Output: r
```

```
x="Albert Einstein grew up in Munich"

print(x)

"""

Output:

Albert Einstein grew up in Munich

"""

print(x.center(50))

"""

Output:

        Albert Einstein grew up in Munich

"""
```

```
import random
for x in range(3):
 print(random.randint(2, 9))

# Output:

                6

                3

                9
```

```
print('Alan', end='')

print('Turing') # Output: AlanTuring
```

```
print('Albert Einsteinian ' + str(1905) + ' papers')

# Output: Albert Einsteinian 1905 papers


print(int(5.2) + 2) # Output: 7

print(float(5.2) + 2) # Output: 7.2
```

```
for x in range(3):

 print('Alan (' + str(x) + ')')

# Output:

Alan (0)

Alan (1)

Alan (2)
```

```
print((3 < 7) and (7 < 9)) # Output: True
```

```
(3 < 7) and (7 < 9)
        ↓
True and (7 < 9)
        ↓
True and True
        ↓
      True
```

```
for x in range(3, -1, -1):
 print(x)
# Output:
              3
              2
              1
              0
```

```
def myfunc():
  print('Alan')
myfunc()
myfunc()
myfunc()
```

Output:

```
Alan
Alan
Alan
```

```
def myfunc(x):
   print('Albert, ' + x)
myfunc('1905')
myfunc('1915')
myfunc('1955')
```

Output:

```
Albert, 1905
Albert, 1915
Albert, 1955
```

```
def myfunc(x):
 return 55 / x
print(myfunc(5)) # Output: 11.0
```

```
print('Albert Einstein')
indent = 3 # 3 spaces to indent
print(' ' * indent, end='')
print('Albert Einstein')
```

Output:

```
Albert Einstein
   Albert Einstein
```

```
x = [['alan', 'albert'], [12, 22, 32, 42, 62]]

print(x[0]) # Output: ['alan', 'albert']

print(x[0][1]) # Output: albert

print(x[1][4]) # Output: 62
```

```
print('Alan Turing'.startswith('Alan'))

# Output: True

print('Alan Turing'.endswith('Turing'))

# Output: True
```

```
x = ['alan', 'albert', 'james', 'mary']

for a in range(len(x)):

   print(x[a], end =" ") # Output: alan albert james mary
```

```
x = 'Albert '

x += 'Einstein'

print(x) # Output: Albert Einstein
```

```
x = 'Albert '

x *= 2

print(x) # Output: Albert Albert
```

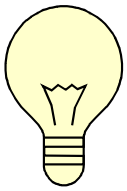
```
x = ['c', 'b', 'C', 'B']

x.sort(key=str.lower)

print(x) # Output: ['b', 'B', 'c', 'C']
```

```
import random

if random.randint(0, 1) == 0:

  print("random number is 0")

else:

  print("random number is 1")
```

```
print('albert'.upper()) # Output: ALBERT

print('albert'.upper().lower()) # Output: albert

print('albert'.upper().lower().upper()) # Output: ALBERT

print('ALBERT'.lower()) # Output: albert

print('ALBERT'.lower().islower()) # Output: True
```

JavaScript

Did you know?



Java and **JavaScript** are conceptually and visually quite distinct languages

Paradigm	Multi-paradigm: event-driven, functional, imperative, procedural, object-oriented programming
Designed by	**Brendan Eich** of Netscape initially; others have also contributed to the ECMAScript standard
First appeared	December 4, 1995; 26 years ago
Stable release	ECMAScript 2021 / June 2021; 15 months ago
Preview release	ECMAScript 2022 / 22 July 2021; 13 months ago
Typing discipline	Dynamic, weak, duck
Filename extensions	`.js` `.cjs` `.mjs`
Website	www.ecma-international.org/publications-and-standards/standards/ecma-262/

Brendan Eich

Creator of JavaScript

Major implementations

V8, JavaScriptCore, SpiderMonkey, Chakra

Influenced by

Java, Scheme Self, AWK, HyperTalk
Influenced
ActionScript, AssemblyScript, CoffeeScript, Dart, Haxe, JS++, Objective-J, Opa, TypeScript

Designed by an American technologist and co-founder of the Mozilla project, the **Mozilla Foundation**, and the Mozilla Corporation **"Brendan Eich"** and it was developed under the name Mocha, the language was officially called **LiveScript** when it first shipped in beta releases of **Netscape Navigator 2.0** in September 1995, but it was renamed JavaScript

JavaScript / **Jscript** is an relatively popular object-oriented scripting interpreted programming language embedded in high level programming language of **Hypertext Markup Language**

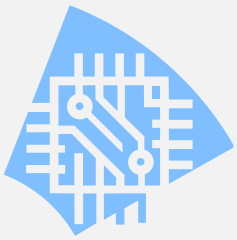
The language that websites are rendered in and basically, everything you and your readers see on the "frontend" is **HTML**), commonly abbreviated as HTML pages

primarily used to design interactive websites with dynamic content and perform functions that the HTML cannot do, because of its reliability, simplicity and easy to understand, easy to use, write, modify and debug and quick to learn.

The primary language used in web design is **HTML**. In addition to **HTML**, **CSS** is employed to further enhance web page appearance. **HTML** and **JavaScript** are combined to build dynamic websites.

Advantages of JavaScript:

- Massive community support and no requirement for compilers
- Simple to learn and accessible to everyone
- Both client-side and server-side usage
- Aids in creating a comprehensive solution



Syntax:

```
<script>

// JavaScript code to be executed

</script>
```

Tag used to incorporate an executable client side scripting language such as **JavaScript** into an HTML page

Inserted in the **<body>** or in the **<head>** section of an HTML document or in both

```
<script>

// Single-line Comment

/* Multi-line

   Comment

 */
```

Describe **JavaScript code** and make it more understandable, but ignored by JavaScript **(will not be executed)**

```
// Display the text "Albert Einstein" on the console screen using document.write() method

document.write('Albert Einstein');

document.write("Albert Einstein");


// Display the result "31" on the console screen using document.write() method

document.write(15 + 16);


</script>
```

Both double and single quotes are supported by **JavaScript**

```
<script>

// Display the text "Albert Einstein" on the console screen using console.log() method

console.log('Albert Einstein');

console.log("Albert Einstein");



// Display the result "31" on the console screen using console.log() method

console.log(15 + 16);



</script>
```

```
<script>

// Declare x, give it the value of 15 using the assignment operator "="

let x = 15;


// Declare y, give it the value of 12 using the assignment operator "="

let y = 12;


// Declare z, give it the value of "x + y" using the assignment operator "="

let z = x + y;


document.write(z); // Display the computed result



</script>
```

The **addition operator** (+) adds 2 numbers 'x' and 'y'

```
<script>

let x = 15;

let y = 12;

let z = x - y;

document.write(z); // Output: 3

</script>
```

The **subtraction operator** (-) subtract the number "y" from the number "x"

```
<script>

let x = 15;

let y = 12;

let z = x * y;

document.write(z); // Output: 180

</script>
```

The **multiplication operator** (*) multiplies 2 numbers 'x' and 'y'

```
<script>

let x = 6;

let y = 2;

let z = x / y;

document.write(z); // Output: 3

</script>
```

The **division operator** (/) divides the number 'x' by the number 'y'

```
<script>

let x = 7;

let y = 2;

let z = x % y;

document.write(z); // Output: 1

</script>
```

The **modulus operator** (%) returns the division remainder

```
<script>

let x = 7;

x++;

let z = x;

document.write(z); // Output: 8

</script>
```

The **increment operator** (++) increments the number "x"

```
<script>

let x = 7;

x--;

let z = x;

document.write(z); // Output: 6

</script>
```

The **decrement operator** (--) decrements the number "x"

```
<script>

let x = 4;

let z = x ** 3;

document.write(z); // Output: 64

</script>
```

The **exponentiation operator** (**) raises the number "x" to the power of 3

```
<script>

let x = 4;

let z = Math.pow(x,3);

document.write(z); // Output: 64

</script>
```

x ** 3 produces the same result as Math.pow(x,3)

```
<script>

let x = 50 + 50 * 3;

document.write(x); // Output: 200
```

50 + 150 = 200

If **parentheses** are used, the operations inside the **parenthesis** are computed first

```
let y = (50 + 50) * 3;
```

100 * 3 = 300

```
console.log(y); // Output: 300

</script>
```

```
<!DOCTYPE html>  <!-- Identifies this document as being an HTML5 document -->
<html> <!-- Root element of an HTML Document -->
<body>  <!-- Defines the body of the HTML document -->

<!--
 The HTML document contains a <p> element that has the id attribute with the value "add".
-->

<p id="add"> 36 </p>

<script>

document.getElementById("add").innerHTML = 15 + 16;

</script>

</body>
</html>
```

Defines a paragraph

Change the HTML content inside an **<p>** element

36 ⟶ (15 + 16) = 31

Find an HTML element with an id that matches a specified string "add"

This program display the computed result "31" on the console screen using document.getElementById() method

Output:

31

```
var x = 15;
var y = 26;
var z = x * y;
```

The "var" **keyword** has been used to declare the variables x, y and z

```
let x = 15;
let y = 26;
let z = x * y;
```

The "let" **keyword** has been used to declare the variables x, y and z

```
x = 15;
y = 26;
z = x * y;
```

The **no keyword** has been used to declare the variables x, y and z

```
const x = 3;
x = 6;   // This will give an error
```

Variable defined with "**const**" keyword cannot be changed or reassigned

```
<script>
let x = 6.25; // A number with decimal
let y = 6; // A number without decimal
console.log(x+y);
// Output: 12.25
</script>
```

```
<script>
x = 549e5;
y = 549e-5;
console.log(x); // Output: 54900000
console.log(y); // Output: 0.00549
</script>
```

x and y are **undeclared variables**

```
<script>
// The outcome of adding two integers is a number
x = 2;
y = 3;
console.log(x+y); // Output: 5

// If we add a number and a string, the outcome will be a string concatenation
x = 1905;
y = " Papers";
console.log(x+y); // Output: 1905 Papers
</script>
```

```
<script>

x = 5;

y = 6;

console.log(" " + x + y); // Output: 56

</script>
```

```
<script>

console.log("Albert \\ John");

</script>
```

```
Output: Albert \ John
```

```
<script>

// Because x and y are both integers, 10 + 9 is added first

// Then, because z is a string, 19 + "15" is concatenated


x = 10;

y = 9;

z = "15";

console.log(x + y + z); // Output: 1915

</script>
```

```
<script>

// When dividing, JavaScript will convert numeric strings to numbers


x = "6"; // numeric string

y = "2"; // numeric string

console.log(x / y); // Output: 3

</script>
```

```
<script>

// When Multiplying, JavaScript will convert numeric strings to numbers


x = "6"; // numeric string

y = "2"; // numeric string

console.log(x * y); // Output: 12

</script>
```

```
<script>

console.log("Albert \"1905\" Papers");

// Output: Albert "1905" Papers

console.log('Alan\'s Papers');

// Output: Alan's Papers

</script>
```

```
<script>

// When Subtracting, JavaScript will convert numeric strings to numbers


x = "6"; // numeric string

y = "2"; // numeric string

console.log(x - y); // Output: 4

</script>
```

```
<script>

// When Adding, JavaScript will not convert numeric strings to numbers


x = "6"; // numeric string

y = "2"; // numeric string

console.log(x + y); // Output: 62

</script>
```

JavaScript uses the "+" operator to concatenate the 2 numeric strings

```
<script>

x = "6"; // numeric string

y = "Albert"; // non-numeric string

console.log(x / y); // Output: NaN (Not a Number)

</script>
```

NaN is the result of dividing a number by a non-numeric string

```
<script>

x = "6";

y = "Albert";

console.log(isNaN(x / y)); // Output: true

</script>
```

```
<script>

x = 6;

y = NaN;

z = "5";

console.log(x + y); // Output: NaN

console.log(y + z); // Output: NaN5

</script>
```

```
<script>

console.log(5/0);   // Output: Infinity

console.log(-5/0); // Output: -Infinity

</script>
```

```
<script>

x = 25; // x is a number

y = new Number(25); // y is a Number object

console.log(x==y); // Output: true

</script>
```

x and y are equal

```
<script>

x = 25; // x is a number

y = new Number(25); // y is a Number object

console.log(x===y); // Output: false

</script>
```

x and y are of different type

```
<script>

// Display the length of a string 'Albert'

x = "Albert";

console.log(x.length); // Output: 6

</script>
```

```
<script>

console.log("Welcome to 'Alberta'");

// Output: Welcome to 'Alberta'

console.log('Welcome to "Alberta"');

// Output: Welcome to "Alberta"

</script>
```

```
<script>

x = "Albert"; // x is a string

y = new String("Albert"); // y is a String object

console.log(x==y); // Output: true

</script>
```

x and y are equal

A block scoped variable:

A variable declared inside a block won't be accessible from the outside. Even though a block can be embedded within a function, a block scoped variable will not be accessible outside of the block even if the block is inside the function

```
console.log("Albert
Einstein");
```

Error: Unterminated string literal

```
console.log("Albert \
Einstein");
```

No Error

Output: Albert Einstein

```
console.log("Albert "
+
"Einstein");
```

No Error

Output: Albert Einstein

```
<script>

x = "Albert"; // x is a string

y = new String("Albert"); // y is a String object

console.log(x===y); // Output: false

</script>
```

x and y are of different type

```
<script>

x = new String("Albert");

y = new String("Albert");

console.log(x==y); // Output: false

</script>
```

JavaScript objects cannot be compared

Comparing 2 JavaScript objects always returns false

```
<script>

x = new String("Albert");

y = new String("Albert");

console.log(x===y); // Output: false

</script>
```

Fallback function: A function that defines a response to be given whenever a call to a remote service is unsuccessful. A fallback function, for example, can be used to deliver the client a personalized message regarding service failures

```
<script>

let x = 20; // Number

let y = "Albert"; // String

let z = {Name:"Albert", Country:"Germany"}; // Object

let w = ["Apple", "Orange", "Mango"]; // Array

console.log(x); // Output: 20

console.log(y); // Output: Albert

console.log(z); // Output: { Name: 'Albert', Country: 'Germany' }

console.log(w); // Output: [ 'Apple', 'Orange', 'Mango' ]

</script>
```

The object "**z**" in the example above has 2 properties: **Name and Country** and 2 property values: **Albert and Germany**

```
<script>

let i;          // Now 'i' is undefined

i = 15;        // Now 'i' is a Number

i = "Albert";  // Now 'i' is a String

console.log(i); // Output: Albert

</script>
```

Polyfill:

A software program used to add current features to outdated browsers that lack native capability

```
<script>

let x = 35;

let y = 35;

let z = 56;

console.log(x==y); // Returns true

console.log(x==z); // Returns false

</script>
```

```
<script>

// The type of a variable or expression is returned by the "typeof" operator

console.log(typeof "Albert"); // Output: string

console.log(typeof ("Albert" + "Einstein")); // Output: string

console.log(typeof 52); // Output: number

console.log(typeof (23 + 54)); // Output: number

console.log(typeof undefined); // Output: undefined

</script>
```

```
<script>

// The "typeof" of a empty string is "string"

console.log(typeof ""); // Output: string

</script>
```

```
<script>

var x = "Albert";

var y = "Einstein";

var z = x + " " + y;

console.log(z); // Output: Albert Einstein

</script>
```

```
<script>

var x = "Albert ";

x += "Einstein";

console.log(x); // Output: Albert Einstein

</script>
```

```
<script>

// Create an object named 'x'

x = {Name: "Albert", Country:"Germany", Age:56};

// Accessing an property_value via object_name.property_name

console.log(x.Name); // Output: Albert

console.log(x.Country); // Output: Germany

console.log(x.Age); // Output: 56

</script>
```

property_name	property_value
Name	Albert
Country	Germany
Age	56

```
<script>

// Create an object named 'x'

x = {Name: "Albert", Country:"Germany", Age:56};

// Accessing an property_value via object_name["property_name"]

console.log(x["Name"]); // Output: Albert

console.log(x["Country"]); // Output: Germany

console.log(x["Age"]); // Output: 56

</script>
```

The object **"x"** in the example above has 3 properties: **Name, Country and Age** and 3 property values: **Albert, Germany and 56**

```
<script>

var x = 20;

x += 15; // x = x + 15

console.log(x); // Output: 35

</script>
```

Tree Shaking:

A phrase frequently used in the context of **JavaScript** to refer to the elimination of dead code

```
<script>

var x = 20;

x -= 15; // x = x - 15

console.log(x); // Output: 5

</script>
```

```
<script>

var x = 20;

x *= 15; // x = x * 15

console.log(x); // Output: 300

</script>
```

In **object-oriented programming**, **side effects** are activities a function accomplishes that aren't communicated and may not be expected

```
<script>

var x = 20;

x *= 15; // x = x / 15

console.log(x); // Output: 1.3333333333333333

</script>
```

```
<script>

var x = 21;

x %= 2; // x = x % 2

console.log(x); // Output: 1

</script>
```

Shim:

A chunk of code added to existing programs to change how they behave; this is typically done by introducing new APIs to get through existing programs' issues

```
<script>

var x = 4;

x **= 2; // x = x ** 2

console.log(x); // Output: 16

</script>
```

```
<script>
/*
This example calls a function "add"
which performs addition of x and y,
and returns the computed result
*/

function add(x, y) {
  return x + y;
}
console.log(add(14, 13)); // Output: 27

</script>
```

```
<script>
var c = mult(8, 4);


// Function is called, return value will end up in variable 'c'

function mult(x, y) {
  return x * y;         // Function returns the product of x and y
}

console.log(c); // Output: 32 (computed result)

</script>
```

```
<script>
let x = "Albert Einstein";
console.log(x.replace("Albert", "Elsa")); // Output: Elsa Einstein
</script>
```

```
<script>
let x = "Albert Einstein";
console.log(x.replace(/ALBERT/i, "Elsa")); // Output: Elsa Einstein
</script>
```

```
<script>
let x = "Albert and Albert Einstein";
console.log(x.replace("Albert", "Elsa"));
// Output: Elsa and Albert Einstein
console.log(x.replace(/Albert/g, "Elsa"));
// Output: Elsa and Elsa Einstein
</script>
```

```
<script>
// A string is converted to upper case with toUpperCase() method
var x = "albert";
console.log(x.toUpperCase());
// Output: ALBERT
</script>
```

```
<script>

// A string is converted to lower case with toLowerCase() method

var x = "ALBERT";

console.log(x.toLowerCase());

// Output: albert

</script>
```

```
<script>

 // The concat() method can be utilized instead of the "+" operator

console.log("Albert" + " " + "Einstein");

// Output: Albert Einstein



console.log("Albert".concat(" ", "Einstein"));

// Output: Albert Einstein

</script>
```

```
<script>

// Whitespace on both sides of a string is removed using the trim() method

var x = "      Albert      ";

console.log(x.trim());

// Output: Albert

</script>
```

```
<script>

var x = "      Albert      ";

// Removes whitespace only from the start of a string

console.log(x.trimStart());

// Removes whitespace only from the end of a string

console.log(x.trimEnd());

</script>
```

0	1	2	3	4	5	6	7
E	i	n	s	t	e	i	n

```
<script>


/*
 Slice out a portion of a string "Einstein"
 from position 2 to position 6 (6 not included)
*/


var x = "Einstein";
console.log(x.slice(2, 6)); // Output: nste


</script>
```

```
<script>

var x = "16";


/*
The padStart() method is used to pad a numeric string "16"
with another string "x" until it reaches the given length
*/


console.log(x.padStart(4,"x")); // Output: xx16
console.log(x.padStart(5,"x")); // Output: xxx16


console.log(x.padEnd(4,"x")); // Output: 16xx
console.log(x.padEnd(5,"x")); // Output: 16xxx


</script>
```

```
<script>

var x = 16;

// convert the number 16 to a numeric string "16"


var y = x.toString();

console.log(y.padStart(4,"x")); // Output: xx16

console.log(y.padEnd(4,"x")); // Output: 16xx

</script>
```

0	1	2	3	4	5	6	7
E	i	n	s	t	e	i	n

```
<script>

let x= "Einstein";

console.log(x[0]); // Output: E

console.log(x[1]); // Output: i

console.log(x[2]); // Output: n

console.log(x[3]); // Output: s

console.log(x[4]); // Output: t

console.log(x[5]); // Output: e

console.log(x[6]); // Output: i

console.log(x[7]); // Output: n

</script>
```

```
<script>

/*

The use of split() method to split the

string "Alan Mathison Turing" into words

*/


var x = 'Alan Mathison Turing';

console.log(x.split(' ')); // Output: [ 'Alan', 'Mathison', 'Turing' ]


</script>
```

```
<script>

var x = 'Alan Mathison Turing';

console.log(x.split(' ', 2)); // Output: [ 'Alan', 'Mathison' ]

</script>
```

```
<script>

var x = "Sleep well! What is up? It is Albert here. Albert and I are connected.";

console.log(x.split(/[!,?,.]/));

</script>
```

```
[ 'Sleep well',
  ' What is up',
  ' It is Albert here',
  ' Albert and I are connected',
  '' ]
```

Output

```
<script>

var x = "Sleep well! What is up? It is Albert here. Albert and I are connected.";

console.log(x.split(/([!,?,.])/));

</script>
```

Output

```
[ 'Sleep well',
  '!',
  ' What is up',
  '?',
  ' It is Albert here',
  '.',
  ' Albert and I are connected',
  '.',
  '' ]
```

```
<script>

/*

The indexOf() method returns the index of

(the position of) the first occurrence of

a specified text "Einstein" in a string

"Elsa Einstein was Albert Einstein's wife"

*/



var x = "Elsa Einstein was Albert Einstein's wife";

console.log(x.indexOf("Einstein")); // Output: 5

</script>
```

```
<script>

/*

The lastIndexOf() method returns the index of

(the position of) the last occurrence of a

 a specified text "Einstein" in a string

"Elsa Einstein was Albert Einstein's wife"

*/

var x = "Elsa Einstein was Albert Einstein's wife";

console.log(x.lastIndexOf("Einstein")); // Output: 25

</script>
```

```
<script>

/*

The search() method returns the index of

(the position of) the first occurrence of

a specified text "Einstein" in a string

"Elsa Einstein was Albert Einstein's wife"

*/



var x = "Elsa Einstein was Albert Einstein's wife";

console.log(x.search("Einstein")); // Output: 5

</script>
```

```
<script>



/*

 Search a string "Elsa Einstein was Albert Einstein's wife" for "ein"

*/



var x = "Elsa Einstein was Albert Einstein's wife";

console.log(x.match(/ein/g)); // Output: [ 'ein', 'ein' ]



</script>
```

```
<script>

var x = "Elsa EINSTEIN was Albert Einstein's wife";

console.log(x.match(/ein/g)); // Output: [ 'ein']



/* Conduct a global, case-insensitive search for "ein" */

console.log(x.match(/ein/gi));

// Output: [ 'EIN', 'EIN', 'Ein', 'ein' ]

</script>
```

```
<script>



/*

The includes() method returns true if a string

"Elsa Einstein was Albert Einstein's wife"

contains a specified text "was"

*/



var x = "Elsa Einstein was Albert Einstein's wife";

console.log(x.includes("was")); // Output: true



</script>
```

```
<script>
/*
Check if a string
"Elsa Einstein was Albert Einstein's wife"
includes a specified text "was",
starting the search at position 10
*/

var x = "Elsa Einstein was Albert Einstein's wife";
console.log(x.includes("was", 10));  // Output: true

</script>
```

```
<script>
/*
The startsWith() method returns true if a string
"Elsa Einstein was Albert Einstein's wife"
begins with a specified text "Elsa",
otherwise returns false
*/

var x = "Elsa Einstein was Albert Einstein's wife";
console.log(x.startsWith("Elsa")); // Output: true
</script>
```

```
<script>
/*
The endsWith() method returns true if a string
"Elsa Einstein was Albert Einstein's wife"
ends with a specified text "wife",
otherwise returns false
*/

var x = "Elsa Einstein was Albert Einstein's wife";
console.log(x.endsWith("wife")); // Output: true
</script>
```

```
<script>

// The toString() method converts a number to a string
var x = 1915;
console.log(x.toString()); // Output: 1915
console.log((156).toString()); // Output: 156
console.log((120 + 42).toString()); // Output: 162
// 120 + 42 = 162
// console.log((162).toString());

</script>
```

```
<script>


/*
Multiline strings are supported through
template literals. Internet Explorer does
not support template literals
*/


let x =
`Elsa Einstein
was
Albert Einstein's
wife`;


console.log(x);
</script>
```

Output:

```
Elsa Einstein
was
Albert Einstein's
wife
```

```
<script>


let x = "Mathison";
let y = "Turing";
console.log(`Alan ${x} ${y}`);
// Output: Alan Mathison Turing


</script>
```

```
<script>
// The toExponential() method converts a number to its exponential form
var x = 5.496;
console.log(x.toExponential(2));    // Output: 5.50e+0
console.log(x.toExponential(4));   // Output: 5.4960e+0
console.log(x.toExponential(6)); // Output: 5.496000e+0
</script>
```

```
<script>

/*
The toFixed() method rounds the number "5.496"
to a specified number of decimals
*/

var x = 5.496;
console.log(x.toFixed(0)); // Output: 5
console.log(x.toFixed(2)); // Output: 5.50
console.log(x.toFixed(4)); // Output: 5.4960
console.log(x.toFixed(6)); // Output: 5.496000

</script>
```

```
<script>


/*

The toPrecision() method formats a number

"5.496" to a specified precision or length

*/


var x = 5.496;

console.log(x.toPrecision());   // Output: 5.496

console.log(x.toPrecision(2)); // Output: 5.5

console.log(x.toPrecision(4)); // Output: 5.496

console.log(x.toPrecision(6)); // Output: 5.49600


</script>
```

```
<script>

// MAX_VALUE returns the largest possible number in JavaScript

console.log(Number.MAX_VALUE);

// Output: 1.7976931348623157e+308


// MIN_VALUE returns the smallest possible number in JavaScript

console.log(Number.MIN_VALUE);

// Output: 5e-324

</script>
```

```
<script>
console.log(Number.POSITIVE_INFINITY); // Output: Infinity
console.log(Number.NEGATIVE_INFINITY); // Output: -Infinity
console.log(Number.NaN); // Output: NaN
</script>
```

```
<script>

// JavaScript variables are converted to numbers using the Number() Method

console.log(Number(true)); // Output: 1
console.log(Number(false)); // Output: 0
console.log(Number("54"));  // Output: 54
console.log(Number("  54")); // Output: 54
console.log(Number("54  ")); // Output: 54
console.log(Number(" 54  ")); // Output: 54
console.log(Number("54.96")); // Output: 54
console.log(Number("54,96")); // Output: NaN
console.log(Number("54 96")); // Output: NaN
console.log(Number("Albert")); // Output: NaN

</script>
```

Not a Number or NaN is returned if the JavaScript variables cannot be converted to numbers

```
<script>
console.log(parseInt("54.96")); // Output: 54
console.log(parseFloat("54.96")); // Output: 54.96
</script>
```

```
<script>

/* Creating an array 'x' and then providing the elements */

x = [];
x[0]= "Apple";
x[1]= "Orange";
x[2]= "Mango";
console.log(x); // Output: [ 'Apple', 'Orange', 'Mango' ]

</script>
```

```
<script>

/* Creating an array 'x' and then assigning values to it */

x = new Array("Apple", "Orange", "Mango");
console.log(x); // Output: [ 'Apple', 'Orange', 'Mango' ]

</script>
```

```
<script>
x = ["Apple", "Orange", "Mango"];
// Change the value of the first Array element
x[0] = "Grapes";
console.log(x); // Output: [ 'Grapes', 'Orange', 'Mango' ]
</script>
```

```
<script>
x = ["Apple", "Orange", "Mango"];
// Display the number of array elements
console.log(x.length); // Output: 3
</script>
```

Getting Data Type

↓

Obtaining the data type of an object

```
<script>
x = ["Apple", "Orange", "Mango"];
// Display the Last Array Element
console.log(x[x.length - 1]); // Output: Mango
</script>
```

Setting Data Type

↓

Setting the data type of an object

```
<script>
x = ["Apple", "Orange", "Mango"];
// Adds a new element "Grapes" to x
x.push("Grapes");
console.log(x); // Output: [ 'Apple', 'Orange', 'Mango', 'Grapes' ]
</script>
```

```
<script>
x = ["Apple", "Orange", "Mango"];
// Adds a new element "Grapes" to x
x[x.length] = "Grapes";
console.log(x); // Output: [ 'Apple', 'Orange', 'Mango', 'Grapes' ]
</script>
```

```
<script>
x = ["Apple", "Orange", "Mango"];
x[6] = "Grapes";
console.log(x);
// Output: [ 'Apple', 'Orange', 'Mango', <3 empty items>, 'Grapes' ]
</script>
```

```
<script>
// Create a new array named "x" containing 6 numbers
x = new Array(50, 80, 15, 25, 65, 70);
console.log(x);
// Output: [ 50, 80, 15, 25, 65, 70 ]
</script>
```

Indentation:

The spaces at the beginning of a code line are termed as indentation. It makes the code easier to read

```
<script>

names = ["Alan", "Albert", "John"]; // an array with 3 elements

console.log(typeof names); // Output: object



// JavaScript array is an object

</script>
```

```
<script>

names = ["Alan", "Albert", "John"];

console.log(Array.isArray(names)); // Output: true

</script>
```

```
<script>

/*

Whenever an object is created by a constructor,

the instanceof operator returns true.

*/



names = ["Alan", "Albert", "John"];

console.log(names instanceof Array); // Output: true



</script>
```

When applied to an array, the instanceof operator returns a true result

```
<script>

// All array elements are combined into a string by the join() method

names = ["Alan", "Albert", "John"];

console.log(names.join(" * ")); // Output: Alan * Albert * John

</script>
```

```
<script>

// The last element from an array is removed by the pop() method

names = ["Alan", "Albert", "John"];

names.pop();

console.log(names); // Output: [ 'Alan', 'Albert' ]

</script>
```

```
<script>


/* The unshift() method inserts "Mary" as a

new element at the beginning of an array

*/


names = ["Alan", "Albert", "John"];

names.unshift("Mary");

console.log(names); // Output: [ 'Mary', 'Alan', 'Albert', 'John' ]


</script>
```

```
<script>

names = ["Alan", "Albert", "John"];

delete names[0];

console.log(names);

// Output: [ <1 empty item>, 'Albert', 'John' ]

</script>
```

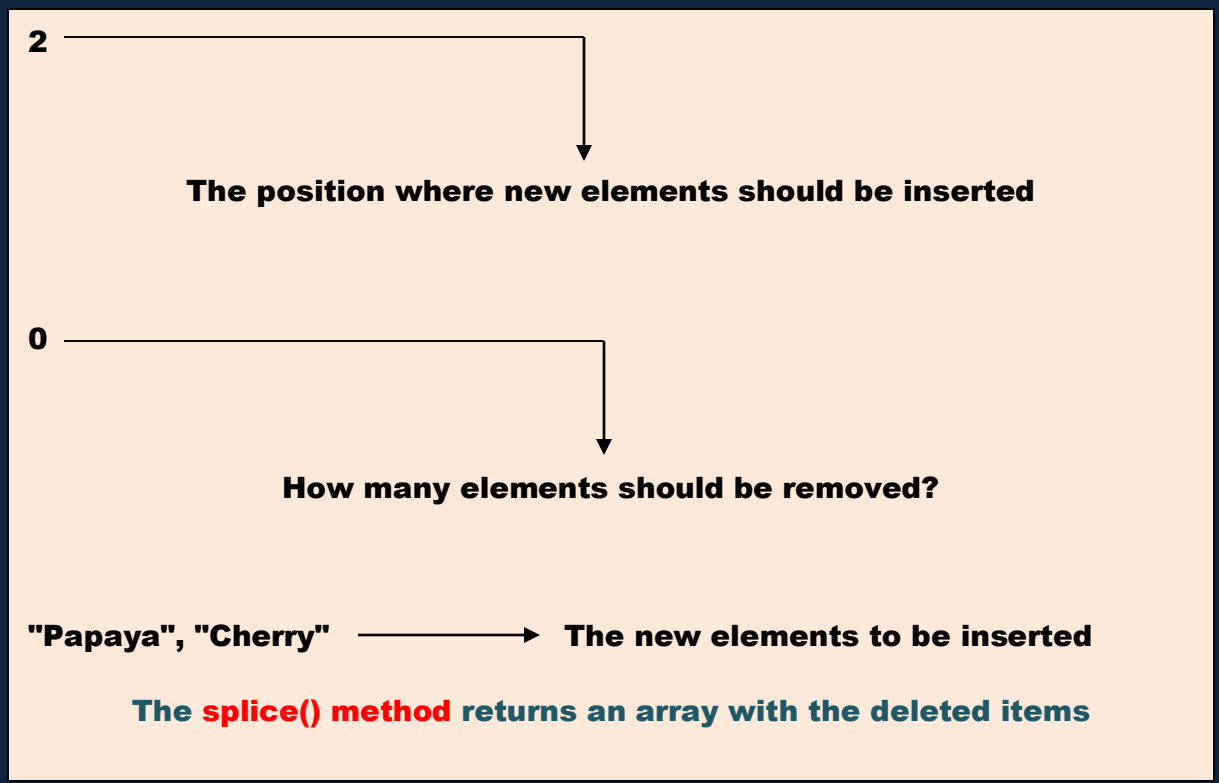
```
<script>

// Merging 2 Arrays "x" and "y"

x = ["Apple", "Orange"];

y = ["Grapes", "Lemon", "Mango"];

console.log(x.concat(y));

// Output: [ 'Apple', 'Orange', 'Grapes', 'Lemon', 'Mango' ]

</script>
```

```
<script>

// Merging 3 Arrays "x", "y" and "z"



x = ["Apple", "Orange"];

y = ["Grapes", "Lemon", "Mango"];

z = ["Pear", "Raspberries", "Banana"];

console.log(x.concat(y, z));

</script>
```

```
[ 'Apple',
  'Orange',
  'Grapes',
  'Lemon',
  'Mango',
  'Pear',
  'Raspberries',
  'Banana' ]
```

```
<script>

x = ["Apple", "Orange"];

console.log(x.concat("Mango"));

// Output: [ 'Apple', 'Orange', 'Mango' ]

</script>
```

```
<script>

x = ["Apple", "Orange", "Mango", "Pear"];

x.splice(2, 0, "Papaya", "Cherry")

console.log(x);

// Output: [ 'Apple', 'Orange', 'Papaya', 'Cherry', 'Mango', 'Pear' ]

</script>
```



```
<script>

x = ["Apple", "Orange", "Mango", "Pear"];

x.splice(2, 2, "Papaya", "Cherry")

console.log(x);

// Output: [ 'Apple', 'Orange', 'Papaya', 'Cherry' ]

</script>
```

```
<script>

// The toString() method turns an array into a string separated by commas

x = ["Apple", "Orange", "Mango", "Pear"];

console.log(x.toString());

// Output: Apple,Orange,Mango,Pear

</script>
```

```
<script>



// The sort() method sorts an array alphabetically



x = ["Apple", "Grapes", "Banana", "Mango"];

console.log(x.sort());

// Output: [ 'Apple', 'Banana', 'Grapes', 'Mango' ]



</script>
```

```
<script>


x = [50, 60, 10, 30, 40, 20];


// Sort the array in ascending order

x.sort(function(a, b){return a - b});

console.log(x);

// Output: [ 10, 20, 30, 40, 50, 60 ]


// Sort the array in descending order

x.sort(function(a, b){return b - a});

console.log(x);

// Output: [ 60, 50, 40, 30, 20, 10 ]


</script>
```

```
<script>

x = [50, 60, 10, 30, 40, 20];

// Display the highest number in an array

console.log(Math.max.apply(null, x)); // Output: 60

// Display the lowest number in an array

console.log(Math.min.apply(null, x)); // Output: 10

</script>
```

```
<script>

console.log(Math.E);  // Displays Euler's number

// Output: 2.718281828459045

console.log(Math.PI);   // Displays PI

// Output: 3.1415926535897935

console.log(Math.SQRT2);  // Displays the square root of 2

// Output: 1.4142135623730951

console.log(Math.SQRT1_2);  // Displays the square root of 1/2

// Output: 0.7071067811865476

console.log(Math.LN2);  // Displays the natural logarithm of 2

// Output: 0.6931471805599453

console.log(Math.LN10); // Displays the natural logarithm of 10

// Output: 2.302585092994046

console.log(Math.LOG2E);  // Displays base 2 logarithm of E

// Output: 1.4426950408889634

console.log(Math.LOG10E);  // Displays base 10 logarithm of E

// Output: 0.4342944819032518

</script>
```

```
<script>



// Math.floor() returns the value of "x" rounded down to its nearest integer

x = 4.6;

console.log(Math.floor(x));

// Output: 4



// Math.ceil() returns the value of "x" rounded up to its nearest integer

console.log(Math.ceil(x));

// Output: 5



</script>
```

```
<script>

// Math.trunc() returns the integer part of "x"



x = 4.6;

console.log(Math.trunc(x));  // Output: 4

y = -4.6;

console.log(Math.trunc(y));  // Output: -4



</script>
```

```
<script>

// Math.sqrt() method returns the square root of a number "16"

console.log(Math.sqrt(16)); // Output: 4

// Math.min() method returns the lowest value in a list

console.log(Math.min(60, 30, 50, 20, 10, 40)); // Output: 10

// Math.max() method returns the highest value in a list

console.log(Math.max(60, 30, 50, 20, 10, 40)); // Output: 60

// Math.random() method returns a random number between 0 and 1

console.log(Math.random()); // Output: 0.6239112970213929

// Math.log() method returns the natural logarithm of a number "2"

console.log(Math.log(2));  // Output: 0.6931471805599453
```


```
// Math.log2() method returns the base 2 logarithm of a number "16"

console.log(Math.log2(16)); // Output: 4
```


```
// Math.log10() method returns the base 10 logarithm of a number "100"

console.log(Math.log10(100)); // Output: 2


</script>
```

```
<script>

// Math.cbrt() method returns the cubic root of a number "81"

console.log(Math.cbrt(81)); // Output: 4.326748710922225

</script>
```

```
<script>

// Displays a random integer from 0 to 10

console.log(Math.floor(Math.random() * 11));

// Displays a random integer from 0 to 100

console.log(Math.floor(Math.random() * 101));

// Displays a random integer from 1 to 10

console.log(Math.floor(Math.random() * 10) + 1);

// Displays a random integer from 1 to 100

console.log(Math.floor(Math.random() * 100) + 1);

</script>
```

```
<script>

console.log(Boolean(10 > 9)); // Output: true

console.log(10 > 9);  // Output: true


// The Boolean value of 1 is true

console.log(Boolean(1)); // Output: true

// The Boolean value of 0 is false

console.log(Boolean(0)); // Output: false

</script>
```

```
var x = true;

var y = new Boolean(true);


// typeof "x" returns boolean

// typeof "y" returns object
```

```
<script>

var x = 19;

console.log((x < 18) ? "x<18":"x>18"); // Output: x>18

</script>
```

If the variable "x" is a value below 18, then x<18 is displayed, otherwise x>18 is displayed

JavaScript if – else statement

```
<script>

var x=4;

var y=6;

if(x>y) {

document.write("x is greater than y");

} else {

document.write("y is greater than x");

}

// Output: y is greater than x

</script>
```

Currying

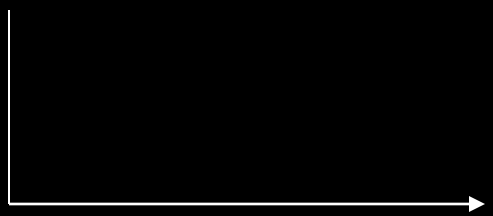
The method of breaking down a function with several arguments into a collection of functions with just one argument each

JavaScript else-if statement

```
<script>

var x=4;

var y=6;

var z=12;

if(x>y&&x>z) {

document.write("x is greater than y and z");

} else if (y>x&&y>z) {

document.write("y is greater than x and z");

} else {

document.write("z is greater than x and y");

} // Output: z is greater than x and y

</script>
```

JavaScript for loop

```
<script>

var i;

for(i=1; i<=10; i++)

document.write(i);

</script>
```



Output:

1 2 3 4 5 6 7 8 9 10

```
<script>

var i=1;

do {

document.write(i++);

} while(i<=10)

</script>
```

JavaScript do-while loop

Output:

1

2

3

4

5

6

7

8

9

10

```
<script>

var i=1;

while(i<=10) {

  console.log(i++);

}

</script>
```

JavaScript while loop

```
<script>

x = [85, 6, 9, 36, 3];

for(i in x) {

  console.log(x[i]);

}

</script>
```

Output:

85

6

9

36

3

JavaScript Hoisting

The **method** by which, prior to code execution, the **interpreter** seems to shift the definition of functions, variables, or classes to the top of their scope. Before a **function is declared**, **hoisting** enables safe use of that function in the code

```
<script>

var ch ='2';

switch(ch) {

case '1':

document.write("Red");

break;

case '2':

document.write("White");

break;

case '3':

document.write("Yellow");

break;

case '4':

document.write("Green");

break;

default:

document.write("Error");

break;

}

</script>
```

JavaScript Switch case

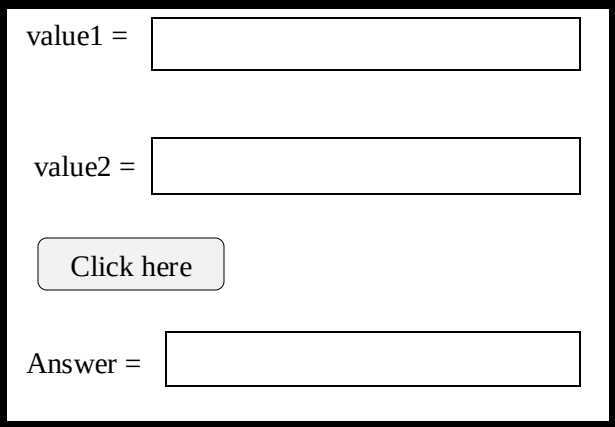
Output:

White

```
<script>

var x = 6;

var y = 2;

if(x > y) {

alert("Albert");

}

</script>
```

alert "Albert" if x is greater than y

```
<!DOCTYPE html>


<html>

<head>

<script>

function addNumbers() {

var x = parseInt(document.getElementById("value1").value);

var y = parseInt(document.getElementById("value2").value);

var z = document.getElementById("answer");

z.value = x + y;

}

</script>

</head>



<body>

value1 = <input type ="text" id="value1"/>

<br> <br> value2 = <input type ="text" id="value2"/>

<br> <br> <input type="button" value="Click here" onclick="addNumbers()"/>

<br> <br> Answer = <input type="text" id = "answer"/>

</body>



</html>
```



The above program demonstrates how to get **two inputs** from the user (i.e., **value1** and **value2**) and have a button (i.e., **Click here**) on the page to call a function:

```
addNumbers()
```

to process the inputs and output the **answer** on the **console screen**.

```
var x = parseInt(document.getElementById("value1").value);
```

Creating the variable "x" and assigning it a value

```
parseInt(document.getElementById("value1").value)
```

```
var y = parseInt(document.getElementById("value2").value);
```

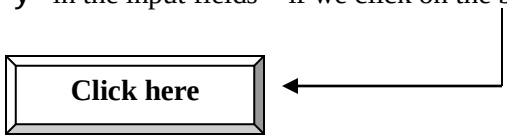
Creating the variable "y" and assigning it a value

```
parseInt(document.getElementById("value2").value)
```

parseInt(document.getElementById("value1").value) → get the integer value for "x" entered in the input field which is defined with id "value1"

parseInt(document.getElementById("value2").value) → get the integer value for "y" entered in the input field which is defined with id "value2"

After entering the values for "x" and "y" in the input fields – if we click on the button



"on click" in the statement:

```
<input type="button" value="Click here" onclick="addNumbers()"/>
```

call the **function** addNumbers() → within which

```
var z = document.getElementById("answer");
z.value = x + y;
```

Statements

are executed – i.e., the entered values for "x" and "y" are added and the **answer** is entered in the input field which is defined with id "answer".

```
<script>

for(var i=1; i<=10; i++) {

if(i==5) {

break;

}

  console.log(i);

}

</script>
```

Output:

1

2

3

4

```
<script>

for(var i=1; i<=10; i++) {

if (i==5) {

continue;

}

  console.log(i);

}

</script>
```

Output

```
1
2
3
4
6
7
8
9
10
```

```
<script>

// for..of loop iterate over the elements of a string "x"

x = "Alan";

for(i of x) {

    console.log(i);

}

</script>
```

Output:

```
A
l
a
n
```

```
<script>

// for..of loop iterate over the elements of an array "x"

x = ["Albert", "John", "James"];

for(i of x) {

    console.log(i);

}

</script>
```

Output:

```
Albert
John
James
```

```
<script>

// for..of loop iterate over the elements of a Set "x"

x = new Set(["A","B","C"]);

for(i of x) {

    console.log(i);

}

</script>
```

Output:

A

B

C

```
<script>

// for..of loop iterate over the elements of a Map "x"



 x = new Map([

  ["Albert", 56],

  ["John", 85],

  ["Elsa", 45]

]);

for (i of x) {

    console.log(i);

}



</script>
```

Output:

['Albert', 56]

['John', 85]

['Elsa', 45]

```
<script>

// Create a Set "x"

x = new Set();

// Add Values to the Set

x.add("A");

x.add("B");

x.add("C");

console.log(x); // Output: Set { 'A', 'B', 'C' }

</script>
```

```
<script>

// Create a Set "x"

x = new Set();

// Create Variables

var a = "A";

var b = "B";

var c = "C";

// Add Variables to the Set using add() Method

x.add(a);

x.add(b);

x.add(c);


console.log(x); // Output: Set { 'A', 'B', 'C' }

</script>
```

```
<script>

// Create a Map "x"

x = new Map();


// Add elements to a Map "x" with the set() method

x.set("Albert", 56);

x.set("John", 38);

x.set("Elsa", 46);

console.log(x); // Output: Map { 'Albert' => 56, 'John' => 38, 'Elsa' => 46 }


// The get() method gets the value of a key "Albert" in a Map

console.log(x.get("Albert")); // Output: 56


// Display the number of elements in a Map

console.log(x.size); // Output: 3


// The has() method returns true if a key "Albert" exists in a Map

console.log(x.has("Albert")); // Output: true


// The delete() method removes a Map element "Albert"

x.delete("Albert");

console.log(x); // Output: Map { 'John' => 38, 'Elsa' => 46 }


console.log(x.has("Albert")); // Output: false

</script>
```

```
<script>

"use strict";

x = 5.496;

console.log(x); // Because x is not declared, this will result in an error

</script>
```

Prevent us from using **undeclared** variables

```
<script>

"use strict";

var x = 5.496;

console.log(x); // Output: 5.496

</script>
```

```
<script>

var add = (x, y) => x + y;

console.log(add(4, 5)); // Output: 9


var mult = (x, y) => x * y;

console.log(mult(4, 5)); // Output: 20


var x = () => "Albert";

console.log(x()); // Output: Albert

</script>
```

One of the new features in the **ES6 edition of JavaScript** is the **arrow function**. Compared to conventional functions, it enables us to create functions in a cleaner manner

PHP

Paradigm	Imperative, functional, object-oriented, procedural, reflective
Designed by	Rasmus Lerdorf
Developer	The PHP Development Team, Zend Technologies
First appeared	1995; 25 years ago
Stable release	7.4.12 / 29 October 2020; 3 days ago
Preview release	8.0.0 Release Candidate 3 / October 30, 2020; 2 days ago
Typing discipline	Dynamic, weak **since version 7.0:** Gradual
Implementation language	C (primarily; some components C++)
OS	Unix-like, Windows
License	PHP License (most of Zend engine under Zend Engine License)
Filename extensions	`.php`, `.phtml`, `.php3`, `.php4`, `.php5`, `.php7`, `.phps`, `.php-s`, `.pht`, `.phar`
Website	www.php.net

Rasmus Lerdorf

Creator of PHP

Major implementations
Zend Engine, HHVM, Phalanger, Quercus, Parrot
Influenced by
Perl, HTML, C, C++, Java, Tcl, JavaScript, Hack
Influenced

Hack, JSP, ASP

PHP / Hypertext Preprocessor (designed by an **Greenlandic-Danish programmer** "**Rasmus Lerdorf**" in 1994 – as an efficient alternative to other scripting languages like Ruby, Perl and **Microsoft's ASP**) is an relatively free (**not licensed by a major corporation**) popular efficient server side programming language (and relatively easy one to master and quick to learn) that carries out common website duties like accepting passwords, authenticating users, and managing forum posts and guest books. Without **PHP** Facebook, Yahoo, Google wouldn't have existed. **PHP** is a server scripting language that may be used to create dynamic, engaging websites.

PHP Keywords

__halt_compiler()	abstract	and	array()	as
break	callable **(as of PHP 5.4)**	case	catch	class
clone	const	continue	declare	default
die()	do	echo	else	elseif
empty()	enddeclare	endfor	endforeach	endif
endswitch	endwhile	eval()	exit()	extends
final	finally **(as of PHP 5.5)**	fn **(as of PHP 7.4)**	for	foreach
function	global	goto **(as of PHP 5.3)**	if	implements
include	include_once	instanceof	insteadof **(as of PHP 5.4)**	interface
isset()	list()	namespace **(as of PHP 5.3)**	new	or
print	private	protected	public	require
require_once	return	static	switch	throw
trait **(as of PHP 5.4)**	try	unset()	use	var
while	xor	yield **(as of PHP 5.5)**	yield from **(as of PHP 7.0)**	

PHP Comments

Code:

```
<?php

// This is a single-line comment

# This is also a single-line comment

/*
This is a
multiple-line
comment
*/

?>
```

Comments are lines of code that are not executed when a program is executed and are typically written inside a block of PHP code to describe **how the code works**. **PHP comments** are only visible while inspecting the **PHP code** in a file; they are not visible to end users

PHP Variables

Code:

```
<?php // A PHP script starts with <?php and ends with ?>

$x = 5 /* + 12 */ + 0;
$y = 5;
echo $x + $y; # Display the sum of $x and $y

?>
```

Declaring and assigning values to Variables

Output on the screen:

```
10
```

Code:

```php
<?php

$txt = "World";
echo "Hello $txt!";

?>
```

In PHP, a variable starts with the **$ sign**, followed by the name of the variable

Output on the screen:

Hello World!

PHP echo and print Statements

Code:

```php
<?php

$txt = "World";
echo "Hello $txt!";
print "Hello $txt!";

?>
```

```php
<?php
# Strip the string from HTML tags
echo strip_tags("Elsa <b>Einstein</b>");
// Output: Elsa Einstein
?>
```

Output on the screen:

Hello World!Hello World!

PHP $GLOBALS

Code:

```
<?php
$x = 75;
$y = 25;

function addition() {
  $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}
addition();echo
$z;
?>
```

Global variables

The superglobal variable **$GLOBALS** is used to access global variables from anywhere in the PHP program. All global variables are stored in PHP in an array named **$GLOBALS[index]**. The **index** holds the name of the variable

Output on the screen:

```
100
```

PHP Data Types

Code:

```
<?php
$x = 7982;
$y = 798.2;
var_dump($x);
var_dump($y);
?>
```

// The **var_dump()** function returns the data type and value of a variable

Output on the screen:

```
int(7982)
float(798.2)
```

PHP Strings

Code:

```
<?php
echo strlen("Hello world!");
?>
```

Return the length of the string "Hello world!"

Output on the screen:

12

Code:

```
<?php
echo str_word_count("Alan Turing");
?>
```

Return the number of words found in the string "Alan Turing"

Output on the screen:

2

Code:

```
<?php
echo strrev("Alan Turing");
?>
```

Reverse the string "Alan Turing"

Output on the screen:

```
gniruT nalA
```

Code:

```php
<?php
echo str_replace("world", "Computer", "Hello world!");
?>
```

Replace the word "world" in the string "Hello world!" with "Computer"

Output on the screen:

```
Hello Computer!
```

PHP Numbers

Code:

```php
<?php

$x = 8885;
var_dump(is_int($x));

$y = 86.85;
var_dump(is_int($y));

$z = 86.85;
var_dump(is_float($z));

?>
```

Check whether a variable is of type integer or not

Check whether a variable is of type float or not

Output:

```
bool(true)
bool(false)
bool(true)
```

PHP Math

Code:

```
<?php
echo(pi());
?>
```

The pi() function returns the value of **PI**

Output on the screen:

```
3.1415926535898
```

Code:

```
<?php
echo(min(0, 150, 30, 20, -8, -200));
echo "\n";
echo(max(0, 150, 30, 20, -8, -200));
?>
```

Display the lowest value in a list

Display the highest value in a list

Output on the screen:

```
-200
150
```

Code:

```
<?php
echo(abs(-6.27));
?>
```

The abs() function returns the absolute (positive) value of a number

Output on the screen:

```
6.27
```

Code:

```
<?php
echo(sqrt(16));
?>
```

The sqrt() function returns the square root of a number

Output on the screen:

```
4
```

Code:

```
<?php
echo(round(4.8));
?>
```

The round() function rounds a floating-point variable to its nearest integer

Output on the screen:

```
5
```

Code:

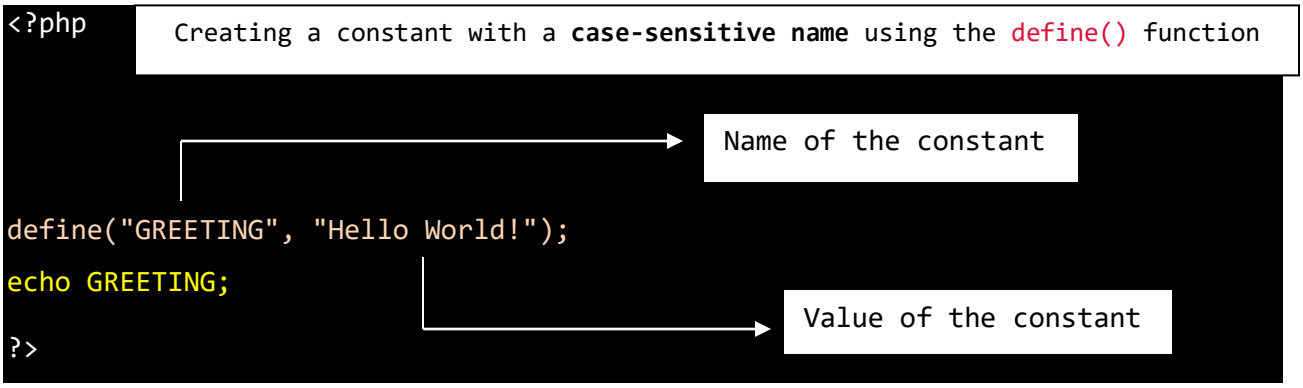
```
<?php
echo(rand(10, 100));
?>
```

The rand() function generates a random number between 10 and 100

Output on the screen:

PHP Constants

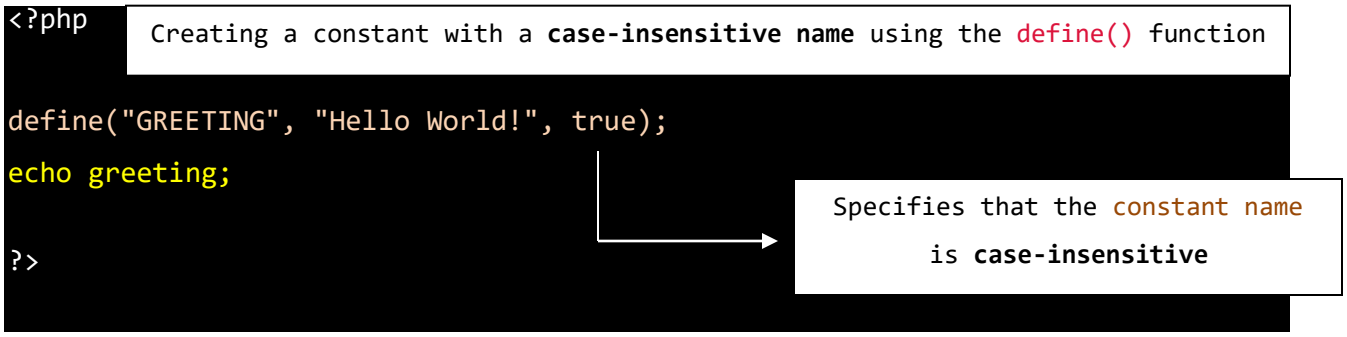
Code:



Output on the screen:

```
Hello World!
```

Code:



Output on the screen:

```
Hello World!
```

Code:

```php
<?php
define("fruits", [
  "Apple",
  "Mango",
  "Orange"
]);
echo fruits[1];
?>
```

Creating an Array constant using the define() function

Output on the screen:

Mango

Code:

```php
<?php

define("GREETING", "Hello World!");

function myfunc() {
  echo GREETING;
}

myfunc();

?>
```

Even though a **constant** in this program is defined outside of a function, the function still uses the constant

Output on the screen:

Hello World!

PHP Operators

Code:

```php
<?php
// Convert the first character of each word to uppercase
echo ucwords("albert einstein");
# Output: Albert Einstein
?>
```

```php
<?php


$x = 12;
$y = 6;


echo $x + $y; // Addition
echo "\n";
echo $x - $y; // Subtraction
echo "\n";
echo $x / $y; // Division
echo "\n";
echo $x * $y; // Multiplication
echo "\n";
echo $x % $y; // Modulus

?>
```

Output on the screen:

```
18
6
2
72
0
```

Code:

```
<?php
$num1=2;
$num2=3;
$num3 = $num1 + 1; // Increments $num1 by one
$num4 = $num1 - 1; // Decrements $num1 by one
$num5 = $num2 + 1; // Increments $num2 by one
$num6 = $num2 - 1; // Decrements $num2 by one
echo ("\n The incremented value of $num1 = $num3");
echo ("\n The decremented value of $num1 = $num4");
echo ("\n The incremented value of $num2 = $num5");
echo ("\n The decremented value of $num2 = $num6");

?>
```

Output on the screen:

```
The incremented value of 2 = 3
The decremented value of 2 = 1
The incremented value of 3 = 4
The decremented value of 3 = 2
```

PHP if Statement

Code:

```php
<?php

$x = 25;

// Output "Albert Einstein" if $x is less than 21

if($x > 21) {
  echo "Albert Einstein";
}

?>
```

Output on the screen:

```
Albert Einstein
```

PHP if – else Statement

Code:

```php
<?php
$x = 4.5;

$y=5;

if($x>$y) {
echo ("x is greater than y");

} else {

echo ("y is greater than x");

}

?>
```

Output "x is greater than y" if $x is greater than $y, and "y is greater than x" otherwise

Output on the screen:

```
y is greater than x
```

PHP else if Statement

Code:

```php
<?php

$x = 4.5;

$y = 5;

$z = 6;

if ($x>$y && $x>z) {

echo ("$x is greater than $y and $z");

} else if ($y>$z && $y>x) {

echo ("$y is greater than $x and $z");

}

else {

echo ("$z is greater than $x and $y");

}

?>
```

```php
<?php

/*
 Replace the characters "ALBERT" (case-insensitive)
 in the string "Albert Einstein" with "Elsa"
*/

echo str_ireplace("ALBERT", "Elsa", "Albert Einstein");
# Output: Elsa Einstein
?>
```

Output on the screen:

```
6 is greater than 4.5 and 5
```

```php
<?php

$x = 5496;

var_dump(is_numeric($x));

# Output: bool(true)

?>
```

Check if the variable is numeric

PHP switch Statement

Code:

```php
<?php

$ch ='3';

switch($ch) {

case '1':

echo "Red";

break;

case '2':

echo "White";

break;

case  '3':

echo "Yellow";

break;

case '4':

echo "Green";

break;

default:

echo "Error";

break;

}

?>
```

The switch statement in PHP is used to execute a single statement from multiple conditions

```php
<?php

$x = 54.960;

/*
   Prints the value of above integer
   variable as a numeric string
*/

echo strval($x);

?>
```

Output on the screen:

```
Yellow
```

PHP Loops

Code:

```
<?php
$i = 1;
while($i <= 10) {
echo "\n $i ";
$i++;
}
?>
```

PHP while Loop

Output on the screen:

```
1
2
3
4
5
6
7
8
9
10
```

```
<?php
# Break a string "x" into an array
$x = "Albert Einstein";
print_r(explode(" ",$x));
?>
```

Output:

```
Array
(
    [0] => Albert
    [1] => Einstein
)
```

Code:

```
<?php
$i = 1;
do {
echo "\n $i ";
$i++;
} while($i <= 9);
?>
```

PHP do-while Loop

Output on the screen:

```
Albert Albert Albert

eAtrbl
```

Code:

```php
<?php
# Repeat the string "Albert" 3 times
echo str_repeat("Albert ",3);
// Output: Albert Albert Albert
?>

<?php
# Randomly shuffle all characters of a string "Albert"
echo str_shuffle("Albert");
// Output: eAtrbl
?>
```

```php
<?php

for($i=1; $i<=10; $i++)
echo ("\n $i");
?>
```

PHP for Loop

Output on the screen:

```
1
2
3
4
5
6
7
8
9
10
```

```php
<?php

// Convert all characters to Lowercase
echo strtolower("ALBERT");
# Output: albert

echo "\n";

// Convert all characters to Uppercase
echo strtoupper("albert");
# Output: ALBERT

?>
```

Code:

PHP foreach Loop

```
<?php
$fruits = array("Apple", "Mango", "Kiwi", "Orange");

foreach ($fruits as $value) {
  echo "$value \n";
}
?>
```

Output on the screen:

```
Apple
Mango
Kiwi
Orange
```

```
<?php

# Convert the first character of "alan" to uppercase
echo ucfirst("alan");
// Output: Alan

echo "\n";

# Convert the first character of "Alan" to lowercase
echo lcfirst("Alan");
// Output: alan

?>
```

Code:

PHP Break

```
<?php

for($x = 0; $x < 6; $x++) {
  if($x == 4) {
    break;
  }
  echo "$x \n";
}

?>
```

This program jumps out of the loop when "x" is equal to 4

Output on the screen:

```
0
1
2
3
```

Code:

```
<?php
for($x = 0; $x < 6; $x++) {
  if ($x == 4) {
    continue;
  }
  echo "$x \n";
}
?>
```

PHP Continue

This program skips the value of "4"

Output on the screen:

```
0
1
2
3
5
```

```
<?php
$x = "Albert";
var_dump(is_string($x));
# Output: bool(true)
?>
```

Check whether a variable "x" is of type **string** or not

PHP Functions

Code:

```php
<?php
function addition($a, $b) {
return $a + $b;
}
$sum = addition(4, 3);
echo "The sum of two numbers = $sum ";
?>
```

```php
<?php
function myfunc() {
  echo "Albert Einstein";
}

myfunc(); // call the function
# Output: Albert Einstein

?>
```

Output on the screen:

```
The sum of two numbers = 7
```

PHP Arrays

Code:

```php
<?php
$x = array("Apple", "Orange", "Mango");

echo "I like " . $x[0] . ", " . $x[1] . " and " . $x[2] . " fruits";
?>
```

An array stores multiple values in one single variable

Output on the screen:

```
I like Apple, Orange and Mango fruits
```

Code:

```php
<?php

$x = array("Apple", "Orange", "Mango");

echo count($x);

?>
```

The array() function is used to create an array

Returns the length (the number of elements) of an array

Output on the screen:

3

```php
<?php

$x = array("Apple", "Orange", "Mango");

var_dump(is_array($x));

# Output: bool(true)

?>
```

Check whether a variable is an array or not

```php
<?php
$x = NULL;
var_dump(is_null($x));
# Output: bool(true)
?>
```

Check whether a variable is NULL or not

```php
<?php
$x = true; // boolean value
var_dump(is_scalar($x)); # Output: bool(true)
$y = 5; // integer value
var_dump(is_scalar($y)); # Output: bool(true)
?>
```

Check whether a variable is a scalar or not

```php
<?php

# Remove the backslash in front of "Einstein"

echo stripcslashes("Albert \Einstein");

// Output: Albert Einstein

?>
```

```php
<?php

# Return the ASCII value of "B"

echo ord("B");

// Output: 66

?>
```

```php
<?php

// Add a backslash in front of each double quote

$x = addslashes('Elsa "Albert" Einstein');

echo($x);

# Output: Elsa \"Albert\" Einstein

?>
```

```php
<?php

// Split the string after each character and add "*" after each split

$x = "Alan";

echo chunk_split($x,1,"*");

# Output: A*l*a*n*

?>
```

Computer network

A system that carries information between 2 or more entities, in the form of electric signals

Local Area Network (LAN)

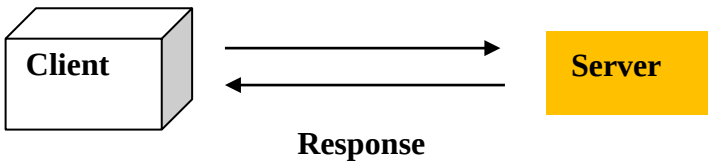
Network inside your home

Wide Area Network (WAN)

Network between Towns, states, countries

Antifragile

A characteristic property that enables software systems to become more capable or perform better when they experience stress, errors, malfunctions, or failures



Continuous Quality

A rigorous strategy to identify and correct software application flaws at every stage of the software development cycle

Agile Project Management

A methodology of managing software development projects that iteratively emphasizes frequent releases and incorporates customer feedback at every stage

- **HTTP**: Hypertext Transfer Protocol
- **URL**: Uniform Resource Locator "**http://www.myw3schools.com**"
- **HTML**: Hypertext Markup Language
- **VRML**: Virtual Reality Markup Language

Computer vision → Write computer programs that can interpret images

Optical character recognition (**OCR**) — Technology to convert scanned docs to text

Root Cause Analysis: The process of identifying the root cause of problems in order to prevent them from happening

Staging Environment: A nearly exact copy of a production environment used for the testing of the software application to make sure it works properly before deployment

Face detection → **Technology to detect faces**

Binary	0 or 1
Octal	(0 −7)
Hexadecimal	(0 – 9) + (A− F)
Decimal	(0 – 9)

Booleans

- 0 = false = no
- 1 = true = yes

Provisioning

The process of setting up IT infrastructure

Model Based Testing

A software testing technique where run time behavior of software application under testing is examined against predictions made by a model

- Requirements analysis
- Design
- Construction
- Testing

Software Project Management

The phrase "internet surfing" was coined by a librarian named Jean Armour Polly in 1992

Symbolics.com was the first registered domain

Internet peer-to-peer (**P2P**)

Enables users to connect to each other's hard disks and exchange files directly

Self-Service Deployment

Automating deployment procedures to the extent where developers can give direct deployment authority to project managers or even clients

Python code:

```
# prints decimal value '144' in binary form
print("{0:b}".format(144)) # Output: 10010000
# prints decimal value '144' in octal form
print("{0:o}".format(144)) # Output: 220
# prints decimal value '144' in hexadecimal form
print("{0:x}".format(144)) # Output: 90
```

Apache ZooKeeper

An open source Apache project that offers a centralized service for configuring large clusters of distributed systems and offering identification, synchronization and group services

Description:

Find files edited more than 3 days ago.

Command:

```
find . -type f -mtime +3
```

Description:

Find files edited in the last 24 hours.

Command:

```
find . -type f -mtime -1
```

```
print("{:5d}".format(16))
print("{:<5d}".format(16))
print("{:<05d}".format(16))
print("{:>5d}".format(16))
print("{:>05d}".format(16))
print("{:^5d}".format(16))
print("{:=5d}".format(-16))
print("{:^10.3f}".format(16.45556))
print("{:=8.3f}".format(-16.45556))
```

Output:

```
   16
16
16000
   16
00016
 16
-  16
  16.456
- 16.456
```

Description:

Find files that have more than 100 characters (bytes) in them.

Command:

```
find . -type f -size +100c
```

Description:

Find files bigger than 100 KB but smaller than 1 MB.

Command:

```
find . -type f -size +100k -size -1M
```

Description:

Deletes all the files edited in the last 24 hours.

Command:

```
find . -type f -mtime -1 -delete
```

Description:

List all files including hidden files.

Command:

```
print("{:*^5}".format("Alan")) # Output: Alan*

- print("{:.3}".format("albert"))
- print("{:5.3}".format("albert"))
- print("{:>5.3}".format("albert"))
- print("{:^5.3}".format("albert"))
- print("{:*^5.3}".format("albert"))

Output:

alb
alb
  alb
 alb
*alb*
```

```
ls -a
```

Description:

List Files and Directories with "/" Character at the End.

Command:

```
ls -F
```

```
names=["John", "Alan", "Albert", "Mary"]
print([i[0] for i in names])
# Output: ['J', 'A', 'A', 'M']
```

Description:

List Files in Reverse Order.

Command:

```
ls -r
```

```
x="{:{a}{b}{c}}"
print(x.format('john', a='*',b='^',c=5))
# Output: john*
print(x.format('john', a='*',b='^',c=6))
# Output: *john*
print(x.format('john', a='*',b='<',c=6))
# Output: john**
print(x.format('john', a='*',b='>',c=6))
# Output: **john
```

Description:

Sort Files by File Size.

Command:

```
ls -lS
```

Description:

List Files with an inode number.

```
x="alan turing".split()
print(x)
print([[i.upper(),len(i)] for i in x])

# Output:
      ['alan', 'turing']
      [['ALAN', 4], ['TURING', 6]]
```

Command:

```
ls -i
```

Description:

Check the version of the ls command.

Command:

```
ls --version
```

```
a=["Alan", "Albert", "John"]
b=["Alan", "Albert", "John"]
c=["ALAN","ALBERT","JOHN"]
print(a==b) # Output: True
print(a==c)  # Output: False
print(a != c)  # Output: True
```

Description:

List files under directory /tmp.

Command:

```
ls -l /tmp
```

```
a={100,200,300,400}
b={300,400,500,600}
print(a.union(b))
# Output: {100, 200, 300, 400, 500, 600}
print(a|b)
# Output: {100, 200, 300, 400, 500, 600}
```

Description:

Display UID and GID of files and directories.

Command:

```
ls -n
```

```
x=(100,200,300,400,500,600)
print(x[2:1000])
# Output: (300, 400, 500, 600)
print(x[1000])
# Output: IndexError: tuple index out of range
```

Description:

Find all 30 MB files.

Command:

```
find / -size 30M
```

```
x={100,200,300,400,500,600}
x.remove(300)
print(x)  # Output: {100, 200, 400, 500, 600}
x.discard(300)
print(x)  # Output: {100, 200, 400, 500, 600}
```

Description:

Find files with sizes between 100 - 200MB.

Command:

```
find / -size +100M -size -200M
```

Description:

List directories larger than 20 KB.

```
a={100,200,300,400}
b={300,400,500,600}
print(a.intersection(b))
# Output: {400, 300}
print(a&b)
# Output: {400, 300}
```

Command:

```
find / -type d -size +20k
```

Description:

Find empty files and directories.

```
a={100,200,300,400}
b={300,400,500,600}
print(a.difference(b))
# Output: {200, 100}
print(a-b)
# Output: {200, 100}
print(b.difference(a))
# Output: {600, 500}
print(b-a)
# Output: {600, 500}
```

Command:

```
find ./ -type f -size 0
```

Description:

List files modified within the last 17 hours.

Command:

```
find . -mtime -17 -type f
```

```
a={100,200,300,400}
b={300,400,500,600}
print(a.symmetric_difference(b))
# Output: {100, 200, 500, 600}
print(a^b)
# Output: {100, 200, 500, 600}
print(b.symmetric_difference(a))
# Output: {100, 200, 500, 600}
print(b^a)
# Output: {100, 200, 500, 600}
```

Description:

List directories modified within the last 10 days.

Command:

```
find . -mtime -10 -type d
```

```
print({i*i for i in range(4)})
# Output: {0, 1, 4, 9}
print({2**i for i in range(4)})
# Output: {8, 1, 2, 4}
```

Description:

List all files modified between 6 and 15 days ago in the home directory.

Command:

```
find /home -type f -mtime +6 -mtime -15
```

Description:

Display files with permission 777.

```
a=lambda i:i*i
print(a(2)) # Output: 4
print(a(6)) # Output: 36
```

Command:

```
find -perm 777
```

```
x=open("1.txt",'w')
print(x.name) # File name
# Output: 1.txt
print(x.mode) # File mode
# Output: w
print(x.readable()) # Is File readable
# Output: False
print(x.writable()) # Is File writable
# Output: True
print(x.closed) # Is File closed
# Output: False
x.close()
print(x.closed) # Is File closed
# Output: True
```

Description:

List files owned by a user (manju).

Command:

```
find /home -user manju
```

Description:

Find all text files owned by user "manju".

Command:

```
find /home -user manju -iname "*.txt"
```

Description:

Find and list files and directories together with their permissions.

Command:

```
find -name "*.conf" | ls -l
```

Description:

List directories only.

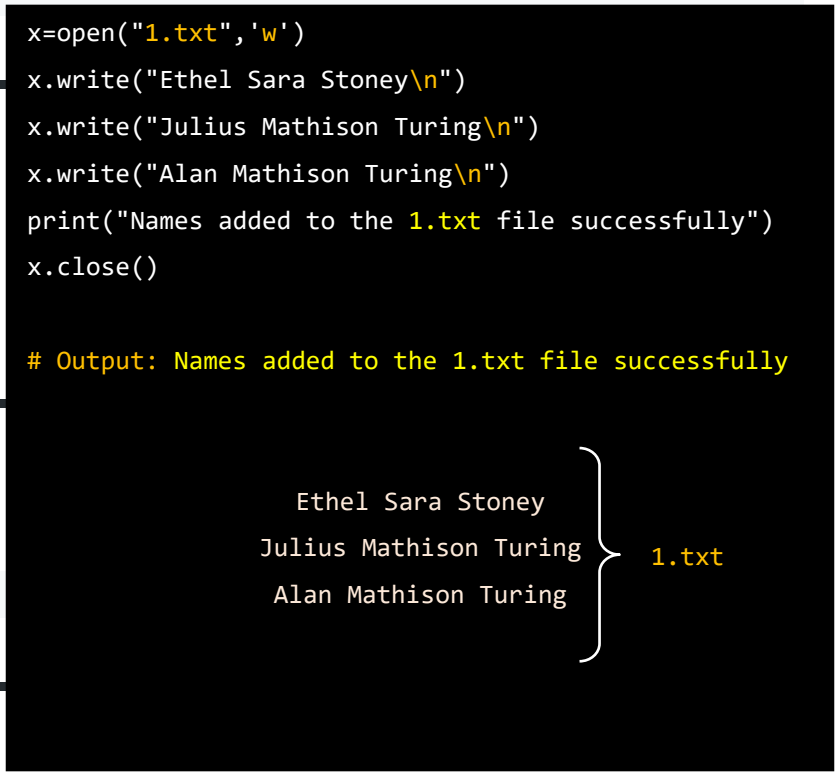
Command:

```
ls -d */
```

Description:

List multiple files on a single line.

```
x=open("1.txt",'w')
x.write("Ethel Sara Stoney\n")
x.write("Julius Mathison Turing\n")
x.write("Alan Mathison Turing\n")
print("Names added to the 1.txt file successfully")
x.close()

# Output: Names added to the 1.txt file successfully
```



Command:

```
ls --format=comma
```

Description:

View the process of a specific user "manju".

> **Attack Vector**
>
> A technique that a Cybercriminal employs to obtain unauthorized access to a computer or network in order to take advantage of security vulnerabilities

Command:

```
ps -u manju
```

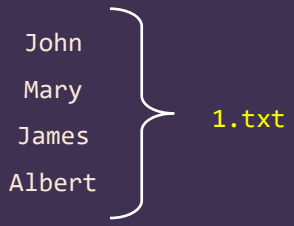
Description:

Execute a previous command starting with a specific letter "c".

Command:

```
!c
```

```
x=open("1.txt",'w')
a=["John\n","Mary\n","James\n","Albert"]
x.writelines(a)
print("List of names added to the 1.txt file successfully")
x.close()

# Output: List of names added to the 1.txt file successfully
```



Description:

Display BIOS information (You need elevated permissions to run this).

Command:

```
dmidecode -t 0
```

Description:

Display CPU information (You need elevated permissions to run this).

Command:

```
dmidecode -t 4
```

Description:

View all the system logs.

Command:

```
gnome-system-log
```

```
x=open("1.txt",'r')
i=x.read()
print(i)
x.close()

# Output:
                    John
                    Mary
                   James
                  Albert


x=open("1.txt",'r')
i=x.read(10)
print(i)
x.close()

# Output:
                    John
                    Mary
```

Description:

Identify SSH Client Version.

Command:

```
ssh -V
```

Description:

Display total connect time of users.

Command:

```
ac -d
```

Description:

Display connect time for all the users.

Command:

```
x=open("1.txt",'r')
print(x.readline(),end='')
print(x.readline(),end='')
print(x.readline(),end='')
print(x.readline(),end='')
x.close()

# Output:
                John
                Mary
                James
               Albert


x=open("1.txt",'r')
print(x.readline(),end='')
print(x.readline(),end='')
x.close()

# Output:
                John
                Mary
```

```
# Display all contents of current working directory
# including directories and files

import os
for dirpath,dirnames,filenames in os.walk('.'):
 print(dirpath) # prints current directory path
 print(dirnames) # prints directories
 print(filenames) # prints files
 print()
```

```
ac -p
```

Description:

Display connect time report for a specific user "manju".

Command:

```
ac -d manju
```

```
import os
# prints all statistics of 1.txt file
print(os.stat("1.txt"))
```

Description:

Display the modules compiled inside Apache.

Command:

```
httpd -l
```

```
import os
import datetime
# prints the File Size in Bytes
print(os.stat("1.txt").st_size)
# prints the File Last Accessed Time
print(datetime.datetime.fromtimestamp(os.stat("1.txt").st_atime))
# prints the File Last Modified Time
print(datetime.datetime.fromtimestamp(os.stat("1.txt").st_mtime))
```

Description:

View Processes Owned by Current User.

Output:

```
25
2022-06-28 02:20:25.945846
2022-07-08 04:53:18.734652
```

Command:

```
ps U $USER
```

Description:

Display the information about the file system Type.

Command:

```
df -Tha
```

Description:

Display Active Connections with Process ID and Program Name.

Command:

```
netstat -tap
```

Description:

Display RAW network statistics.

Operator	Used for

+	• **Arithmetic addition**
	`print(22+55) # Output: 77`
	• **String concatenation**
	`print('Alb'+'ert') # Output: Albert`

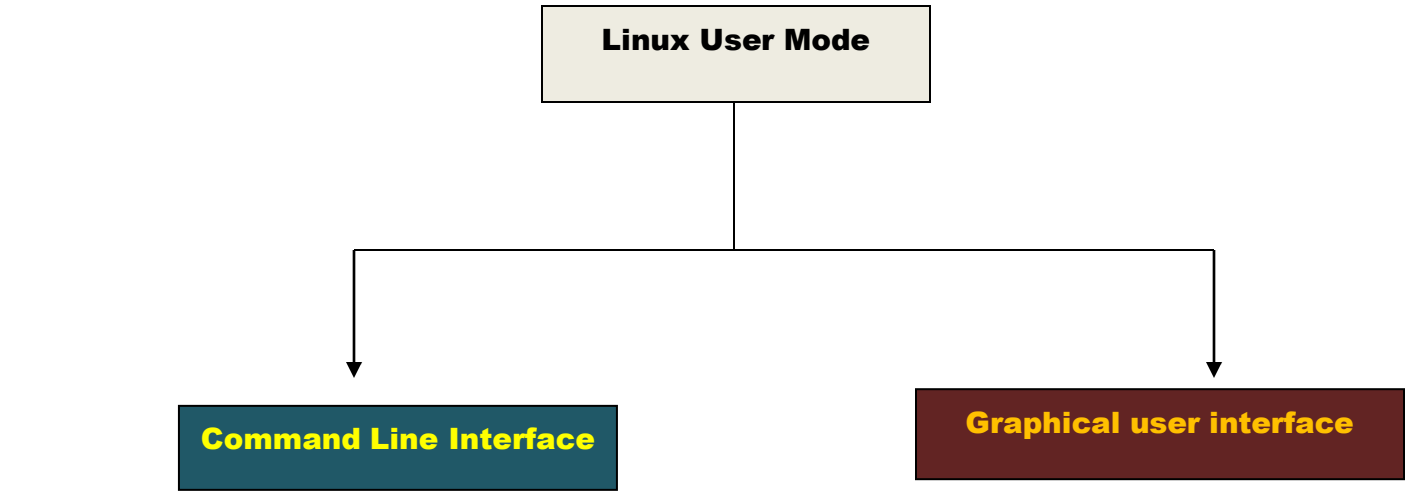
*	• **Multiplication**
	`print(2*5) # Output: 10`
	• **String repetition**
	`print('alan'*2) # Output: alanalan`

```
import gc
print(gc.isenabled()) # Is Garbage Collector enabled
# Output: True
gc.disable() # Garbage Collector is disabled
print(gc.isenabled()) # Is Garbage Collector enabled
# Output: False
gc.enable() # Garbage Collector is enabled
print(gc.isenabled()) # Is Garbage Collector enabled
# Output: True
```

Command:

```
netstat --statistics --raw
```

```
import re
print(re.sub("[a-z]","#","x9y8z7w6c"))
# Output: #9#8#7#6#
print(re.subn("[a-z]","#","x9y8z7w6c"))
# Output: ('#9#8#7#6#', 5)
# 5: The number of replacements
```



LILO (Linux Loader)

Load Linux into memory and start the OS

The maximum length for a filename under Linux is 255 bytes.

Logical Volume Manager (LVM)

Manage hard drives and other storage devices on linux

C Exercises

Question 1

Maturity Quality Attribute:

How well a software system satisfies dependability requirements when used normally

Question:

Write a program to print Hello World!.

Solution:

```c
#include <stdio.h>
int main() {
printf("Hello, World!");
return 0;
}
```

Modeling Tool

A tool for building models

Question 2

Accessibility Quality Attribute:

How well and how efficiently a software application can be used to achieve a specific purpose in a specific context of use by people with the broadest range of features and skills

Question:

Write a program to Find Sum of two numbers.

Solution:

```c
#include <stdio.h>
int main() {
int x;
int y;
int sum;

printf("Enter a number : \n");
scanf("%d", &x);
printf("Enter a number : \n");
scanf("%d", &y);
sum = x + y;
printf("Sum of %d + %d = %d", x, y, sum);
return 0;
}
```

Acyclic Dependencies Principle:

A key strategy for developing the structure of software systems in such a way that the dependencies between the components of the software system do not form a cycle

---

Question 3

Assessment: The process of collecting data about a software system's performance, threats, or weaknesses

Question:

Write a program to Find Difference of two numbers.

Solution:

```c
#include <stdio.h>
int main() {
int x;
int y;
int difference;
```

Adaptability Quality Attribute:

How well and how efficiently a software system may be modified to work in various operational or usage situations with various hardware, software, or other factors

```
printf("Enter a number : \n");
scanf("%d", &x);
printf("Enter a number : \n");
scanf("%d", &y);
difference = x - y;
printf("Difference of %d - %d = %d", x, y, difference);
return 0;
}
```

A pattern that appears on a regular basis is referred to as **seasonality** in time-series data

---

Question 4

Architecture evaluation:

Evaluation of a software architecture using quantitative or qualitative methods

Question:

Write a program to Find Division of two numbers.

Solution:

Architectural Decision

A decision that significantly or sustainably affects the software's architecture

```
#include <stdio.h>
int main() {
int x;
int y;
int division;

printf("Enter a number : \n");
scanf("%d", &x);
printf("Enter a number : \n");
scanf("%d", &y);
division = x / y;
```

Magnet loss:

A kind of loss function utilized in machine learning issues with a distance metric learning focus

```
printf("Division of %d / %d = %d", x, y, division);
return 0;
}
```

Question 5

Availability Quality Attribute:

How well and how efficiently a software system is usable and accessible when needed

Question:

Write a program to Calculate Area and Circumference of Circle.

Solution:

Appropriateness Recognizability Quality Attribute:

How well consumers can determine whether a software application is appropriate and suitable for their needs

Architecture rational:

Documentation of explanation for architectural decisions taken

```
#include <stdio.h>
int main() {
int r;
static float PI = 3.141592;
float area;
float Circumference;

printf("Enter the radius of Circle : \n");
scanf("%d", &r);
area = PI*r*r;
Circumference = 2*PI*r;
printf("Area of circle with radius %d = %f  \n", r, area);
printf("Circumference of circle with radius %d = %f  \n", r, Circumference);
return 0;
```

```
}
```

Capacity Quality Attribute:

How well a software product parameter's maximum limits conform to specifications

Question 6

Question:

Write a program to Calculate Area of Rectangle.

Solution:

Learnability Quality Attribute:

How well a software application may be utilized by specific users to meet specific goals like becoming effective, efficient, risk-free, and satisfied users of the application in a certain usage environment

```c
#include <stdio.h>
int main() {
int l;
int b;
long area;

printf("Enter the Length of a Rectangle : \n");
scanf("%d", &l);
printf("Enter the Breadth of a Rectangle : \n");
scanf("%d", &b);
area = l*b;
printf("Area of Rectangle with Length as %d and Breadth as %d = %ld", l, b, area);
return 0;
}
```

The Common Reuse Principle

We should not be dependent on resources we don't actually need

Question 7

The Reusability Principle:

It should be simple to reuse software components

Question:

Write a program to Calculate Area of Parallelogram.

Solution:

```
#include <stdio.h>
int main() {
int h;
int b;
long area;

printf("Enter the height of a Parallelogram : \n");
scanf("%d", &h);
printf("Enter the Base of a Parallelogram : \n");
scanf("%d", &b);
area = h*b;
printf("Area of Parallelogram with Height as %d and Base as %d = %ld ", h, b, area);
return 0;
}
```

Coupling:

A measure of how closely two software components are connected

---

- **Accidental complexity:** Difficulties that coders accidentally create for themselves when attempting to tackle a problem
- **Essential complexity:** How challenging a task is − regardless of our level of expertise, the tools we utilize, or the innovative and eye-catching architecture pattern we used to address the issue

Question 8

Mahout: An Apache open-source framework intended for the development of machine learning models

Question:

Write a program to Calculate Area of Trapezoid.

Solution:

Confidentiality Quality Attribute:

How well and how efficiently a software system makes sure that information is only available to those who have access

```
#include <stdio.h>
int main() {
int h;
int b1,b2;
long area;


printf("Enter the Height of a Trapezoid : \n");
scanf("%d", &h);
printf("Enter the Base 1 value : \n");
scanf("%d", &b1);
printf("Enter the Base 2 value : \n");
scanf("%d", &b2);
area = (h/2)*(b1 + b2);
printf("Area of Trapezoid with Height as %d, Base 1 as %d and Base 2 as %d = %ld", h, b1, b2, area);
return 0;
}
```

Kerckhoffs' Principle:

In order to accomplish the security of the encrypted messages, a cryptographic method must not need to be kept hidden

---

Constraint:

A limit on how flexible we may be when developing, planning, executing, or otherwise delivering a solution to a problem

Question 9

Installability Quality Attribute:

The extent of efficacy and efficiency required to install or remove a software application in a given environment

Question:

Write a program to Calculate Volume of Cube.

Solution:

Co-Existence Quality Attribute:

How well and how efficiently a software application may effectively carry out its necessary functions while coexisting with other applications in the same environment and using the same resources without negatively affecting any other application

```
#include <stdio.h>
int main() {
int s;
long volume;

printf("Enter the side of cube : \n");
scanf("%d", &s);
volume = s*s*s;
printf("Volume of Cube with Side as %d = %ld ",s, volume);
return 0;
}
```

Question 10

Fault Tolerance Quality Attribute:

How well and how efficiently a software system performs as expected despite possessing hardware or software issues

Question:

Write a program to Calculate Volume of Cylinder.

Solution:

Interoperability Quality Attribute:

The extent to which two or more software systems can share information and then use that information

```c
#include <stdio.h>
int main() {
const float PI = 3.141592;
float r, h, volume;

printf("Enter Radius and Height of the Cylinder: \n");
scanf("%f%f", &r, &h);
volume = PI * r * r * h;
printf("Volume of Cylinder is: %f", volume);
return 0;
}
```

Question 11

Functional Appropriateness Quality Attribute:

How well and how efficiently the functions facilitate the completion of particular tasks and goals

Question:

Write a program to Calculate Area of an Ellipse.

Solution:

Functional Correctness Quality Attribute:

How well and how efficiently a software system delivers the required level of precision and the accurate results

```c
#include <stdio.h>
#define PI 3.141592
int main() {

float major, minor, area;
printf("Enter length of major axis: \n");
scanf("%f", &major);
```

```
printf("Enter length of minor axis: \n");
scanf("%f", &minor);
area = PI * major * minor;
printf("Area of an ellipse = %0.4f", area);
return 0;
}
```

Integrity Quality Attribute:

How well a software system prevents against unwanted access and data alteration

---

Question 12

Hybrid architecture style:

Combination of 2 or more already-existing architectural designs or patterns

Question:

Write a program to Calculate Volume of Pyramid.

Solution:

```
#include <stdio.h>
#define PI 3.141592
int main() {

int b;
int h;
float volume;
printf("Enter the Base Area of Pyramid : \n");
scanf("%d", &b);
printf("Enter the Height of Pyramid : \n");
scanf("%d", &h);
volume = (1/3)*b*h;
```

Modifiability Quality Attribute:

How well a software application may be updated quickly and economically without introducing flaws or lowering the quality of the application

MapReduce:

A Google-developed application that enables huge data processing on a parallel, distributed cluster

```
printf("Volume of Pyramid with Base Area as %d and Height as %d = %f", b, h, volume);
return 0;
}
```

Question 13

Modularity Quality Attribute:

The extent to which a software system is made up of distinct components – so that changing one of them barely affects the other components

Question:

Write a program to Calculate Volume of Cone.

Solution:

Non-repudiation Quality Attribute:

How well events or acts may be definitively demonstrated to have occurred so that they cannot thereafter be reversed

```
#include <stdio.h>
int main() {

int h;
int r;
float volume;
float PI = 3.141592;
printf("Enter the Radius of Cone : \n");
scanf("%d", &r);
printf("Enter the Height of Cone : \n");
scanf("%d", &h);
volume = 1/3 * PI*r*r*h;
printf("Volume of Cone with Radius as %d and Height as %d = %f", r, h, volume);
return 0;
}
```

Markov chain:

A stochastic model that depicts a series of conceivable events where each event's likelihood solely depends on the condition obtained in the preceding event

Question 14

Operability Quality Attribute:

The extent to which a software application contains features that make it simple to use and manage

Question:

Write a program to Calculate Volume of Sphere.

Solution:

```c
#include <stdio.h>
int main() {

int r;
float volume;
float PI = 3.141592;
printf("Enter the Radius of Sphere : \n");
scanf("%d", &r);
volume = (4/3)*(PI*r*r*r);
printf("Volume of Sphere with Radius as %d = %f", r, volume);
return 0;
}
```

Open-Close-Principle:

It should not be necessary to modify current code in order to add functionality to the software application

Question 15

Portability Quality Attribute:

The ease and efficacy with which a software system can be adapted to a different hardware, software, or operating environment

Question:

Write a program to Calculate Volume of Ellipsoid.

Solution:

Perfect Forward Secrecy:

A cryptographic protocol's ability to prevent an attacker from obtaining anything about short-term session keys through the breach of long-term keys

```
#include <stdio.h>
int main() {

int r1,r2,r3;
float volume;
float PI = 3.141592;
printf("Enter the Radius of the ellipsoid of axis 1 : \n");
scanf("%d", &r1);
printf("Enter the Radius of the ellipsoid of axis 2 : \n");
scanf("%d", &r2);
printf("Enter the Radius of the ellipsoid of axis 3 : \n");
scanf("%d", &r3);
volume = (4/3)*(PI*r1*r2*r3);
printf("Volume of Ellipsoid with (r1,r2,r3) as (%d,%d,%d) = %f",r1,r2,r3,volume);
return 0;
}
```

Question 16

Performance Efficiency Quality Attribute:

Performance as a function of the quantity of resources consumed under the specified circumstances

Question:

Write a program to Calculate Surface area of Cube.

Solution:

Markov decision process:

A stochastic model in which an agent makes decisions and those decisions have random effects

```
#include <stdio.h>
```

```
int main() {

int side;
long surfacearea;
printf("Enter the Side of Cube : \n");
scanf("%d", &side);
surfacearea = 6*side*side;
printf("Surface area of Cube with side as %d = %ld", side, surfacearea);
return 0;
}
```

Qualitative evaluation:

Examining or determining whether a software's architecture can achieve the desired or necessary key objectives

---

Question 17

Quality attribute:

The extent to which a software application has the desired set of characteristics

Question:

Write a program to Calculate Surface area of Sphere.

Solution:

Recoverability Quality Attribute:

The extent to which a software system can restore its desired state and recover any immediately damaged data in the case of an interruption or failure

```
#include <stdio.h>
int main() {

int r;
float PI = 3.141592;
float surfacearea;
printf("Enter the Radius of Sphere : \n");
scanf("%d", &r);
surfacearea = 4*PI*r*r;
```

Markov property:

Future Markov process evolution is solely dependent on the current situation and is not influenced by the past

```
printf("Surface area of Sphere with radius as %d = %lf", r, surfacearea);
return 0;
}
```

Question 18

Redesign:

The modification of software components so that they carry out the same function as previously but in a new method and maybe by a different means

Question:

Write a program to Generate Multiplication Table.

Solution:

Replaceability Quality Attribute:

The extent to which a software application can perform the function of another specific software application in the same environment and for the same purpose

```
#include <stdio.h>
int main() {

int number;
int i = 1;
printf("Enter number to generate Multiplication Table : \n");
scanf("%d", &number);
printf("Multiplication Table of %d : \n", number);
for(i; i <11; i++) {
printf("%d x %d = %d \n", number, i, i*number);
}
return 0;
}
```

Matplotlib:

A Python open-source framework that makes it simple to produce animated and interactive data visualization

---

Question 19

Mean Reciprocal Rank:

A statistic used to assess any process that generates a list of potential answers to a sample of questions, sorted by likelihood that the answers are correct

Question:

Write a program to Compute Simple Interest.

Solution:

```c
#include <stdio.h>
int main() {

int principal;
int rate;
int duration;
float SI;
printf("Enter Principal Amount : \n");
scanf("%d", &principal);
printf("Enter Rate of Interest : \n");
scanf("%d", &rate);
printf("Enter Duration : \n");
scanf("%d", &duration);
SI = (principal*duration*rate)/100;
printf("Simple Interest :  %lf", SI);
return 0;
}
```

Reliability Quality Attribute:

The extent to which a software application carries out specific tasks under specific circumstances for a specific amount of time

Resource Utilization Quality Attribute:

The extent to which a software application complies with requirements in terms of the types and amounts of resources it uses to carry out its functions

---

Reuse/Release Equivalence Principle:

(Anything that we reuse needs to be released and tracked as well)

The anything of reuse cannot be less than the anything of release

Question 20

Mini-batch:

A tiny collection of data that is randomly chosen from all possible examples and delivered to the machine learning model in a single iteration for training

Question:

Write a program to Convert Degree to Fahrenheit.

Solution:

Round-trip engineering:

The idea of being able to update a model in any way and also the code produced by that model

```
#include <stdio.h>
int main() {

float degree, fahrenheit;
printf("Enter Temperature in Degrees : \n");
scanf("%f", &degree);
fahrenheit = (1.8*degree)+32;
printf("%f Degree in Fahrenheit :  %f", degree, fahrenheit);
return 0;
}
```

Question 21

Stable Abstractions Principle:

The more stable a software component is, the more abstract it is

Question:

Write a program to Convert Fahrenheit to Degree.

Solution:

User Error Protection Quality Attribute:

How well a software system prevents users against committing errors

```c
#include <stdio.h>
int main() {

float degree;
float fahrenheit;
printf("Enter Temperature in Fahrenheit : \n");
scanf("%f", &fahrenheit);
degree = ((fahrenheit-32)*5)/9;
printf("%f Fahrenheit in Degree Celsius = %f", fahrenheit, degree);
return 0;
}
```

Question 22

3 Goals of Information Security:

- **Reliability**
- **Confidentiality**
- **Accessibility**

Question:

Write a program to Convert Meters to Feet.

Solution:

Sensitivity point:

One component is a sensitivity point if it controls both runtime performance and reliability

```c
#include <stdio.h>
int main() {

float meter;
float feet;
printf("Enter Length in Meters  : \n");
scanf("%f", &meter);
```

```
feet = (meter*3.2808);
printf("%f meter in feet = %f", meter, feet);
return 0;
}
```

Question 23

Security Quality Attribute: How well a software system safeguards information and data so that users or other software systems can access it to the extent necessary for their various levels of authorization

Question:

Write a program to Convert Feet to Meters.

Solution:

System of Interest:

The software system whose architecture is being taken into account when creating an architectural description

```
#include <stdio.h>
int main() {

float meter;
float feet;
printf("Enter Length in Feet  : \n");
scanf("%f", &feet);
meter = (feet/3.2808);
printf("%f Feet in Meter = %f", feet, meter);
return 0;
}
```

Testability Quality Attribute:

How effectively and efficiently test requirements for a software application can be defined, and tests can be conducted to see if those requirements are fulfilled

---

Time Behavior Quality Attribute: How well a software system satisfies requirements in terms of response, processing and delivery performance while carrying out its functions

Question 24

Minimax: A decision rule for reducing the potential loss in a worst-case (**maximum loss**) situation that is utilized in artificial intelligence, decision theory, game theory, statistics and philosophy

Question:

Write a program to Convert Feet to Centimeters.

Solution:

User Interface Aesthetics Quality Attribute:

How well a user interface enables the user to interact in a pleasant and rewarding way

```c
#include <stdio.h>
int main() {

float cm;
float feet;
printf("Enter Length in Feet  : \n");
scanf("%f", &feet);
cm = 30.48 * feet;
printf("%f Feet in Meter = %f", feet, cm);
return 0;
}
```

ML-as-a-service:

Various Machine Learning tools and technologies that are offered in the cloud as a service as part of cloud computing services

---

Question 25

- **Forward traceability:** All requirements are met by the software's components
- **Backward traceability:** At least one requirement justifies each component of the software

Question:

Write a program to Convert Feet to millimeters.

Solution:

```c
#include <stdio.h>
int main() {

float mm;
float feet;
printf("Enter Length in Feet  : \n");
scanf("%f", &feet);
mm = (feet * 304.8);
printf("%f Feet in Millimeter = %f", feet, mm);
return 0;
}
```

Broker:

A kind of software that coordinates communication, including forwarding requests and transmitting outcomes and exceptions

- **Lubarsky's Law of Cybernetic Entomology:** One additional bug is always present
- **Gallois' Revelation:** Foolishness is the only thing that emerges from a computer when we put foolishness in it

Question 26

Question:

Write a program to Convert Kilometer to Miles.

Solution:

The Architecture Tradeoff Analysis Method

(ATAM)

A technique for assessing software architectural vulnerabilities that can prevent a company from achieving its objectives

```c
#include <stdio.h>
int main() {

float km;
float miles;
printf("Enter Length in Kilometer  : \n");
scanf("%f", &km);
```

```
miles = (0.621371*km);
printf("%f Kilometer in Millimeter = %f", km, miles);
return 0;
}
```

AWS Management Console:

A simple and user-friendly web-based interface for managing Amazon Web Services

Question 27

Question:

Write a program to Convert Yard to Foot.

Solution:

AWS Auto Scaling:

AWS service that analyses the software application and optimizes capacity to ensure consistent, anticipated performance while being cost-effective

```
#include <stdio.h>
int main() {

float yard;
float foot;
printf("Enter Length in Yard  : \n");
scanf("%f", &yard);
foot = (3*yard);
printf("%f Yard in Foot = %f", yard, foot);
return 0;
}
```

Model capacity:

The degree of complexity of problems that a ML model can learn. The complexity of the problem the ML model can learn is inversely correlated with model capacity

Elastic Beanstalk: AWS solution that enables users to manage and deploy web applications in the **AWS Cloud** without having to worry about the infrastructure

Question 28

Lambda:

AWS service that enables users to run code without having to manage servers or provide security

Question:

Write a program to Convert Inch to Cm.

Solution:

AWS Route 53

AWS Cloud Domain Name System web service that helps business organizations turn domain names into IP addresses

```c
#include <stdio.h>
int main() {

float inch;
float cm;
printf("Enter Length in inch  : \n");
scanf("%f", &inch);
cm = (2.54*inch);
printf("%f inch in cm = %f", inch, cm);
return 0;
}
```

In an algorithm for a neural network, a **neuron** represents a node. One output is produced by a **neuron** from many inputs.

Question 29

AWS Direct Connect:

A cloud solution that provides constant, low-latency performance by connecting your network directly to AWS cloud

Question:

Write a program to convert bytes to kilobytes.

Solution:

```c
#include <stdio.h>
int main() {

int bytes;
double kilobytes;
printf("Enter number of bytes: \n");
scanf("%d", &bytes);
kilobytes=bytes/1024.00;
printf("Kilobytes: %.2lf",kilobytes);
return 0;
}
```

AWS Storage Gateway:

AWS service that makes it possible for the user's existing application to access AWS cloud storage

---

Question 30

Amazon glacier: An affordable AWS storage solution that offers security measures for data backup and data preservation

Question:

Write a program to convert MB to KB.

Solution:

Snowball:

A data transport solution that moves a lots of data in and out of the AWS cloud

```c
#include <stdio.h>
int main() {

double megabytes, kilobytes;
printf("Please enter the amount of megabytes to convert: \n");
scanf("%lf", &megabytes);
kilobytes = megabytes * 1024;
printf("There are %lf kilobytes in %lf megabytes.", kilobytes, megabytes);
```

```
return 0;
}
```

Question 31

AWS Simple Notification Service:

AWS completely managed message delivery solution that updates the user on their AWS Account status

Question:

Write a program to convert GB to MB.

Solution:

Amazon Aurora:

A large-scale relational database solution created for the cloud that is fully compatible with **MySQL** and **PostgreSQL** and combines the flexibility, availability and efficiency of high-end economical databases

```c
#include <stdio.h>
int main() {

double gb, mb;
printf("Enter Length in Gigabyte  : \n");
scanf("%lf", &gb);
mb = gb*1024;
printf("%lf GB  =  %lf MB", gb, mb);
return 0;
}
```

Moving average:

A stock indicator frequently employed in technical analysis that establishes a continuously updated average price to assist smooth out price data

Amazon ElastiCache

↓

AWS managed caching service that sets up, runs, and scales well-known in-memory data stores that are compatible with open source

Question 32

Data Pipeline:

AWS web service that facilitates processing and movement of data between several AWS computing and storage services

Question:

Write a program to convert KB to MB.

Solution:

Amazon Kinesis:

AWS service that gathers, processes, and evaluates live streaming data so we can quickly respond to new data and gain reliably accurate insights

```
#include<stdio.h>
int main() {

double kb, mb;
printf("Enter Length in Kilobyte  : \n");
scanf("%lf", &kb);
mb = (kb/1024);
printf("%lf KB  =  %lf MB", kb, mb);
return 0;
}
```

Question 33

Amazon Simple Queue Service:

AWS managed messaging service that facilitates the scaling and decoupling of serverless applications, distributed systems and microservices

Question:

Write a program to Convert Kilogram to Pounds.

Solution:

```c
#include <stdio.h>
int main() {

float kg, lbs;
printf("Enter Weight in Kilogram  : \n");
scanf("%f", &kg);
lbs = kg*2.20462;
printf("%f Kg  =  %f Pounds", kg, lbs);
return 0;
}
```

AWS CloudSearch:

AWS managed service that sets up, manages and scales a search solution for our web application in a simple and affordable way

---

Question 34

AWS Simple Email Service:

A scalable, adaptable and inexpensive AWS email delivery service

Question:

Write a program to Convert Kilogram to Ounce.

Solution:

AWS Identity and Access Management:

A framework that controls secure access to AWS resources and services

```c
#include <stdio.h>
int main() {

float kg, ounce;
printf("Enter Weight in Kilogram  : \n");
scanf("%f", &kg);
ounce = kg*35.274;
printf("%f Kg  =  %f Ounce \n", kg, ounce);
return 0;
}
```

AWS Simple Workflow Service:

An AWS solution that assists developers in managing, monitoring, and auditing multi-step, multi-machine application jobs

Question 35

Amazon Cloudwatch:

AWS monitoring service that provides us with insight into the functionality and condition of our AWS applications and resources

Question:

Write a program to Convert Pounds to Grams.

Solution:

AWS Key Management Service:

AWS solution that aids in the creation and management of the encryption keys used to encrypt data

```
#include <stdio.h>
int main() {

float pound, gram;
printf("Enter Weight in Pounds  : \n");
scanf("%f", &pound);
gram = pound*453.592;
printf("%f Pound  =  %f Grams", pound, gram );
return 0;
}
```

Question 36

AWS CloudTrail:

AWS solution that records usage of the API and AWS accounts for risk assessment, accountability and surveillance

Question:

Write a program to Calculate total marks and Percentage.

Solution:

AWS CloudFormation:

AWS solution that aids in modeling and setting up our AWS resources so we may spend more time concentrating on our AWS-running apps and less time managing those resources

```c
#include <stdio.h>
int main() {

int algebra, geometry, arithmetic;
int sum;
float percentage;
printf("Enter Marks Scored in Algebra  : \n");
scanf("%d", &algebra);
printf("Enter Marks Scored in Geometry  : \n");
scanf("%d", &geometry);
printf("Enter Marks Scored in Arithmetic  : \n");
scanf("%d", &arithmetic);
sum = algebra + geometry + arithmetic;
percentage = (sum*100)/300;
printf("Total marks scored  = %d /300 \n", sum );
printf("Percentage Scored  = %f", percentage );
return 0;
}
```

Application virtualization:

The process of an operating system and an application communicating to each other even when they are not directly connected

---

Question 37

- **AWS Developer Tools:** A set of services intended to aid users in creating, testing and deploying their own code
- **AWS Management Tools:** A set of services that enable users to manage, monitor, automate and deploy all cloud environment components programmatically

Question:

Write a program to Calculate Discount Amount and Discounted Price.

Solution:

Multi-class classification problem:

A problem where there are more than two classes for the target variable

```c
#include <stdio.h>
```

```
int main() {

float price;
float discountpercentage;
float discountamount;
float discountedprice;
printf("Enter Price of Item  : \n");
scanf("%f", &price);
printf("Enter Discount Percentage on Item  : \n");
scanf("%f", &discountpercentage);
discountamount = (discountpercentage*price)/100;
discountedprice = (price-discountamount);
printf("Discount amount : %f \n",  discountamount);
printf("Discounted price : %f",  discountedprice);
return 0;
}
```

Block storage:

A technique of storing data where each storage volume functions as a separate hard disc that is set up by the storage admin

MXNet:

An open-source platform that facilitates the deployment and training of deep neural networks

---

Question 38

Composable infrastructure: Resources for storing, computing and networking can be handled by software using a web-based interface since they are separated from their actual locations

Question:

Write a program to Swap two numbers using a Temporary variable.

Solution:

Cloudless Computing:

Releasing developers from restrictions that require them to decide whether to deploy in a private or public cloud for any given workload and then stick with that decision for the lifetime of the application

```
#include <stdio.h>
int main() {


double first, second, temp;
```

```
printf("Enter first number :   \n");
scanf("%lf", &first);
printf("Enter second number : \n");
scanf("%lf", &second);
printf("Before Swapping => first number : %lf   second number : %lf \n",  first, second);
temp = first;
first = second;
second = temp;
printf("After Swapping => first number : %lf  second number : %lf",  first, second);
return 0;
}
```

Question 39

Digital manufacturing: The application of technologies to the manufacturing process in order to increase industrial operations' scalability, effectiveness, and robustness to changing market demands

Question:

Write a program to Swap two numbers without using a Temporary variable.

Solution:

Hybrid Workplace: A scenario where employees have the choice of working remotely or in an office as their workplace

```
#include <stdio.h>
int main() {

double first, second;
printf("Enter first number : \n");
scanf("%lf", &first);
printf("Enter second number : \n");
scanf("%lf", &second);
printf("Before Swapping => first number : %lf  second number : %lf \n", first, second);
```

```
first = first + second;
second = first - second;
first = first - second;
printf("After Swapping => first number : %lf  second number : %lf",  first, second);
return 0;
}
```

Question 40

Exascale computing:

Computing systems capable of performing a billion billion calculations per second (10^{18} calculations per second)

Question:

Write a program to Reverse Digits of a Number.

Solution:

Hypervisor :

Software that makes it possible for several virtual machines (VMs) to share resources from a single server

```
#include <stdio.h>
int main(){

int number, reverse = 0;
printf("Enter any number: \n");
scanf("%d", &number);
while(number != 0){
   reverse = (reverse * 10) + number % 10;
   number = number/10;
}
printf("Reversed number : %d", reverse);
return 0;
}
```

Hyperscale:

The complete set of resources – both physical and technological – that may extend a distributed computing environment to thousands of servers

---

Question 41

Packet-Switched Network:

A network that packetizes data transfers between computing devices

Question:

Write a program to Calculate sum of Digits of Entered Number.

Solution:

Intelligent edge:

A location where data is created, examined, evaluated and dealt with

Infrastructure Security:

The process of defending key infrastructure against physical and digital threats

In-memory computing:

A technique of using RAM to store data instead of databases that are hosted on drives

```c
#include <stdio.h>
int main() {

int n, m, sum = 0;
printf("Enter a number: \n");
scanf("%d", &n);
m = n;
while (m != 0) {
sum = sum + m % 10;
m = m / 10;
}
printf("Sum of digits of %d = %d", n, sum);
return 0;
}
```

Question 42

Intelligent Storage:

A hybrid cloud environment data storage system that employs AI to constantly adapt and evolve to improve data management and service

Question:

Write a program to calculate the largest of 3 numbers.

Solution:

IT Asset Management:

The technique of controlling and managing IT assets while under ownership to maximize their value to the enterprise organization

```c
#include <stdio.h>
int main() {

int A, B, C;
printf("Enter the numbers A, B and C: \n");
scanf("%d %d %d", &A, &B, &C);
if (A >= B && A >= C)
printf("%d is the largest number.", A);
if (B >= A && B >= C)
printf("%d is the largest number.", B);
if (C >= A && C >= B)
printf("%d is the largest number.", C);
return 0;
}
```

IT procurement:

A set of procedures for acquiring and implementing information technology in alignment with business strategy

Question 43

Kubernetes:

An open-source Cluster Management Software for managing, scaling and automating the deployment of containerized applications

Question:

Write a program to Check Entered character is a Vowel or Consonant.

Solution:

Object storage:

An approach to data storage that effectively organizes data as objects with supplementary Meta information and unique IDs

```c
#include <stdio.h>
int main() {
```

```
char ch;
printf("Enter a character : \n");
scanf("%c", &ch);
if(ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||
   ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {
          printf("%c is a Vowel!", ch);
}
else {
         printf("%c is a Consonant!", ch);
}
return 0;
}
```

Hybrid WAN: A wide area network that distributes traffic among two or more different types of connections

Question 44

Operational technology:

Monitoring and controlling physical processes, devices and infrastructure using hardware and software

Question:

Write a program to Check Entered Number is a Leap Year or not.

Solution:

Multi-Cloud:

A cloud computing architecture wherein an enterprise organization divides its resources over at least 2 clouds from various vendors, either public or private

```
#include <stdio.h>
int main() {

int year;
printf("Enter a Number : \n");
scanf("%d", &year);
if ((year % 400 == 0) || ( ( year % 100 != 0) && (year % 4 == 0 ))) {
       printf("%d is a leap year!", year);
   }
```

```
   else {
       printf("%d is not a leap year!", year);
   }
return 0;
}
```

Question 45

Memory-Driven Computing:

An architecture that is practically indefinitely adaptable and expandable, capable of completing any computing work far more quickly and efficiently than traditional methods

Question:

Write a program to find factorial of a number.

Solution:

Reference Architecture:

A set of documentation that recommends how to integrate IT services and products to create a solution

Zero Trust:

An authentication and authorization strategy built on the principle that no user or piece of software can be trusted by default

```
#include <stdio.h>
int main() {

int num, i;
long int fact;
printf("Enter an integer number: \n");
scanf("%d", &num);
fact=1;
for(i=num; i>=1; i--)
    fact=fact*i;
printf("Factorial of %d is : %ld", num, fact);
return 0;
}
```

Question 46

Remote Infrastructure Management:

The method of administering an enterprise organization's full or a portion of its IT infrastructure from a distant location

Question:

Write a program to Display day of week using switch case.

Solution:

Tech Refresh:

The process of consistently upgrading important components of our IT infrastructure to improve performance of the system

```c
#include <stdio.h>
int main() {

int day;
printf("Enter Day (1 to 7) : \n");
scanf("%d", &day);

switch(day)  {
    case 1:
    printf("The 1st day of week is : Sunday");
    break;

    case 2:
    printf("The 2nd day of week is : Monday");
    break;

    case 3:
    printf("The 3rd day of week is : Tuesday");
    break;

    case 4:
    printf("The 4th day of week is : Wednesday");
    break;
```

Apache Spark:

An open-source data processing technology that makes it easier to create and do data analytics tasks

Supercomputing:

A kind of high-performance computing that effectively combines the processing capacity of numerous computing systems to handle exceedingly complicated data-intensive challenges

```
    case 5:
    printf("The 5th day of week is : Thursday");
    break;

    case 6:
    printf("The 6th day of week is : Friday");
    break;

    case 7:
    printf("The 7th day of week is : Saturday");
    break;

    default:
    printf("Invalid input!! .....");
}
return 0;
}
```

Storage area network: A standard storage networking architecture that provides business critical applications with maximum bandwidth and reduced latency

Security Monitoring:

The computerized process of identifying and analyzing potential security threat indicators, then classifying and responding to those threats

---

Question 47

Hyperconverged Infrastructure:

A software-based approach to fuse the data center's networking, storage and computation systems

Question:

Write a program to Check Entered number is positive or negative.

Solution:

Converged infrastructure:

A hardware-based approach to fuse the data center's networking, storage and computation systems

```
#include <stdio.h>
int main() {

int number;
```

```
printf("Enter a number : \n");
scanf("%d", &number);
if(number > 0) {
      printf("Entered number %d is a positive number", number);
}
    else if(number < 0) {
      printf("Entered number %d is a negative number", number);
}
    else {
      printf("Entered number %d is neither positive nor negative", number);
}
return 0;
}
```

Question 48

Immutable infrastructure:

Infrastructure that – once implemented – cannot be changed

Question:

Write a program to Check Entered Character is Upper Case or Lower Case.

Solution:

Network Functions Virtualization:

The use of virtual machines to replace the hardware of network devices

```
#include <stdio.h>
int main() {

char ch;
printf("Enter a alphabet : \n");
scanf("%c", &ch);

if(ch>=65 && ch<=90)
```

FlowVisor:

A network virtualization hypervisor that enables several clients to share the same physical infrastructure

```
     printf("upper case");

    else if(ch>=97 && ch<=122)
     printf("lower case");

    else
     printf("Invalid input!!");

    return 0;
}
```

Anti-Virus Gateway:

A built-in antivirus protection feature of network security device that examines files included in network traffic such as emails, web downloads and attachments and tries to prevent potential threats before they reach the network

---

Question 49

Operation Support System:

A piece of software that enterprise organizations typically use to govern their operating systems

Question:

Write a program to Print Number of Days in a Month.

Solution:

Application firewall:

A kind of firewall that checks, keeps an eye on and manages access to and from applications and services on networks, the Internet and local systems

```
#include <stdio.h>
int main() {

int a[12]={31,28,31,30,31,30,31,31,30,31,30,31},m;
printf("Enter the month: \n");
scanf("%d", &m);

if(m>12 || m<1) {
printf("Invalid input");
}
```

Data Lineage:

A lifespan of data that takes into account the origin of data, what happens to the data and where data goes over time

```
else if(m==2) {
printf("Number of days in month 2 is either 29 or 28");
}
else {
printf("Number of days in month %d is %d", m, a[m-1]);
}
return 0;
}
```

Scaling:

A feature engineering approach that aids in ensuring that features have the same value range. **For example**, we can scale a feature by dividing it by 100 if we want to change its range from (0 to 100) to (0 to 1)

Question 50

Business Support System:

A piece of software that Enterprise organizations typically employ to manage all their business operations

Question:

Write a program to calculate the length of string.

Solution:

Cisco onePK:

A framework that allows developers to create an application that is simple to incorporate into a Cisco infrastructure

```
#include <stdio.h>
#include <string.h>
int main() {

char Str[1000];
int i;
printf("Enter the String: \n");
scanf("%s", Str);
for (i = 0; Str[i] != '\0'; ++i);
  printf("Length of '%s' is %d", Str, i);
return 0;
}
```

Converged Storage:

A scalable storage design that unifies computing and storage

Question 51

Enterprise Data Center Networking:

A business data center's networking that links users and servers together

Question:

Write a program to Concatenate Two Strings.

Solution:

Corporate Social Responsibility:

A means of expressing how business organizations evaluate and regulate their business action on financial, ecological and social responsiveness

```c
#include <stdio.h>
#include <string.h>
int main() {

char a[1000], b[1000];
printf("Enter the first string: \n");
scanf("%s", a);
printf("Enter the second string: \n");
scanf("%s", b);
strcat(a, b);
printf("String obtained on concatenation: %s", a);
return 0;
}
```

Endpoint Protection:

A centralized system with an integrated security solution that incorporates several security services

File Level Protection

A widely used technology that enables file extraction from network traffic and inspects it to look for malware, Trojan horses, bugs or computer viruses

Question 52

Data observability: Enterprise organization's ability to record, evaluate and recommend a course of action regarding the status and health of the data in their system

Question:

Write a program to Find Maximum Between Two Numbers.

Solution:

Hardware Security Module:

A piece of hardware that combines security mechanisms with software security technologies to protect sensitive information and safely store cryptographic keys

```c
#include <stdio.h>
int main() {

int a, b;
printf("Enter two numbers: \n");
scanf("%d%d",&a,&b);
if(a>b)
printf("%d is a maximum number", a);
else
printf("%d is a maximum number", b);
return 0;
}
```

Intent-Based Networking:

A technological idea that seeks to replace manual methods for setting up networks and responding to network problems with a greater level of intellect and desired configuration

---

Question 53

Intrusion Prevention System:

A network security device that continuously scans for fraudulent attacks on a network and takes action to report and block it

Question:

Write a program to check whether the entered number is a prime number or not.

Solution:

```
#include <stdio.h>
int main() {

int num, i, x=0;
printf("Enter a number: \n");
scanf("%d", &num);

for(i=2; i<num; i++) {
   if(num%i == 0) {
            x++;
            break;
}
}
if(x==0) {
        printf("%d is a prime number", num);
}
    else {
        printf("%d is not a prime number", num);
  }
   return 0;
}
```

Intrusion Detection System:

A passive computer security tool that keeps track of all incoming and outgoing network traffic

Active security:

The steps the business organizations take to stop the malicious attack before it happens

Passive security:

The steps the business organizations take to address malicious attacks once they have already happened

---

Question 54

- **Active attack:** A perpetrator tries to change the message content
- **Passive attack:** A perpetrator views the message content and duplicates it

Question:

Write a program to Get Input from User.

Packet Filtering:

A technique for managing network access

Solution:

```
#include <stdio.h>
int main() {

int num;
printf("Enter the Number: \n");
scanf("%d", &num);
printf("You've entered: %d", num);
return 0;
}
```

Load Balancer:

A solution that aims to boost application performance, efficiency and availability while preventing server overload

---

Question 55

Kernel-based Virtual Machine:

An open-source virtualization platform created for the Linux operating system and intended to run on x86-based CPU architecture

Question:

Write a program to Print ASCII Value of entered Alphabet.

Solution:

Overlay network:

A network that is supported by the infrastructure of another network and is constructed on top of it

```
#include<stdio.h>
#include<conio.h>
int main() {

char ch;
int i;
printf("Enter a Alphabet: \n");
scanf("%c", &ch);
i = ch;
printf("ASCII Value of %c = %d", ch, i);
```

Open Systems Interconnection:

A method for sending messages between any two networked devices

```
return 0;
}
```

Question 56

Quality of Service:

The use of network-based technology to maintain the functioning of crucial applications despite a limited available bandwidth

Question:

Write a program to Compare Two Strings.

Solution:

Stateful Inspection:

A firewall methodology used by security tools to keep track of the connections that are currently active on the network

```
#include <stdio.h>
#include <string.h>
int main() {

char a[100], b[100];
printf("Enter a string: \n");
scanf("%s", a);
printf("Enter a string: \n");
scanf("%s", b);

if (strcmp(a, b) == 0) {
      printf("The strings are equal.");
}
   else {
      printf("The strings are not equal.");
}
  return 0;
}
```

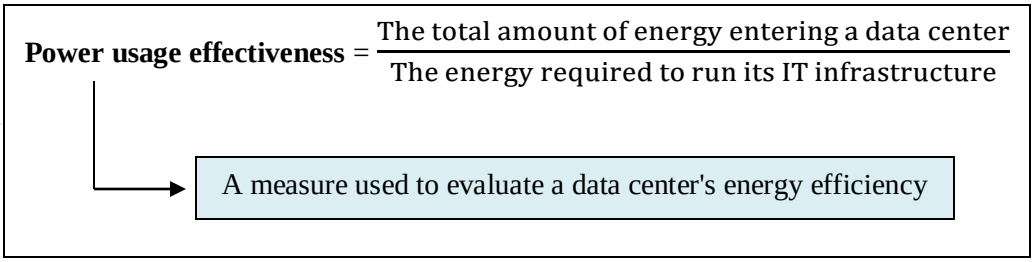
Security service edge:

A set of cloud-centric security features that are integrated and enable safe access to websites, SaaS applications and Cloud services

User and entity behavior analytics:

A security tool that aids security teams in recognizing and managing unusual user activity that may be indication of accounts that have been hacked

Question 57



Question:

Write a program to Convert Lowercase to Uppercase.

Python code:

```
x = 1/5.77

print(x) # Output: 0.1733102253032929

print(format(x, '0.3f')) # Output: 0.173

print(format(1/5.77, '.1%')) # Output: 17.3%
```

Solution:

```c
#include <ctype.h>
#include <stdio.h>
int main() {

char ch;
ch = 'g';
printf("%c in uppercase is represented as  %c", ch, toupper(ch));
return 0;
}
```

Question 58

```
x = None

print(x) # Output: None
```

Question:

Write a program to Convert Uppercase to Lowercase.

Solution:

```
from datetime import datetime

print("Time now: {:%I:%M %p}".format(datetime.now()))

# Output: Time now: 03:28 AM
```

```c
#include <ctype.h>
#include <stdio.h>
int main() {

char ch;
ch = 'G';
printf("%c in lowercase is represented as  %c", ch, tolower(ch));
return 0;
}
```

Question 59

$$\text{Data center infrastructure efficiency} = \frac{1}{\text{Power usage effectiveness}}$$

Question:

Write a program to find the Average of two numbers.

Solution:

```
print(slice(None, 2, 6).indices(8)) # Output: (0, 2, 6)

print(slice(None, None, 6).indices(8)) # Output: (0, 8, 6)

print(slice(None, None, None).indices(8)) # Output: (0, 8, 1)
```

```c
#include <stdio.h>
int main() {

int num1, num2;
float avg;
printf("Enter first number: \n");
scanf("%d", &num1);
printf("Enter second number: \n");
scanf("%d", &num2);
```

```
for i in range(0, 2):

   print('Albert %s' % i)

# Output:

              Albert 0

              Albert 1
```

```
avg= (float)(num1+num2)/2;
printf("Average of %d and %d is: %.2f", num1, num2, avg);
return 0;
}
```

Question 60

```
import itertools

print(list(itertools.takewhile(lambda x: x < 13, itertools.count(11, .5))))

# Output: [11, 11.5, 12.0, 12.5]
```

Question:

Write a program to find Quotient and Remainder.

Solution:

```
#include <stdio.h>
int main() {

int dividend, divisor, quotient, remainder;
printf("Enter dividend: \n");
scanf("%d", &dividend);
printf("Enter divisor: \n");
scanf("%d", &divisor);
quotient = dividend / divisor;
remainder = dividend % divisor;
printf("Quotient = %d\n", quotient);
printf("Remainder = %d", remainder);
return 0;
}
```

```
import time

print(time.asctime())

# Output: Tue Jul 5 06:53:12 2022
```

Question 61

All pairs Testing:

The testing technique for identifying flaws that combines two test scenarios

Question:

Write a program to find the square of a number.

Burndown chart:

A graphical representation of the amount of work remaining versus the amount of time

Solution:

```
#include<stdio.h>
#include<math.h>
int main() {

int a, b;
a=2;
b = pow((a), 2);
printf("The square of a = %d", b);
return 0;
}
```

Actual Outcome:

Actual output that a tester receives after running the test

Question 62

Age Testing:

A testing methodology that examines how software application might function in the future

Question:

Write a program to print the average of the elements in the array.

Solution:

```c
#include<stdio.h>
int main() {

int i, avg, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num [i];
avg = sum/5;
printf("Sum of the Elements in the array = %d \n", sum);
printf("Average of the elements in the array = %d", avg);
return 0;
}
```

API Testing:

A software testing approach that examines an API to ensure that it meets its requirements for usability, security, efficiency and dependability

---

Question 63

Decision Coverage = $\frac{\text{Number of decision outcomes that were carried out}}{\text{Total number of decision outcomes}} \times 100$

Question:

Write a program to find the greatest of two numbers using pointers.

Solution:

Assertion based testing

→ Employed to test a logical statement

```c
#include<stdio.h>
int main() {

int x, y, *p, *q;
printf("Enter any integer: \n");
scanf("%d", &x);
printf("Enter any integer: \n");
scanf("%d", &y);
p = &x;
```

Automated Software Testing:

Before software application is launched into production, a manual process of assessing and validating whether it is functioning properly and satisfying specifications is automated using software testing tools

```
q = &y;
if(*p>*q) {
printf("x is greater than y");
}
if(*q>*p) {
printf("y is greater than x");
}
return 0;
}
```

Backward Compatibility Testing:

The method for testing if a software application created for an older version of a platform still functions on a later version of that platform

---

Question 64

Bebugging:

The deliberate insertion of known bugs and flaws into the software application in order to track how quickly they are found and fixed

Question:

Write a program to print the address of x and the value assigned to x.

Solution:

Basis Path Testing: A white box testing methodology where test cases are designed in accordance with logical paths of execution that can be followed within the software program

```
#include <stdio.h>
int main() {

int x, *p;
x = 1;
p = &x;
printf("The address of the variable x =%d \n", p);
printf("The value of the variable x =%d", *p);
return 0;
}
```

Equally likely outcomes:

An experiment's results are all equally likely to occur

Question 65

Boundary Testing:

A black box testing approach where tests are executed utilizing the boundary values

Question:

Write a program to print the first 10 numbers starting from one together with their squares and cubes.

Solution:

Big Bang Integration Testing:

A methodology where all software modules or components are merged at once, after which the entire system is tested

```
#include<stdio.h>
int main() {

int i;
for(i=1; i<=10; i++)
printf("Number=%d its square=%d its cube=%d\n", i , i*i, i*i*i);
return 0;
}
```

Question 66

Portability Testing:

Transferring a software application from one environment to another while testing it without difficulty. For example: testing a software application that works well in Windows 8 and observing how it behaves in Windows 10

Question:

Write a program:

If you enter a character M

Output must be: ch = M.

Solution:

```
#include<stdio.h>
int main() {

char M;
printf("Enter any character: \n");
scanf("%c", &M);
printf("ch=%c", M);
return 0;
}
```

Breadth Testing:

A set of tests that verifies a software application's full functionality but does not carefully examine each feature

---

Question 67

Code Freeze:

Prohibiting software programmers from making additional changes to the software code

Question:

Write a program to print the product of the first 10 digits.

Solution:

Capture/Replay Tool:

A kind of test execution tool where the manual testing inputs are recorded with the aim of producing automated test scripts that may be executed later

```
#include<stdio.h>
int main() {

int i, product = 1;
for(i=1; i<=10; i++)
product = product * i;
printf("The product of the first 10 digits =%d\n", product);
return 0;
}
```

Question 68

Code Free Testing:

A testing methodology that doesn't involve running any code

Question:

Write a program to check the equivalence of two numbers.

Solution:

Code-Based Testing:

Checking each line of a software code for flaws, bugs or inaccuracies as part of the software development process

Comparison Testing:

The process of testing a software application to determine its advantages and disadvantages in comparison to other software applications already on the market

```
#include<stdio.h>
int main() {

int x, y;
printf("Enter any number: \n");
scanf ("%d", &x);
printf("Enter any number: \n");
scanf ("%d", &y);
if(x-y==0) {
printf("The two numbers are equivalent");
}
else {
printf("The two numbers are not equivalent");
}
return 0;
}
```

Compliance Testing: A software testing methodology to check if a software application satisfies a specified set of enterprise organization's predefined requirements before it is made available for use in production

Question 69

Context Driven Testing:

A methodology for creating and testing software applications that examines how they will be used or are predicted to be used in the real world

Question:

Write a program to print the characters from A to Z.

Solution:

Configuration Testing:

A software testing methodology that compares the performance of the software application being developed with various software and hardware configurations to determine the ideal configuration for the application to operate flawlessly and meet its functional criteria

```
#include<stdio.h>
int main() {

char a;
for( a='A'; a<='Z'; a++)
printf("%c\n", a);
return 0;
}
```

Question 70

Conformance Testing:

A software testing methodology used to confirm that a software application complies with IEEE, **W3C** or ETSI standards

Question:

Write a program to find the incremented and decremented values of two numbers.

Solution:

Delta Release:

A hardware or software module's partial release

```
#include<stdio.h>
```

```c
int main() {

int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
printf("The incremented value of a =%d \n", c);
printf("The incremented value of b =%d \n", d);
printf("The decremented value of a =%d \n", e);
printf("The decremented value of b =%d \n", f);
return 0;
}
```

Conversion Testing:

A testing methodology used to ensure that one data format can be transformed into another, with the converted data format being used correctly and effectively by the software application being tested

Learning rate:

A tuning parameter in an optimization method that determines the rate at which model parameters are updated based on the gradient

---

Question 71

Data Integrity Testing:

A testing methodology that confirms the database's data is reliable and performs as expected for a particular software application

Question:

Write a program to print the output:

Einstein [0] = E

Einstein [1] = I

Einstein [2] = N

Einstein [3] = S

Einstein [4] = T

Einstein [5] = E

Einstein [6] = I

Einstein [7] = N

Data Driven Testing:

Data that is loaded and utilized to amplify our automated test cases but is not part of our functional tests

Data Flow Testing:

A technique for examining the software program's data flow

Solution:

Defect Logging and Tracking:

A methodology of identifying flaws in an software application under test through testing or capturing customer comments, followed by the development of subsequent iterations of the application that address the flaws or the client's comments

```
#include<stdio.h>
int main() {

int i;
char name [8] = {' E', ' I', ' N', ' S', ' T ', ' E', ' I', ' N'};
for(i=0; i<8; i++)
printf("\n Einstein [%d] = %c", i, name[i]);
return 0;
}
```

Question 72

Depth Testing:

A testing method that thoroughly examines each feature of a software application

Question:

Write a program to find square of a number using functions.

Solution:

Destructive Testing:

A testing method that causes the software application to crash uncontrollably in order to assess the application's resilience and pinpoint the reason of failure

```
#include<stdio.h>
int square();
int main() {

int answer;
answer = square();
printf("Square of the given number: %d", answer);
return(0);
```

```
}
int square() {
int x;
printf("Enter any integer: \n");
scanf("%d", &x);
return x*x;
}
```

Domain Testing:

A method for testing software that involves picking a tiny portion of test cases from an almost unlimited collection of test cases

---

Question 73

Durability Testing:

A method of software testing that offers a precise evaluation of how our software application might function in both typical and unexpected circumstances

Question:

Write a program to print "Hello world" 10 times.

Solution:

```
#include<stdio.h>
int main() {

int i;
for(i =1; i<=10; i++)
printf("Hello world \n");
return 0;
}
```

Endurance Testing:

A software testing methodology where a software system is tested under huge workload for a long period of time in order to assess how the system will behave when used continuously

---

Error Seeding:

The process of purposefully injecting bugs into a software program to determine whether test cases can catch the bugs

Question 74

End-to-End Testing:

A testing procedure for estimating whether a software application's flow functions as planned from beginning to end

Question:

Write a program to print first 5 numbers using do while loop statement.

Solution:

- **Fair coin:** A coin with an equal chance of landing heads-up or tails-up
- **Fair game:** A game in which every participant has an equal opportunity to succeed
- **Infinite sample space:** A sample space with an essentially limitless number of outcomes

```
#include<stdio.h>
int main() {

int i =1;
do {
printf("%d\n", i++);
} while (i<=5);
return 0;
}
```

Question 75

Entry Criteria:

A set of circumstances under which a test activity can be completed, or in the absence of any of these circumstances, the test activity cannot be completed

Question:

Write a program to print the output:

body [b] = b

body [o] = o

body [d] = d

body [y] = y

Error Guessing:

A type of testing methodology in which previous testing expertise is utilized to find software flaws

Solution:

Exhaustive Testing:

The method of thoroughly testing a software application to ensure that it cannot be damaged or crashed by an unforeseen circumstance

```
#include <stdio.h>
int main() {

char i;
char body [4] = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
printf("\n body[%c] = %c", body[i] , body[i]);
return 0;
}
```

Quality gate:

Software that constantly evaluates and keeps track of the source code's quality

Question 76

Exit Criteria:

The standards or specifications that are used to evaluate if a particular test activity has been finished or NOT

Question:

Write a program to check whether a character is an alphabet or not.

Solution:

Expected Outcome:

The ideal output that should be obtained after running a test case

Inter Systems Testing:

A methodology for evaluating the integration points and functionalities between several systems that share same data source

```
#include <stdio.h>
#include <ctype.h>
int main() {

int a =2;
if(isalpha(a)) {
printf("The character a is an alphabet");
}
```

```
else {
printf("The character a is not an alphabet");
}
return 0;
}
```

Symbolic execution:

A technique for examining program to ascertain what inputs trigger the execution of each part of a program

Question 77

Globalization Testing:

A kind of software testing carried out to make sure the software application can operate regardless of the location and environments shaped by human activities

Question:

Write a program to calculate the discounted price and the total price after discount

Given:

If purchase value is greater than 1000, 10% discount

If purchase value is greater than 5000, 20% discount

If purchase value is greater than 10000, 30% discount.

Solution:

Keyword Driven Testing:

A methodology that involves using data files to store keywords associated with the software application being tested

Localization Testing:

A method for evaluating the performance, reliability and applicability of software application for certain areas and regions

```
#include<stdio.h>
int main() {

double PV;
printf("Enter purchased value:");
scanf("%lf", &PV);
if(PV>1000) {
printf("Discount=%lf \n", PV* 0.1);
printf("Total=%lf \n", PV - PV* 0.1);
}
else if(PV>5000) {
```

```
printf("Discount =%lf \n", PV* 0.2);
printf("Total=%lf \n", PV - PV* 0.2);
}
else {
printf("Discount=%lf \n", PV* 0.3);
printf("Total=%lf \n", PV - PV* 0.3);
}
return 0;
}
```

Negative Testing:

Testing carried out to make sure the software application being tested DOES NOT malfunction when an unforeseen input is provided

---

Question 78

Partial Automation Testing:

The method used by testers − which combines manual and automated tests

Question:

Write a program to print the first ten natural numbers using while loop statement.

Solution:

Progressive Testing:

Testing modules one after another

```
#include<stdio.h>
int main() {

int i = 1;
while(i<=10) {
printf("%d\n", i++);
}
return 0;
}
```

Positive Testing:

A kind of testing that involves using authentic datasets as input to test a software application

---

Question 79

Prototype Testing:

Testing carried out to look for issues before the software application goes live

Question:

Write a program to check whether the person is a senior citizen or not.

Solution:

Release Notes:

Technical documents developed and released in association with the release of a new software application or an application update

```c
#include<stdio.h>
int main() {

int age;
printf("Enter age: \n");
scanf("%d", &age);
if(age>=60) {
printf("senior citizen");
}
else {
printf("not a senior citizen");
}
return 0;
}
```

Requirements traceability:

Keeping track of requirements during the entire software application development process

One-tail test:

A statistical test designed to demonstrate that the sample mean would either be more than or less than the population mean, but not both

---

Question 80

Requirement Based Testing:

A testing strategy in which requirements are used to generate test cases, conditions and data

Question:

Write a program to find the size of data types.

Solution:

Scenario Testing:

A method of testing software applications that does not rely on test cases but rather on real-world scenarios

```
#include<stdio.h>
int main() {

printf("Size of char: %ld byte\n",sizeof(char));
printf("Size of int: %ld bytes\n",sizeof(int));
printf("Size of float: %ld bytes\n",sizeof(float));
printf("Size of double: %ld bytes", sizeof(double));
return 0;
}
```

Thread Testing:

A software testing method used to check if a software application can execute a particular task or thread or not

---

Question 81

State Transition Testing:

A black box testing strategy used to track how a software application behaves when a series of various input conditions are sent through it

Question:

Write a program to check whether a triangle is valid (given sides of triangle).

Solution:

Structural Testing:

A white box testing strategy where the tests are designed by understanding the underlying workings of the software application

```
#include <stdio.h>

int main() {
int side1, side2, side3;

    printf("Enter Length of Three Sides of a Triangle: \n");
    scanf("%d %d %d", &side1, &side2, &side3);
```

```
    if((side1 + side2 > side3)&&(side2 + side3 > side1) &&(side3 + side1 > side2)) {
        printf("It is a Valid Triangle");
    } else {
        printf("It is an invalid Triangle");
    }

    return 0;
}
```

Storage Testing:

A software testing method carried out to verify if the application being tested saves the necessary information in the proper directories and has adequate space to prevent unexpected application terminations brought on by a lack of disc space

Question 82

Question:

Write a program to check whether a triangle is valid (given angles of triangle).

Solution:

Shift Left:

A methodology used early in the software delivery process to identify and prevent BUGS

```
#include<stdio.h>
int main() {

int angle1, angle2, angle3, sum;
printf("Enter Three Angles of a Triangle : \n");
scanf("%d%d%d", &angle1, &angle2, &angle3);
sum = angle1 + angle2 + angle3;
if(sum == 180) {
  printf("This is a Valid Triangle");
}
else {
  printf("This is an Invalid Triangle");
}
```

Unreachable Code: A section of the source code that can never be executed because of improper control flow or exit points

Shift right:

The act of conducting testing, quality assurance and performance evaluation in real production environments

```
 return 0;
}
```

Question 83

Joint probability:

A statistical measure that determines the probability of 2 events happening simultaneously and at the same time

Question:

Write a program to print your name, date of birth and email address.

Solution:

Verification Testing:

The process of verifying that the software complies with all established software requirements including documents, code, designs and programs

```
#include <stdio.h>
int main() {

printf("Name   : Albert Einstein\n");
printf("DOB    : 14 March 1879\n");
printf("Email : einstein@physics.com\n");
return(0);
}
```

Volume Testing:

A kind of software testing that uses a large amount of data to test a software application

---

Question 84

Exogenous variable:

A variable that is unaffected by other variables in the system

Question:

Write a program to convert specified days into years, weeks and days.

Solution:

Service Level Indicator:

A measurement of the level of service that a service provider offers to a consumer

```c
#include <stdio.h>
int main() {

int days, years, weeks;
printf("Enter number of days: \n");
scanf("%d", &days);
years = days/365;
weeks = (days % 365)/7;
days = days-((years*365) + (weeks*7));
printf("Years: %d\n", years);
printf("Weeks: %d\n", weeks);
printf("Days: %d", days);
return 0;
}
```

Basis Test Set:

A series of tests performed on a software component's internal structures to ensure complete coverage of a given requirement

Question 85

Loop Testing:

A type of software testing that only examines if loop constructs are accurate

Question:

Write a program to Check the password until it is correct.

Solution:

Coverage Item: An entity or characteristic that serves as the foundation for test coverage

```c
#include <stdio.h>
int main() {
```

```
int pass, a;
while (a!=0) {
    printf("Input the password: \n");
    scanf("%d", &pass);

    if (pass==1988) {
        printf("Correct password");
        a=0;
    }
    else {
       printf("Wrong password, try again");
    }
       printf("\n");
   }
return 0;
}
```

Feedback loops:

Establishing quick and constant feedback between operations and development in the earliest stages of the software delivery process

Policy as code:

The concept of managing and automating policies by writing code in a high-level language

---

Question 86

Bitmap:

An image file format that is suitable for storing and producing computer graphics

Question:

Write a program to Read an integer and find its entire divisor.

Solution:

A variable without restrictions is said to be **free variable**. On the other hand, a **bound variable** is a variable with restrictions

```
#include <stdio.h>
int main() {

int x, i;
printf("Enter a number : \n");
```

```
 scanf("%d", &x);
 printf("All the divisor of %d are: \n", x);
    for(i = 1; i <= x; i++) {
        if((x%i) == 0){
            printf("\n %d", i);
            printf("\n");
        }
   }
return 0;
}
```

CommonJS:

A set of guidelines for implementing JavaScript modules

Fingerprinting:

The practice of correlating data sets to discover network services, software programs, databases, configurations, operating system details, and more

---

Question 87

Context switch:

A method that the central processing unit **(CPU)** of a computer uses to switch from one task (**or process**) to another while making sure the tasks do not clash

Question:

Write a program to shift given data by two bits to the left.

Solution:

Customized built-in elements:

Custom elements that extend a standard HTML tag provided by the browser

```
#include<stdio.h>
int main() {

int a, b;
printf("Enter a integer : \n");
scanf("%d", &a);
printf("Integer value = %d \n", a);
a<<=2;
b=a;
printf("The left shifted data is = %d \n", b);
return 0;
```

Display advertising:

The practice of promoting a service or product using visuals such as pictures and videos on networks of publisher websites like Facebook and Google Display Network, etc.

```
}
```

Question 88

Folksonomy:

The method of categorizing or annotating digital content with tags

Question:

Write a program to find Absolute Value of a Number.

Solution:

Graceful Degradation:

The capacity of a computer, machine, electronic system, or network to keep operating partially even after a significant section of it has been damaged or rendered ineffective

```c
#include<stdio.h>
#include<stdlib.h>
int main() {

int num;
printf("Enter a positive or negative number: \n");
scanf("%d", &num);
printf("Absolute Value of |%d| is %d", num, abs(num));
return 0;
}
```

```python
import itertools

x = itertools.count(11, .5)

print(next(x)) # Output: 11

print(next(x)) # Output: 11.5

print(next(x)) # Output: 12.0

print(next(x)) # Output: 12.5
```

```python
x = 'Einsteinian %s to %s Papers'

print(x % (1905, 1915))

# Output: Einsteinian 1905 to 1915 Papers
```

Question 89

Jamstack:

Architecture created to increase the web's speed, security, and scalability. It expands on many of the procedures and technologies that maximize efficiency for developers

Question:

Write a program to accept the height of a person in centimeter and categorize the person according to their height.

MAMP:

- **macOS** (the operating system)
- **Apache** (the web server)
- **MySQL** or **MariaDB** (the database management system)
- **PHP**, **Perl**, or **Python** (programming languages used for web development)

Solution:

```
#include <stdio.h>
int main() {

float height;
printf("Enter the height of the person (in cm) : \n");
scanf("%f", &height);

  if (height < 150.0) {
       printf("The person is Dwarf.");
    }
  else if ((height >= 150.0) && (height < 165.0)) {
        printf("The person is  average heighted.");
    }
  else if ((height >= 165.0) && (height <= 195.0)) {
        printf("The person is taller.");
    }
  else {
        printf("Abnormal height.");
    }
return 0;
}
```

Object-Oriented CSS:

A coding methodology that applies strong, reusable styles to "objects" or "modules" − nestable HTML blocks that constitute a portion of a website

Obfuscation

The practice of writing machine or source code that is challenging for humans to comprehend

Question 90

Offline-First:

A model for programming where the software application must function equally well offline as it does online

Question:

Write a program to compute the area of the various geometrical shape.

Solution:

Open Web Application Security Project:

An online group or network that creates free resources on web application security, including papers, techniques, documentation, tools, and technologies

```
#include <stdio.h>
int main() {

int choice, r, l, w, b, h;
float area;
printf("Enter 1 for area of circle\n");
printf("Enter 2 for area of rectangle\n");
printf("Enter 3 for area of triangle\n");
printf("Enter your choice : ");
scanf("%d", &choice);

switch(choice) {
           case 1:
                 printf("Enter the radius of the circle : \n");
                 scanf("%d", &r);
                 area=3.14*r*r;
                 break;
            case 2:
                  printf("Enter the length and width of the rectangle : \n");
                  scanf("%d%d", &l, &w);
                  area=l*w;
                  break;
```

Palpable content:

When content is rendered and substantive, it is palpable since it is neither empty nor concealed

TOML: A basic configuration file format with clear semantics that is easy to read

```
            case 3:
                  printf("Enter the base and height of the triangle : \n");
                  scanf("%d%d", &b, &h);
                  area=0.5*b*h;
                  break;
          }
printf("The area is : %f", area);
return 0;
}
```

Robots exclusion standard:

A protocol that websites employ to communicate with web crawlers and other web robots

---

Question 91

Round Trip Time: 2 × Propagation delay

The amount of time, measured in milliseconds (ms), it takes for a network request to go from its origin to its destination before returning to its origin

Question:

Write a program to Find Length of String Using Library Function.

Solution:

Sectioning content:

Definitional content for headings and footers

```
#include<stdio.h>
#include<string.h>
int main() {

char str[100];
int x;
printf("Enter the String : \n");
scanf("%s", str);
x = strlen(str);
printf("Length of Given String : %d", x);
return(0);
```

Service Level Expectation:

The standard that the service provider must adhere to in order to provide the services

Solution stack:

A collection of many applications or programs that are used to deliver the desired result or solution

```
}
```

Three-Click Rule:

An official standard that claims that a user should always be able to get the information they're looking for in three clicks or fewer, but there is no evidence to back this up, and it's now commonly believed to be a misconception rather than a regulation that should be rigidly followed

Question 92

Question:

Write a program to illustrate the bitwise operators.

Solution:

Transcompiler:

A specific kind of compiler that converts the source code of one programming language into the source code of another

```
#include<stdio.h>
int main() {

int a, b;
    a=125;
    b=35;
    a=a>>1;
printf("After right-shifting by 1, a=%d\n", a);
    b=b<<2;
printf("After left-shifting by 2, b=%d\n", b);
return(0);
}
```

TrustRank:

An algorithm that uses link analysis to distinguish between spam and valuable websites and aids in the fight against web spam

Trust On First Use

A security model where a client must establish trust with an unidentified server

Question 93

Type coercion:

Value conversion that takes place automatically or implicitly between different data types (**such as strings to numbers**)

Question:

Write a program to illustrate the ternary operation.

Typography:

The technique of organizing words and letters so that the reader can easily read, understand, and find the copy visually appealing

Solution:

```
#include<stdio.h>

/*

c will be assigned the value of a
if a is less than b.
Otherwise, it will be assigned the value of b

*/

int main() {

int a = 10, b = 20, c;
c = (a < b) ? a : b;
printf("%d", c);
 return(0);
}
```

Web bug:

Small visuals (**such as .gif, .jpeg, and.png**) are used by businesses and enterprise organizations in their **HTML-supporting emails**, web pages, and other documents to track information about the consumers of their online web services

Uninitialized variable:

A declared variable that isn't given a clear, predetermined value before use. When a variable is not initialized, its value is equal to whatever garbage value is in the memory assigned to the variable

UNIX philosophy:

A Philosophical approach that places a focus on creating code that is straightforward, concise, clear, modular, and extensible so that it may be readily maintained and reused by developers other than its original authors

Question 94

YAML Ain't Markup Language:

An input format for several software programs that uses a data-oriented language structure

Question:

Write a program to accept a character from the keyboard and print "yes" if it is equal to y. Otherwise print "No".

Solution:

```c
#include<stdio.h>
int main() {

char ch;
printf ("Enter a character: \n");
ch = getchar ();
if (ch == 'y' || ch == 'Y')
printf ("Yes");
else
printf ("No");
return(0);
}
```

Web worker:

A JavaScript code launched from an **HTML page** runs independently of other scripts that might have also been launched from the **same HTML page** in the background

Fine-tuning:

A methodology that takes a model that has already been trained to perform one task and adjusts or modifies the model to make it accomplish another similar task

---

Question 95

Ridge regression:

A model tuning technique that is applied to analyze any multicollinear data

Question:

Write a program to Demonstrate goto Statement.

Solution:

Discriminative models:

A group of **supervised machine learning models** that may be applied to classification or regression and make predictions by calculating conditional probabilities

Generative modeling:

A type of unsupervised machine learning in which the model learns to identify patterns in input data

Kernel Trick: An easy method for classifying data that cannot be linearly divided by a plane by projecting nonlinear data onto a higher dimension space

```c
#include <stdio.h>
int main() {

int sum=0;
for(int i = 0; i<=10; i++){
sum = sum+i;
  if(i==4){
      goto addition;
    }
}

  addition:
  printf("%d", sum);
  return 0;
}
```

Question 96

Server farm:

A large group of servers, possibly thousands, working together to improve functionality and accessibility

Question:

Write a program to Demonstrate Infinite Loop.

Solution:

The correlation's value ranges from -1 to +1. On the other hand, the covariance's value is between -∞ and +∞

```c
#include <stdio.h>
```

```c
int main() {

int i = 10;
for( ; ;) {
printf("%d\n", i);
}
return 0;
}
```

Data annotation:

The labeling and classification of data for AI applications

It is always preferable to use the **median** rather than the **mean** when the data is skewed. The more skewed the data is, the less accurate is the mean and the more accurate is the median

---

Question 97

A/A Testing:

A technique that involves comparing two identical versions of the same web application to see whether there is any difference

Question:

Write a program to Demonstrate Macro Substitution.

Solution:

A/Z Testing:

More than two variations of the same concept are being compared

```c
#include <stdio.h>
#define A 15
int main() {

int x;
x=A;
printf("%d", x);
return 0;
}
```

Multivariate:

Comparing multiple presentations of the same concept

AdaBoost:

A meta-learning approach to ensemble learning that was initially developed to boost the effectiveness of binary classifiers

---

Question 98

AdaGrad:

A family of sub-gradient stochastic optimization techniques

Question:

Write a program to Check Whether a Number is Even or Odd.

Solution:

$$\text{Amount of say} = \frac{1}{2}\log\left(\frac{1-\text{Total error}}{\text{Total error}}\right)$$

$\downarrow$

will be positive when the sample is **miss-classified**

```c
#include <stdio.h>
int main() {

int num;
printf("Enter a number : \n");
scanf("%d", &num);
   if(num % 2 == 0)
        printf("%d is even.", num);
   else
        printf("%d is odd.", num);
return 0;
}
```

Adaptive Moment Estimation (ADAM):

A stochastic gradient descent replacement optimization technique for deep learning models

---

Question 99

Agent:

In **reinforcement learning**, an agent is the active component that decides what should be performed next

Question:

Write a program to multiply given number by 4 using bitwise operators.

Solution:

AlexNet:

Deep learning image recognition architecture

```c
#include <stdio.h>
int main() {

    long number, x;
    printf("Enter a integer: \n");
    scanf("%ld", &number);
    x = number;
    number = number << 2;
    printf("%ld × 4 = %ld\n", x, number);
    return 0;
}
```

You Only Look Once (YOLO):

A real-time object detection algorithm that recognizes objects using features obtained from a deep convolutional neural network

Question 100

Attention mechanism:

Instead of using the complete input sequence, the ML model just considers portions of it where the most significant information is concentrated when predicting an output word. In other words, it just takes into account a select few input words

Question:

Write a program to check whether a number is power of 2 or not.

Solution:

Auto Regressive Integrated Moving Average (ARIMA):

A statistical analysis method that employs time series data to forecast future patterns or to better comprehend the current data collection. If a **methodology** forecasts future values using data from the past, it is said to be **autoregressive**

```c
#include <stdio.h>
int main() {

int num;
printf("Enter a number : \n");
scanf("%d", &num);
```

```
if((num != 0) && ((num &(num-1)) == 0)) {
      printf("%d is a power of 2", num);
}
else {
      printf("%d is not a power of 2", num);
}
return 0;
}
```

Attribute:

The feature of an observation

Area under the curve (AUC)

Assist in assessing a classification model's effectiveness. The performance of the model improves with increasing **area under the curve**. AUC of 1 indicates a flawless model, whereas AUC of 0 indicates a model that is completely incapable of differentiating between classes

Question 101

Question:

Write a program to check whether the triangle is equilateral, isosceles or scalene.

Solution:

Autocorrelation:

The extent of similarity (**correlation**) between close-by observations. The autocorrelation ranges from -1 to 1, with -1 denoting a negative correlation, 1 denoting a positive correlation, and 0 denoting no correlation

```
#include <stdio.h>
int main() {

int side1, side2, side3;
printf("Enter the sides of triangle : \n");
scanf("%d%d%d",&side1,&side2,&side3);
if(side1 == side2 && side2 == side3) {
      printf("The given Triangle is equilateral.");
}
else if(side1 == side2 || side2 == side3 || side3 == side1) {
      printf("The given Triangle is isosceles.");
```

```
}
else {
      printf("The given Triangle is scalene.");
   }
return 0;
}
```

Automatic summarization:

The method of conveying the desired message by using a software program to condense lengthy texts and produce summaries

---

Question 102

"Automatic text summarization is the task of producing a concise and fluent summary while preserving key information content and overall meaning."

– Text Summarization Techniques: A Brief Survey, 2017

Question:

Write a program to print ASCII values of all the letters of the English alphabet from A to Z.

Solution:

Automation bias:

The tendency for people to dismiss contradicting information supplied without automation, even if it is true, and to favor ideas from automated decision-making systems

```
#include <stdio.h>
int main() {

int i;
for(i='A'; i<='Z'; i++) {
     printf("ASCII value of %c = %d\n", i, i);
}
return 0;
}
```

Black box

A system in which the input and output are the only things that are known

---

Question 103

Candidate generation:

How many actual recommendations will be used to create a recommender system

Question:

Write a program to find sum of even numbers between 1 to n.

Solution:

Bootstrap aggregating:

An approach to ensemble learning that boosts the efficiency and precision of machine learning algorithms

```
#include <stdio.h>
int main() {

int i, num, sum=0;
printf("Enter a number: \n");
scanf("%d", &num);
for(i=2; i<=num; i=i+2) {
     sum = sum + i;
 }
printf("Sum of all even number between 1 to %d: %d", num, sum);
return 0;
}
```

Baseline:

The standard by which any future outcomes can be measured

Question 104

Baseline model:

A basic model that is trained first, and whose performance serves as the standard for all future algorithm training and performance

Question:

Write a program to find sum of odd numbers between 1 to n.

Solution:

Candidate sampling:

A method to reduce training time for problems involving many labels

Databricks → Unified data science platform

```c
#include <stdio.h>
int main() {

int i, num, sum=0;
printf("Enter a number: \n");
scanf("%d", &num);
for(i=1; i<=num; i=i+2) {
        sum = sum + i;
}
printf("Sum of all odd number between 1 to %d: %d", num, sum);
return 0;
}
```

Question 105

Centroid:

The center of every cluster in models of unsupervised machine learning

Question:

Write a program to find maximum number using switch case.

Solution:

Centroid-based algorithm:

An unsupervised machine learning approach to clustering where data is arranged around a cluster center

```c
#include <stdio.h>
int main() {

int num1, num2;
printf("Enter two numbers : \n");
scanf("%d%d", &num1, &num2);
```

```
switch(num1 > num2) {
        case 0: printf("%d is Maximum number", num2);
            break;

        case 1: printf("%d is Maximum number", num1);
            break;
}
return 0;
}
```

Checkpoint:

A technique that periodically saves the models while training

---

Question 106

CIFAR-10:

A dataset that is well-understood and frequently used in the field of machine learning to evaluate computer vision algorithms

Question:

Write a program to create simple calculator using switch case.

Solution:

$$\text{Cohen's kappa} = \frac{\text{observed agreement} - \text{expected agreement}}{1 - \text{expected agreement}}$$

is always less than or equal to 1. Values of 0 or less signify the classifier is ineffective

```
# include <stdio.h>
int main() {

char operator;
double x, y;
printf("Enter an operator (+, -, *,): \n");
scanf("%c", &operator);
printf("Enter two operands: ");
scanf("%lf %lf", &x, &y);

switch(operator) {
        case '+':
```

Connectivity-based algorithm:

A machine learning approach that groups mutually related observations into clusters for unsupervised machine learning tasks

```
          printf("%.1lf + %.1lf = %.1lf", x, y, x + y);
            break;

        case '-':
            printf("%.1lf - %.1lf = %.1lf", x, y, x - y);
            break;

        case '*':
            printf("%.1lf * %.1lf = %.1lf", x, y, x * y);
            break;

        case '/':
            printf("%.1lf / %.1lf = %.1lf", x, y, x / y);
            break;

        default:
            printf("Error! Please enter correct operator");
    }
return 0;
}
```

Convergence:

A machine learning model reaches convergence when a trained model no longer sees its error decreased or only slightly if we add additional data for training

---

Question 106

Continuous learning:

The potential of a ML model to learn in a production environment while being trained with a stream of real-time data

Question:

Write a program to find Square Root of a Number.

Solution:

Convenience sampling:

A kind of sampling where the first main data source that is accessible will be used for the study without any further criteria

```
#include<stdio.h>
```

```c
#include<math.h>
int main() {

int number;
double result;
printf("Enter any integer number: \n");
scanf("%d", &number);
result = sqrt(number);
printf("Square root of %d is %.2lf", number, result);
return 0;
}
```

Cosine similarity:

A parameter used in text mining to evaluate how similar documents are

---

Question 107

Data preparation:

The phase of a data science project where unclean and unusable raw data is cleaned up and processed to produce clean and useable data for data investigation or modeling

Question:

Write a program to Find Power of a Number Using Library Function.

Solution:

DataFrame:

A two-dimensional data structure consisting of rows and columns

```c
#include<stdio.h>
#include<math.h>
int main() {

int base, exponent;
long power;
printf("Enter Base and Exponent: \n");
scanf("%d %d", &base, &exponent);
power = pow(base, exponent);
printf("Result = %ld", power);
```

Davies-Bouldin index:

The index that uses the separations between cluster centers to evaluate how similar the clusters are to one another

```
return 0;
}
```

Question 108

Density-Based Spatial Clustering of Applications with Noise:

A density-based clustering technique for unsupervised machine learning problems that creates clusters by combining points that are close to one another

Question:

Write a program to check whether the entered Character is Alphabet or Digit.

Solution:

Decile:

A descriptive statistic with nine values dividing the data into ten equally sized pieces. Each part represents one-tenth of the data

```
#include <stdio.h>
#include<ctype.h>
int main() {

char ch;
printf("Enter a character :  \n");
scanf("%c", &ch);
if (isdigit(ch)) {
    printf("%c is a Digit", ch);
  }
else if (isalpha(ch)) {
    printf("%c is an Alphabet", ch);
  }
else {
    printf("%c is not an Alphabet, or a Digit", ch);
}
return 0;
}
```

Decision boundary

A technique that divides the vector space into areas, each of which has a class assigned to it, while solving a classification problem

Eclipse Deeplearning4j:

A Java-based programming library made for the Java virtual machine. It is a framework that offers deep learning algorithms broad support

Question 109

DeepDream:

A **Google-developed computer vision approach** that generates images or transforms existing images into a dreamlike vision using a deep convolutional neural network

Question:

Write a program to check whether the entered character is alphanumeric or not.

Solution:

Dense feature:

A tensor or array with a large percentage of non-zero values

```
#include<stdio.h>
#include<ctype.h>
int main() {

char a;
printf( "Enter a character : \n");
scanf( "%c", &a );
if(isalnum(a)) {
     printf( "%c is an alphanumeric character.", a );
}
else {
        printf( "%c is NOT an alphanumeric character.", a );
}
return 0;
}
```

Dplyr:

A robust R package for manipulating data. It makes data transformation, cleaning, and summarization really simple

Dropout:

A regularization technique that simulates the concurrent training of several neural networks with various architectures

Question 110

Dunn index = $\frac{\text{The smallest intercluster distance}}{\text{The largest intracluster distance}}$

A high **Dunn index** indicates better clustering since observations inside each cluster are closer together while the clusters themselves are farther apart

Question:

Write a program to check whether the entered character is punctuation character or not.

Embedding space:

A relatively low-dimensional space into which high-dimensional vectors can be translated

Solution:

```c
#include<stdio.h>
#include<ctype.h>
int main() {

char a;
printf( "Enter a character : \n");
scanf( "%c", &a );
if(ispunct(a)) {
       printf( "%c is an punctuation character.", a );
}
else {
        printf( "%c is NOT an punctuation character.", a );
}
return 0;
}
```

Feature cross:

A feature created synthetically by multiplying **(crossing)** two or more features

```python
i = 1

while i < 4:

  print('Alan')

  i = i + 1
```

Output:

Alan

Alan

Alan

Question 111

Apache flume:

A distributed, dependable, and accessible piece of software for quick gathering, combining, and transporting enormous amounts of log data

Question:

Write a program to check whether the entered character is a character with graphical representation or not.

Solution:

Federated learning:

A method of machine learning that trains an algorithm on a number of distributed edge devices or servers that keep local data samples without exchanging them

```c
#include<stdio.h>
#include<ctype.h>
int main() {

char a;
printf( "Enter a character : \n");
scanf( "%c", &a );
if(isgraph(a)) {
        printf( "%c is a character with graphical representation.", a );
}
else {
        printf( "%c is NOT a character with graphical representation.", a );
}
return 0;
}
```

```
x = ' ' * 6

print('%s 1905 Papers' % x)

# Output:

        1905 Papers
```

```
print(5 + int('5'))

# Output: 10

print(5.3 + float('5.3'))

# Output: 10.6
```

Question 112

Generalization curve:

A curve that displays the error or loss during training and validation over several iterations

Question:

Write a program to List all Printable Characters Using isprint() function.

Forget gate:

A concept in the recurrent neural network's long short-term memory that determines how much of the past information will be discarded and how much of the past information will be used in subsequent steps

Solution:

```c
#include <stdio.h>
#include <ctype.h>
int main() {

int i;
for(i = 1; i <= 127; i++)
   if(isprint(i)!= 0)
         printf("%c ", i);
return 0;
}
```

Generator

A part of a generative adversarial neural network that creates bogus data out of randomness in a latent space

Question 113

Frechet inception distance:

A metric for evaluating the quality of images produced by generative models such as generative adversarial networks

Question:

Write a program to check whether the entered character is a hexadecimal digit character or not.

Intersection over Union = $\frac{\text{Area of Overlap}}{\text{Area of Union}}$ **:**

A statistic used to assess the effectiveness of the machine learning model for image detection

Solution:

```
#include<stdio.h>
#include<ctype.h>
int main() {

char a;
printf("Enter a character : \n");
scanf( "%c", &a );
if(isxdigit(a)) {
       printf("%c is a hexadecimal digit character.", a );
}
else {
        printf("%c is NOT a hexadecimal digit character.", a );
}
return 0;
}
```

Ggplot2:

A well-liked R programming library for data visualization

Highway network:

A neural network that assists in very deep neural network training

---

Question 114

Greedy policy:

A reinforcement learning strategy that chooses an action by attempting to maximize the reward or return

Question:

Write a program to Print ASCII value of All Control characters.

Solution:

```
#include <stdio.h>
#include <ctype.h>
```

Apache Hadoop:

A software framework that enables the distributed processing of massive data sets utilizing simple programming models across computer clusters

```
int main() {

int i;
printf("The ASCII value of all control characters are: \n");
for (i=0; i<=127; ++i) {
        if (iscntrl(i)!=0)
            printf("\n %d ", i);
}
return 0;
}
```

Implicit bias:

Making an incorrect or partially true assumption before beginning a machine learning project

---

Question 115

ImageNet large scale visual recognition challenge:

A yearly software competition in computer vision and artificial intelligence based on the **ImageNet project** that compares object detection and image classification techniques

Question:

Write a program to illustrate isprint() and iscntrl() functions.

Solution:

Inception:

The deep learning architecture for image classification, particularly CNNs (convolutional neural networks)

```
#include <stdio.h>
#include <ctype.h>
int main() {

char ch = 'a';
if (isprint(ch)) {
        printf("%c is printable character \n", ch);
} else {
        printf("%c is not printable character", ch);
}
```

Instance-based learning:

A sort of memory-based learning machine learning method that compares newly arriving data with previously stored data rather than generalizing based on training data

```
if (iscntrl(ch)) {
        printf("%c is control character\n", ch);
} else {
        printf("%c is not control character", ch);
}
return (0);
}
```

Kernel support vector machine:

A classification technique that transforms the input data vectors into a higher-dimensional space to address the issue as though it were non-linear

Question 116

Latent variables:

Variables that are derived from observable variables but cannot be observed directly

Question:

Write a program to check whether the entered character is a white-space character or not.

Iteration:

How often a machine learning model will update its parameters

Solution:

```
#include <stdio.h>
#include <ctype.h>
int main() {

char c;
printf("Enter a character : \n");
scanf("%c", &c);
if (isspace(c) == 0) {
        printf("Not a white-space character.");
}
else {
```

Kullback–Leibler divergence:

A statistic to assess how two probability distributions differ from one another. It can be used to evaluate how far the continuous and discrete probability distributions diverge

```
        printf("White-space character.");
}
return 0;
}
```

```
print(list(zip(range(4), 'ALAN')))

# Output: [(0, 'A'), (1, 'L'), (2, 'A'), (3, 'N')]

print(list(zip(range(4), 'ALAN', [1, 2, 3, 4])))

# Output: [(0, 'A', 1), (1, 'L', 2), (2, 'A', 3), (3, 'N', 4)]


import itertools

print(list(itertools.accumulate([1, 2, 3, 4, 5])))

# Output: [1, 3, 6, 10, 15]

# [1, 1+2, 1+2+3, 1+2+3+4, 1+2+3+4+5]
```

```
import itertools

print(list(itertools.repeat(7, 3)))

# Output: [7, 7, 7]


print(list(itertools.repeat(6, 2)))

# Output: [6, 6]


x = itertools.repeat(6)

print(next(x), next(x), next(x))

# Output: 6 6 6
```

```
print(True + False == 1)

# Output: True (1 + 0 == 1)

print(True * True == 1)

# Output: True (1 * 1 == 1)
```

```
i = 3
for a in range(1, 5):
    print('Albert %s' % a)
    if a == i:
      print('Einstein')
      break
```

Output:

Albert 1

Albert 2

Albert 3

Einstein

```
i = 3
for a in range(1, 5):
    print('Albert %s' % a)
    if a == i:
      print('Einstein')
      pass
```

Output:

Albert 1

Albert 2

Albert 3

Einstein

Albert 4

C# program:

```
public class HelloWorld {
public static void Main(string[] args) {
System.Console.WriteLine("Albert Einstein");
 }
}
# Output: Albert Einstein
```

```
x= ('Albert', 1905)
y = x + ('Papers',)
print(y)
# Output:('Albert', 1905, 'Papers')
```

```
for _ in range(4):
 print('Albert Einstein')
```

Output:

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

```
x = ['5.36', '4.39', '3.55']
print([float(i) for i in x])
# Output: [5.36, 4.39, 3.55]


x = [2, 4, 6]
print([i* 2.16 for i in x])
# Output: [4.32, 8.64, 12.96]
```

```
print(('Albert', object(), 1905))
# Output:('Albert', <object object at 0x000001E34E03BF00>, 1905)
```

```
import dis # dis() function generates disassembled representation of the Python code
dis.dis(compile("(19, '05', 'Papers', 'Albert')", '', 'eval'))


# Output:
        1           0 LOAD_CONST               0 ((19, '05', 'Papers', 'Albert'))
                    2 RETURN_VALUE
```

```
for i in range(3):
  print(i, "\t", 3**i)
```

 # Output:

0 1

1 3

2 9

```
x = {1905: 'Papers'}
x[1915] = 'Papers'
print(x)
# Output:
{1905: 'Papers', 1915: 'Papers'}

x = str("1905 Papers")
print(x)
# Output: 1905 Papers

x = "Alb"
print(x + "ert" == "Albert")
# Output: True

x=2
print(x * 0 == 0)
# Output: True

print(not (not x) == x)
# Output: True
```

```
print ('\\')

# Output: \


print ('A 5\'6" tall object')

# Output: A 5'6" tall object


i = 4

while i> 0:

 print (i)

 i= i-1
```

Output:

4

3

2

1

```
# replaces all occurrences
print('This is a string'.replace('i','x'))

# replaces first 2 occurrences
print('This is a string'.replace('i','x', 2))

# Output:

Thxs xs a strxng
Thxs xs a string
```

```
x = ['Apple', 'Orange', ['Banana', 'Grapes'], 'Figs', ['Guava']]
print(x, False)
print(x, True)

# Output:

    ['Apple', 'Orange', ['Banana', 'Grapes'], 'Figs', ['Guava']] False
     ['Apple', 'Orange', ['Banana', 'Grapes'], 'Figs', ['Guava']] True
```

```
print(11<12) # Output: True

print(11<12<13) # Output: True

print(11<12<13>14) # Output: False

print(11==12==13) # Output: False

print(12==12==12) # Output: True
```

```
x=11
y=12
z=13
print("x value is %i" %x)
print("y value is %d and z value is %d" %(y, z))

# Output:
                              x value is 11
                    y value is 12 and z value is 13
```

```
x="alan"

print(x.count('a')) # Output: 2

print(x.count('al')) # Output: 1
```

```
x = int(input("Please enter a number:"))

for a in range(1, x+1):

   print("+ " * a)


# Output:

Please enter a number:2 # entered number
+
+ +
```

OpenCV:

A free graphics library that was first created by Intel and focuses on real-time image processing

```
x=[5,10,15,20]

for i in x:

  if i==0:

    continue

  print("50/{} = {}".format(i, 50/i))
```

Output:

```
50/5 = 10.0

50/10 = 5.0

50/15 = 3.3333333333333335

50/20 = 2.5
```

```
x=[11,12,13,14,15]

for i in x:

  if i>=20:

   print("Failed to process")

   break

  print(i)

else:

   print("Successfully processed")
```

Output:

```
11
12
13
14
15
Successfully processed
```

```
print("{:06d}".format(5496))
print("{:8.3f}".format(549.6555))
print("{:08.3f}".format(549.6555))
print("{:08.3f}".format(549.65))
print("{:08.3f}".format(54961988.6555))
```

Output:

```
005496
 549.655
0549.655
0549.650
54961988.656
```

```
a='Albert'
b ='Einstein'
c=1905
d='Papers'
print("{} {}'s {} {}".format(a,b,c, d))
# Output: Albert Einstein's 1905 Papers

print("{0} {1}'s {2} {3}".format(a,b,c, d))
# Output: Albert Einstein's 1905 Papers

print("{w} {x}'s {y} {z}".format(w=a, x=b, y=c, z=d))
# Output: Albert Einstein's 1905 Papers
```

Part of speech tagging:

A method of identifying a word in a text (compilation) as belonging to a specific part of speech based on both its definition and its context

```
print("{:+d}".format(420)) # Output: +420

print("{:+d}".format(-420)) # Output: -420

print("{:+f}".format(420.840)) # Output: +420.840000

print("{:+f}".format(-420.840)) # Output: -420.840000
```

Information scent:

The effectiveness of appropriate messaging along the customer journey, as well as the presence of visual and textual signals that notify website visitors of the type of content that is available there

Nash equilibrium:

A game theory decision-making principle that claims a player can attain the desired result by adhering to their initial approach

Negative class:

There are two classes in binary classification, one of which is classified as positive and the other as negative. As an illustration, consider the contradiction **spam** or **not spam**, where **spam** is the positive class and **not spam** is the negative class

Neural Turing Machine:

A kind of recurrent neural network that incorporates programmable computers with neural networks

No free lunch theorem:

There isn't a single model that works best in all circumstances. It is typical in machine learning to test several models in order to find the one that works the best for a given problem because the assumptions of a good model for one issue could not hold true for another

Java Program:

```
public class MyClass {

public static void main(String[] args) {

for(int i = 0; i <= 25; i++) {

char x = (char) ('a' + i);

System.out.println(x);

}

}
```

Output

```
a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
q
r
s
t
u
v
w
x
y
z
```

Nominal variable:

A categorical variable without a clear hierarchy and having two or more categories. **For example**, a variable termed color may include the names of many colours, such as orange, green, black, and white. This is a nominal variable since it includes categories without any established hierarchy

Null accuracy:

A minimum level of accuracy determined by just projecting the most common class

NVIDIA:

A graphics processing unit designed for working with graphics and image processing

Python code:

```
import math

print(math.sin(3) + math.cos(3))

# Output: -0.8488724885405782
```

Q-function:

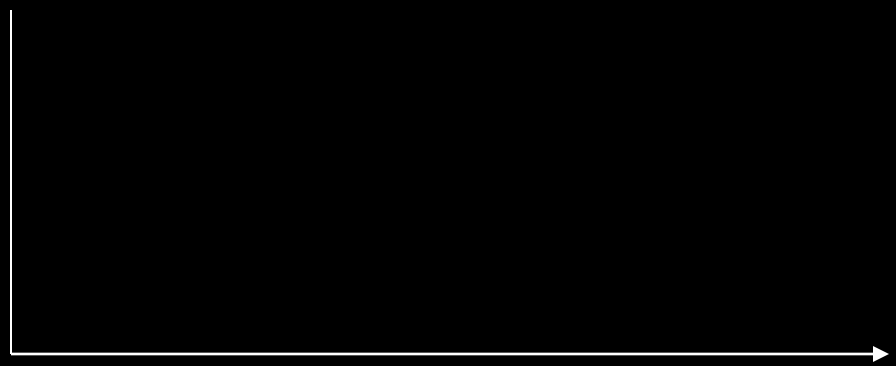
The probability that a normally distributed (**Gaussian**) random variable will have a value higher than the standard deviations

```
x = [True, "6", 5.0, 8]

print([type(i) for i in x])

# Output: [<class 'bool'>, <class 'str'>, <class 'float'>, <class 'int'>]
```

'i' is a type code indicating the contents are integers

```
import array

print(array.array('i', list(range(5))))
```

```
import numpy as np

np.random.seed(0) # seed for reproducibility

x = np.random.randint(10, size=6) # One-dimensional array

print(x)

# Output: [5 0 3 3 7 9]


y = np.random.randint(10, size=(3, 4)) # Two-dimensional array

print(y)

# Output:

                                    [[5 0 3 3]

                                     [7 9 3 5]

                                     [2 4 7 6]]


z = np.random.randint(10, size=(3, 4, 5)) # Three-dimensional array

print(z)

# Output
```



```
[[[5 0 3 3 7]
  [9 3 5 2 4]
  [7 6 8 8 1]
  [6 7 7 8 1]]

 [[5 9 8 9 4]
  [3 0 3 5 0]
  [2 3 8 1 3]
  [3 3 7 0 1]]

 [[9 9 0 4 7]
  [3 2 7 2 0]
  [0 4 5 5 6]
  [8 4 1 4 9]]]
```

Objective:

The goal that a machine learning algorithm seeks to maximize. Every machine learning algorithm must have a defined objective for each issue

Apache Oozie:

A server-based workflow management framework for scheduling Hadoop operations

```
import numpy as np

i = np.arange(3)

print(i)

# Output: [0 1 2]

print(i + 3)

# Output: [3 4 5]

print(i - 3)

# Output: [-3 -2 -1]

print(i * 2)

# Output: [0 2 4]

print(i / 2)

# Output: [0.  0.5 1. ]

print(i // 2) # floor division

# Output: [0 0 1]
```

```
import numpy as np

x = np.arange(3)

print(-(0.5*x + 1) ** 2)

# Output: [-1.   -2.25 -4.  ]
```

Parameter update:

The parameters of a model are updated depending on an iteration of gradient descent during the training of a **machine learning model**. Based on the calculated error or loss, this value will be updated

Pig:

An advanced programming language used with Apache Hadoop. Without learning Java, data engineers may perform sophisticated data transformations with Pig

Post processing:

The methodology used to refine and assess the information obtained by the data mining technique

Pre-processing:

A process of data transformation and cleaning in which unprocessed data is adjusted and made ready for machine learning algorithms. **Pre-processing** is an important stage in machine learning since it lowers interference and improves model performance

```
import numpy as np

i = np.array([11, 12, 13, 14, 15])


# Are all values equal to 15?

print(np.all(i == 15))

# Output: False


# Are there any values equal to 15?

print(np.any(i == 15))

# Output: True
```

Prior belief

All the presumptions made regarding the data, including distribution assumptions, correlation assumptions, assumptions on the data gathering process, and so forth

Random initialization:

The procedure of initializing the parameters of a machine learning model using random numbers

```
import pandas as pd

import numpy as np

x = pd.DataFrame(np.random.rand(4, 2),

            index=[['w', 'x', 'y', 'z'], [1, 2, 3, 4]], columns=['A', 'B'])

print(x)



# Output:

            A         B

w 1  0.921158  0.083112

x 2  0.277719  0.009357

y 3  0.842342  0.647174

z 4  0.841386  0.264730
```

Reward:

A crucial idea in reinforcement learning because it instructs the agent on how successfully it is carrying out a task and enhances the model as a whole

Relationship extraction

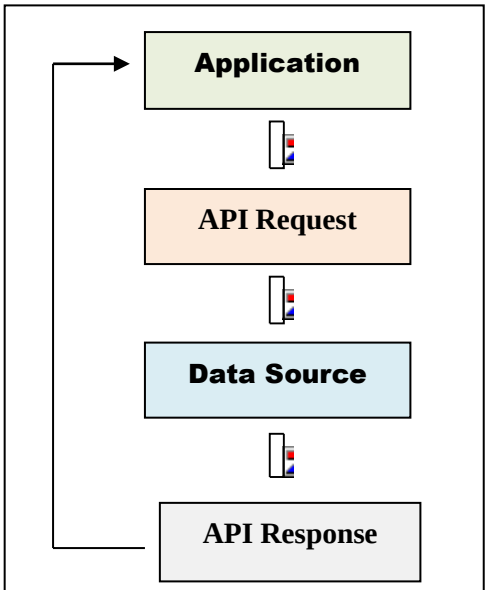
A methodology for extracting the semantic relationship between two or more entities in natural language processing. For example, the relationship between the entities **Bangalore** and **India** is conveyed by the word "**is in**" in the sentence "**Bangalore is in India**."

Cobra Effect:

Unexpected negative impacts when a reward or encouragement is provided to address a problem

Application Programming Interface (API)

A **code** that helps two different software applications to communicate and exchange data with each other



When you type an URL into browser – that is a GET Request

When you hit submit on a form – that is a POST Request

Margin of error:

The extent of error in survey findings that were chosen at random. A **higher statistical margin of error** denotes a lower possibility of relying on survey results

Multivariate test

A procedure that allows for the concurrent evaluation of variations of various elements on a webpage. We can find out which mix of variations performs best by using **multivariate testing**

Summary statistics:

Descriptive statistics that provide a concise summary of a dataset and can be used to extract important information

Cherry picking:

Choosing only the results that support your argument and discarding all others

An **API** act like a messenger who delivers our request for information to somebody and then delivers their response back to us.

- API makes **data management** more efficient
- API bridges the gap between 2 computer systems

An **API** can be used to securely transfer data stored by our web browser to a mobile application

- **Open API:** No restriction and available to the public.
- **Internal API:** Only available to internal systems of an enterprise.
- **Partner API:** Only available to certain groups of users through authentication and authorization mechanisms

- C → Create → **POST**
- R → Read → **GET**
- U → Update → **PUT**
- D → Delete → **DELETE**

HTTP Methods

Machine Instructions

The instructions executed by a **CPU**

Data Dredging:

Data mining techniques are misused in the practise of data dredging, also known as data snooping or data fishing, in order to present false **scientific** "**research**"

Survivorship Bias:

Focusing only on things that were chosen and ignoring others that were not, usually due to their lack of visibility, is a logical mistake. Inaccurate inferences may result from this

False Causality:

To incorrectly believe that if two things happen at the same time, one must have triggered the other

Gambler's Fallacy

↓

The false assumption that since something has occurred more frequently than usual, it will now happen less frequently in the future and vice versa

Boolean

A **logical data type** that can have only the values true or false

Latency

↓

The time delay between a user's action and a web application's response to that action

High latency → poor user experience

In networking, Latency is the time it takes for data to pass from one point on a network to another.

All computers consist of 3 main components:

- Computing
- Networking
- Storage

Cyber security:

The process of protecting data from unauthorized users

McNamara fallacy:

Making a choice based only on quantitative observations, while ignoring all other observations

Code review

A process where someone other than the author of a piece of code examines that code to find and remove the weaknesses or flaws in the code that impact software performance and security

Inputting more data than an application is expecting from a particular input — for example, by entering 500 characters into a field that was only expecting 50

Buffer overflow

Buffer overrun that occurs when a program attempts to put more data in a **buffer** than it can hold

An area of memory used to temporarily store data while it is being moved from one place to another

When more data gets placed by a program, the extra data overflows. It causes some of that data to leak out into other fixed length block of memory (**buffers**) – which can corrupt or overwrite whatever data they are holding.

Data processing steps:

- **Verification:** Ensures accurate and appropriate information
- **Sorting:** Arranges information in a specific order
- **Aggregation:** Combining several sources of information
- **Summarization:** Simplifies complex information into key points
- **Analysis:** Identifying, explaining and presenting underlying features of information
- **Data presentation** using a wide range of graphical methods to visually convey to the viewer the relationships between various data sets
- **Classification:** Separates data into various groups

Regression Toward the Mean:

A theory that states that if one sample of a random variable is extreme, the following sample of the same random variable will probably be closer to its mean

Hawthorne effect

When participants in an experiment try to alter or enhance their behavior solely because they are being examined or researched

3 principles that form the cornerstone of any organization's security infrastructure:

- Confidentiality
- Integrity
- Availability

Digital footprint

A record of our online activity

Remix

The process of creating a new version of a program by recombining and modifying parts of existing programs

Troubleshooting

A systematic approach to solve a problem

- Define the problem
- Identify the potential cause of problem
- Analyze
- Eliminate the potential cause of problem
- Propose Hypothesis
- Test Hypothesis
- Solve Problem
- Document solution

Publication Bias:

The likelihood of a research discovery being published depends on how entertaining it is, which distorts our perception of reality

Anchoring Bias:

Largely relying on the first piece of information one come across while making a decision **(treating future information as being less important)**

Stop words:

A **stop word** is a frequently used term that a search engine has been configured to ignore − both while indexing entries for searching and when retrieving them as the result of a search query. **Examples of stop words** include "the," "a," "an," and "in."

Parts-of-speech (POS) Tagging:

A methodology of identifying a word in a text (corpus) as belonging to a particular part of speech based on both its description and its context

Data Journalism

A kind of **journalism** that acknowledges the growing importance of numerical data in the creation and dissemination of information in the **digital age**

Data Backup

The process of creating a copy of the data and storing elsewhere that we use for recovery in case our original data is lost or corrupted

Acquiescence bias:

The tendency of survey participants to agree with research findings without actually reflecting their own views or the topic being studied

Adaptive sampling design:

A sampling strategy where the experiment is being modified as it is being conducted using the observations (or collecting data)

Applet

A small software program that is intended not to be run on its own – but rather to be embedded inside another larger application program

Administrative databases:

Huge databases of information gathered in health care for a variety of objectives. Hospitals, healthcare institutions, and health insurance firms are the main institutions that keep such databases

Authentication

The process of recognizing a user's identity

Admissibility:

A fairly broad idea that can be used in any statistical inference technique

Business continuity plan

A written document that outlines how a business will operate during an emergency

Aliasing:

A scenario when several symbolic names within the program can be used to access the same data location in memory

- Risk assessment (Analysis)
- Design strategies and tactics for recovery
- Implementation
- Testing and acceptance
- Maintenance

Audit trail

A software program that logs database updates

Content management system

A **software application** that allows users to create, manage and modify content on a website − without having to code it from scratch − or know how to code at all

Desktop as a Service (DaaS)

Delivering virtual desktops to end users over the Internet

Dynamic Host Configuration Protocol **(DHCP)**

A **network protocol** that dynamically assign the IP address to each host on the network so that they can communicate efficiently

Flash memory

A type of memory that retains information even after power is turned off

Ascertainment bias:

Bias is when some members of the target group are less likely to be included in the results than others when data is collected **(or surveyed, filtered, or documented)** for a study or analysis

Behrens-Fisher problem:

The challenge of comparing the means of two normal distributions with different variances to see if they are equal

Canonical correlation analysis:

A technique for analyzing the relationship between two groups of variables by identifying linear functions of one set of variables that most strongly correlate with linear functions of the variables in the other set

Catastrophe theory:

A hypothesis explaining the rapid, discontinuous effects that minor, continuous changes in independent variables can have on dependent variables

Central Limit Theorem:

A statistical concept that claims that when a large sample size has a finite variance, the samples will have a normal distribution and their means will be roughly equal to the population's mean

System call:

A request from computer software to an operating system's kernel

Library call

A request to use a function provided by a programming library

Library

A package of prewritten code that a programmer can call when writing code so that the programmer doesn't have to write it

Library

Pre-compiled object code which is used in source code file to provide functionality for a **pre-defined function** or **user-defined functions**

Static Library

Called during the compilation of the program

Dynamic Library

Called during run-time and not during the compilation of the program

Bagplot:

A method for finding outliers in bivariate data

Box-and-whisker plot:

A visual way for highlighting the key characteristics of a group of observations

Cluster sampling:

A **probability sampling technique** where a population is divided into clusters and some of these clusters are randomly chosen as the sample

Chernoff faces:

A method of data visualization where multivariate data is presented in the form of a human face

Coarse data:

A word occasionally used when information is neither fully missing nor entirely present

Ceiling effect:

The point at which a dependent variable is no longer affected by an independent variable

IP address:

A number assigned to any hardware device that is connected to the Internet

Used to identify hardware devices on a network

```
#include<iostream>
using namespace std;
int addition (int x, int y) {
int z;
z = x + y;
return z;
}

int main() {
int a;
a = addition(6,2);
cout << "The result is: " << a;
return 0;
}
```

int x and int y are **parameters** − while 6 and 2 are the arguments passed to the **function** to add the values (6 and 2) and return the result (i.e., 6 + 2 i.e., 8) to the statement:

```
cout << "The result is: " << a;
```

to make provision to display the output:

The result is: 8

IDE

(Integrated Development Environment)

A software tool to help you write software

Eclipse and **NetBeans**

Wi-Fi

A wireless method of sending high speed information over short distances using radio waves

Bandwidth:

The maximum amount of information transmitted over an internet connection in a given amount of time

- **Compute Intensive** – A single application program requiring a large amount of computation.
- **Memory Intensive** – A single application program requiring a large amount of memory.
- **Data Intensive** – A single application program operating on a large data set.

Interpreter

(Compiler + runtime environment)

that translates and executes a program in one step

Packet:

A unit of information into which data is broken up for transmission in a network

Ransomware:

Malware designed to deny a user access to files on their computer

Functional abstraction:

The details of how a function is implemented are hidden from the consumer of the function

```
The user is only able view basic functionalities
    where the internal details of the basic
  implementation of functionalities are hidden
```

Windows:

The proprietary operating system distributed by Microsoft.

Architectural design

The process of defining a collection of **hardware and software** components and their interfaces to establish the framework for the development of a **computer system**

Bootstrap:

A free and open source front-end framework for designing websites and web applications

Debugger:

A tool that a programmers use to find mistakes in their code

Time-sharing:

The process of sharing a computing resource among many users by means of multiprogramming and multitasking at the same time

Allows **multiple users** to use a computer system at the same time

- **Code** → Computer instructions
- **Coding** → programming
- **Coder** → programmer

Multitasking:

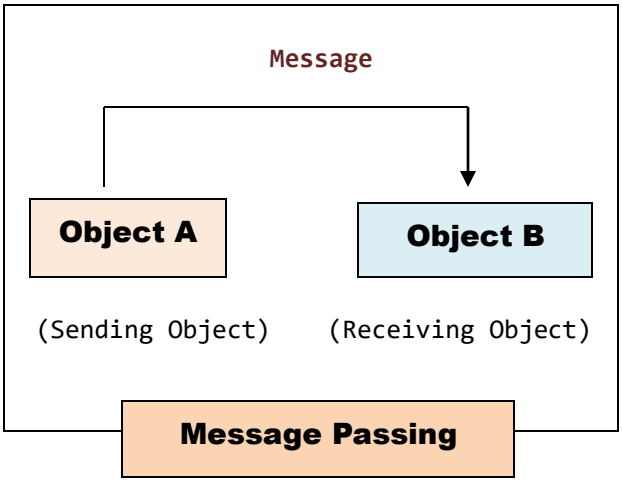
The concurrent execution of multiple tasks or processes over a certain period of time

Allows **multiple tasks** or **processes** to use a computer system at the same time

Scalability

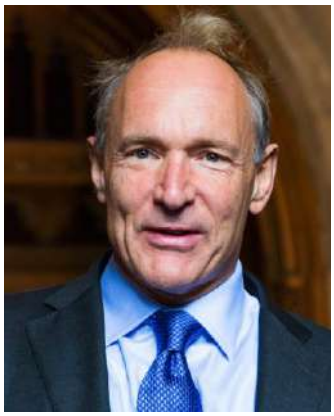
How well a hardware system performs when the number of users is increased

How well a database withstands growing numbers of queries

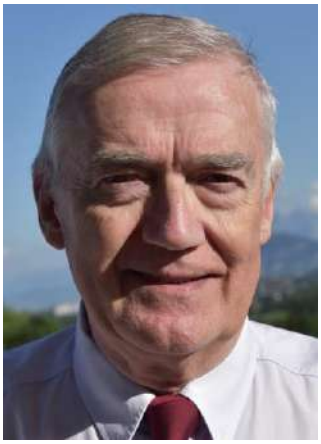


HTML

(HyperText Markup Language)



Tim Berners-Lee



Robert Cailliau

The majority of Web Pages are written in HTML. Making web pages and online applications functional requires the use of HTML. Beginning in 1989, **Tim Berners-Lee**, **Robert Cailliau**, and others developed HTML for the first time. **Hyper Text Markup Language** is what it stands for. Only HTML can be used to construct static web pages. We will concentrate on **CSS**, the code used to create visually pleasing web pages, in a subsequent chapter. We'll concentrate on teaching you how to construct right now rather than how to design.

HTML Documents

- All HTML documents must start with a document type declaration: **`<!DOCTYPE html>.`**
- The HTML document itself begins with **`<html>`** and ends with **`</html>.`**
- The visible part of the HTML document is between **`<body>`** and **`</body>`**.

Example

```
<!DOCTYPE html>
```

Describes that this document is an **HTML5** document

```
<html> <!-- The root element of an HTML Document -->
<head>  <!-- HTML tag that hold meta information about the HTML Document -->
  <title> Page Title </title>
</head>

<body>  <!-- Defines the document's body that hold the content of HTML -->

<h1> HTML </h1> <!-- HTML heading tag -->
<p> Hyper Text Markup Language.</p> <!-- HTML paragraph tag -->

</body>
</html>
```

Specifies a title for the HTML Document that will appear in the title bar or tab of the web browser

HTML Headings

- HTML headings are defined with the **<h1>** to **<h6>** tags.
- **<h1>** defines the **largest heading tag**. **<h6>** defines the smallest one.

Example

```
<!DOCTYPE html>
<html>
<body>

<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
</body>
</html>
```

 # Output:

This is heading 1

This is heading 2

This is heading 3

This is heading 4

This is heading 5

This is heading 6

Used to display heading content

```
<h1 style="font-size: 40px;"> Heading no 1 </h1>
```

Using the **CSS** font-size property of the style attribute, we can alter the size of a heading.

HTML Paragraphs

- HTML paragraphs are defined with the **<p>** tag

Example

```
<!DOCTYPE html>
<html>
<!-- Body Section content -->
<body>
<p>This is a paragraph. </p>
<p>This is another paragraph. </p>
</body>
</html>
```

 # Output:

This is a paragraph.

This is another paragraph.

Used to display paragraph content

A **paragraph** always begins on a new line. Each paragraph begins with the ** ** tag and ends with the ** ** tag

HTML Links

- HTML links are defined with the **<a>** tag.

Example

```
<!DOCTYPE html>
<html>
<body>
<a href="https://www.google.com" target="_blank"> This is a link </a>
</body>
</html>
```

This example demonstrates how to create a link to **google.com**

The HTML <a> tag defines a hyperlink that connects one page to another page

HTML Images

- HTML images are defined with the **<img>** tag.
- The source file **(src)**, alternative text **(alt)**, width, and height are provided as attributes.

Example

```
<!DOCTYPE html>

<html>

<body>

<img src="Alan-Turing.jpg" alt="Alan Turing" width="100" height="100">

</body>

</html>
```

The HTML <img> tag is used to display image on the web page

- **src** - Specifies the path to the image
- **alt** - Describe the image in words

HTML Buttons

- HTML buttons are defined with the **<button>** tag.

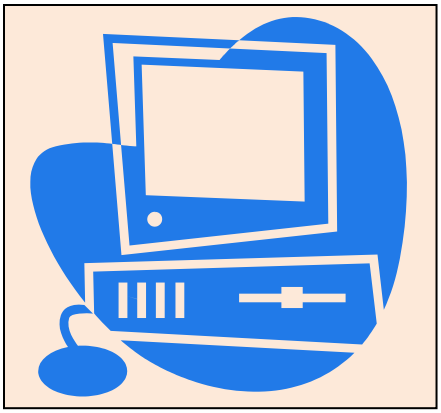
Example

```
<!DOCTYPE html>

<html>

<body>

<button> Click me </button>  <!-- The <button> tag defines a clickable button -->

</body>

</html>
```

HTML Lists

There are 3 different types of HTML lists:

- Ordered HTML List
- Unordered HTML List
- HTML Description List



Example

```
<!DOCTYPE html>
<html>
<body>

<ul>
  <li> Apple </li>
  <li> Orange </li>
  <li> Mango </li>
</ul>


<ol>
  <li> Apple </li>
  <li> Orange </li>
  <li> Mango </li>
</ol>

<dl>
  <dt> Plum </dt>
  <dd> Stone Fruit </dd>
  <dt> Orange </dt>
  <dd> Citrus Fruit </dd>
</dl>

</body>
</html>
```

Unordered HTML List

(Bullet list)

Ordered HTML List

(Numbered list)

HTML Description List

(Definition List)

HTML Tag	Description
	Defines an unordered list
	Defines an ordered list
	Defines a list item
<dl>	Defines a description list
<dt>	Defines a term in a description list
<dd>	Describes the term in a description list

 # Output:

- Apple
- Orange
- Mango

1. Apple
2. Orange
3. Mango

Plum

Stone Fruit

Orange

Citrus Fruit

Empty HTML Elements

- HTML elements with no content are called empty elements.
- **
** is an empty element without a closing tag (the **
** tag defines a line break)

Example

```
<!DOCTYPE html>
<html>
<body>
<p> This is a <br> paragraph with a line break. </p>
</body>
</html>
```

 # Output:

This is a

paragraph with a line break.

HTML Attributes

The href Attribute

- HTML links are defined with the **<a>** tag. The link address is specified in the **href** attribute.

Example

```
<!DOCTYPE html>
<html>
<body>
<a href="https://www.google.com"> This is a link </a>
</body>
</html>
```

The URL of the page to which the link leads is specified by the "href" property

The src Attribute

- HTML images are defined with the **<img>** tag.
- The filename of the image source is specified in the **src** attribute.

Example

```
<!DOCTYPE html>
<html>
<body>
<img src="https://www.git.com/images/Alan-Turing.jpg">

<img src="Alan-Turing.jpg">
</body>
</html>
```

Absolute URL

A link to an external image hosted on another website

Relative URL

A link to an image that is hosted within the website

The width and height Attributes

- HTML images also have width and height attributes, which specifies the width and height of the image

Example

```
<!DOCTYPE html>
<html>
<body>
<img src="Alan-Turing.jpg" width="100" height="100">
</body>
</html>
```

The alt Attribute

If an image cannot be displayed for some reason, the **"alt"** attribute for the **<img>** tag provides an alternative text for the image. This can be the result of a **poor network connection** or an issue with the src attribute

Example

Output if we provide a link to an image that does not exist.


```
<!DOCTYPE html>
<html>
<body>
<img src="Alan-Turing.jpg" alt="Alan Turing">
</body>
</html>
```

The style Attribute

- The style attribute is used to add styles to an element, like color, font, size etc.

Example

```
<!DOCTYPE html>
<html>
<body>
<p style="color:red"> Alan Mathison Turing. </p>
</body>
</html>
```

 # Output:

Alan Mathison Turing.

```
<!DOCTYPE html>
<html lang="en">
<body>
...
</body>
</html>
```

Specifies the language of the Web page as "English"

The title Attribute

- Here, a title attribute is added to the **<p>** element. The value of the title attribute will be displayed as a **tooltip** when we mouse over the paragraph.

Example

```
<!DOCTYPE html>
<html>
<body>
<p title="Albert Einstein"> Special theory of Relativity. </p>
</body>
</html>
```

When the mouse moves over the <p> element then it displays the text "Albert Einstein"

```
<!DOCTYPE html>
<html lang="en-IN">
<body>
...
</body>
</html>
```

Specifies the "English" as the language and "India" as the country

HTML Horizontal Rules

- The **<hr>** tag defines a thematic break in an HTML page, and is most often displayed as a horizontal rule.
- The **<hr>** element is used to separate content (or define a change) in an HTML page.

Example

```
<!DOCTYPE html>
<html>
<body>
<p> Albert </p>
<hr>
<h1> Einstein </h1>
</body>
</html>
```

The **<hr>** tag is an empty tag and it does not require an end tag

 # Output:

Albert

Einstein

```
<!DOCTYPE html>
<html>
<body>
<p> Albert </p>
<hr>
<h1> Einstein </h1>
<hr size="30">
</body>
</html>
```

 # Output:

Albert

Einstein

Specify the height of the horizontal rule → **"30 pixels"**

The HTML <pre> Element

- The HTML **<pre>** element defines preformatted text.
- The text inside a **<pre>** element is displayed in a fixed-width font (usually `Courier`), and it preserves both spaces and line breaks, and without word wrap.

Example

```
<!DOCTYPE html>
<html>
<body>


<p> <!-- HTML "p" tag starts here -->
  Albert Einstein was a German-born theoretical
  physicist    who developed the theory of relativity.
</p> <!-- HTML "p" tag ends here -->


<pre> <!-- HTML "pre" tag starts here -->
  Albert Einstein was a German-born theoretical
  physicist   who developed the theory of relativity.
</pre> <!-- HTML "pre" tag ends here -->


</body>
</html>
```

Line Breaks in HTML without the <pre> Tag

Line Breaks in HTML with the <pre> Tag

Output:

Albert Einstein was a German-born theoretical physicist who developed the theory of relativity.

```
Albert Einstein was a German-born theoretical
physicist    who developed the theory of relativity.
```

HTML Styles

Background Color

- The CSS background-color property defines the background color for an **HTML** element.

Example

This **example** sets the background color for a web page to "orange"

```
<!DOCTYPE html>
<html>
<body style="background-color: orange;">
<h1> HTML </h1>
<p> HyperText Markup Language. </p>
</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

Text Color

- The CSS color property defines the text color for an **HTML** element.

Example

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color: green;"> HTML </h1>
<p style="color: red;"> HyperText Markup Language.</p>
</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

Fonts

- The CSS font-family property defines the font to be used for an **HTML** element.

Example

```
<!DOCTYPE html>
<html>
<body>
<h1 style="font-family: verdana;"> HTML </h1>
<p style="font-family: courier;"> HyperText Markup Language. </p>
</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

Text Size

- The CSS font-size property defines the text size for an **HTML** element.

Example

```
<!DOCTYPE html>
<html>
<body>
<p style="font-size:300%;"> HTML </p>
<p style="font-size:160%;"> HyperText Markup Language. </p>
</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

Text Alignment

- The CSS text-align property defines the horizontal text alignment for an **HTML** element.

Example

```
<!DOCTYPE html>
<html>
<body>

<p style="text-align:left;">HTML</p>
<p style="text-align:center;">HTML</p>
<p style="text-align:right;">HTML</p>
</body>
</html>
```

 # Output:

HTML

HTML

HTML

HTML Text Formatting

```
<!DOCTYPE html>
<html>
<body>
<p> <b> Albert </b> </p> <! -- Defines bold text --->
<p> <strong> Albert </strong> </p> <! -- Defines important text --->
<p> <i> Albert </i> </p> <! -- Defines italic text --->
<p> <em> Albert </em> </p> <! -- Defines Emphasized text --->
<p> <mark> Albert </mark> </p> <! -- Defines Marked or highlighted text --->
<p> <del> Albert </del> </p> <! -- Defines Deleted text --->
</body>
</html>
```

 # Output:

Albert

Albert

Albert

Albert

Albert

```
<!DOCTYPE html>
<html>
<body>

<p> Alan Mathison Turing </p>
<p> <small> Alan Mathison Turing </small> </p> <! -- Defines smaller text --->

</body>
</html>
```

 # Output:

Alan Mathison Turing

Alan Mathison Turing

```
<!DOCTYPE html>
<html>
<body>

<p> E = mc <sup>2</sup> </p> <! -- Defines superscripted text --->
<p> O <sub>2</sub> </p> <! -- Defines subscripted text --->

<p> Alan <ins> Mathison </ins> Turing </p> <! -- Defines inserted text --->

</body>
</html>
```

In most cases, **web browsers** will highlight **inserted text**

 # Output:

E = mc ^{2}

O $_{2}$

Alan Mathison Turing

HTML Quotation and Citation Elements

```
<!DOCTYPE html>
<html>
<body>
<!--blockquote tag starts here -->
<blockquote cite= "https://en.wikipedia.org/wiki/Albert_Einstein">

<p>
Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time. Einstein is best known for developing the theory of relativity, but he also made important contributions to the development of the theory of quantum mechanics.
</p>

</blockquote> <!--blockquote tag ends here -->
 </body>
</html>
```

The HTML **<blockquote>** tag defines a long quotation

blockquote tag with attribute **"cite"**

which is used to identify the source of the quotation

 # Output:

Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time. Einstein is best known for developing the theory of relativity, but he also made important contributions to the development of the theory of quantum mechanics.

- The element content is the text that appears between the start and end tags.
- **HTML** tags are case-insensitive; <p> and <P> have the same meanings.

```
<!DOCTYPE html>
<html>
<body>

<p>Albert Einstein was a <q> German-born theoretical physicist. </q> </p>

</body>
</html>
```

The HTML <q> tag defines a short quotation

 # Output:

Albert Einstein was a “German-born theoretical physicist.”

```
<!DOCTYPE html>
<html>
<body>

<p> The <abbr title="United Nations">UN</abbr> was founded in 1945. </p>

</body>

</html>
```

The HTML <abbr> tag defines an abbreviation

abbr tag with attribute **"title"**

which is used to display the description for the **abbreviation** when you hover your cursor over the element

 # Output:

The UN was founded in 1945.

```
<!DOCTYPE html>
<html>
<body>

<address>
Written by Albert.<br>
Visit us at:<br>
Science.com<br>
Box 564, London<br>
UK
</address>

</body>
</html>
```

The HTML <address> tag defines the contact information for the author of a HTML document

 # Output:

Written by Albert.
Visit us at:
Science.com
Box 564, London
UK

```
<!DOCTYPE html>
<html>
<body>

<p> <cite> Cosmos </cite> is a popular science book. </p>

<!-- This line will be printed from right to left -->
<p> <bdo dir="rtl"> Cosmos is a popular science book. </bdo> </p>

</body>
</html>
```

The HTML <cite> tag defines the title of a work (e.g. movie, book, literature, sculpture, painting or drawing)

 # Output:

Cosmos is a popular science book.

The HTML <bdo> tag defines the text direction

 # Output:

.koob ecneics ralupop a si somsoC

HTML Comments

- Comment tags are used to insert comments in the **HTML** source code.

HTML tags typically come in pairs like ** ** and ** **. The first tag in a pair is the **start tag**, the second tag is the **end tag**

Example

```
<!DOCTYPE html>
<html>
<body>

<!-- This is a comment -->

<h1> HTML </h1>

<!-- <p> HyperText Markup Language. </p> -->

</body>
</html>
```

Although **HTML comments** are not visible in the **web browser**, they can aid in the explanation of your **HTML** source code.

Comments may be employed to hide content.

HTML content between the <!-- and the --> will be concealed from the visibility

 # Output:

HTML

Because each line of **HTML** code may be individually commented out to check for flaws, comments are excellent for debugging **HTML**.

Comments hide a part of a paragraph

```
<p> Albert <!-- Einstein --> was a German-born theoretical physicist. </p>
```

 # Output:

Albert was a German-born theoretical physicist.

```html
<!DOCTYPE html>
<html>
<body>

<!-- A <span> element is used to color a part of a paragraph -->

<p> I have 2 <span style="color: red"> red </span> shirts. </p>

</body>
</html>
```

 # Output:

I have 2 red shirts.

```html
<!DOCTYPE html>
<html>
<body>

<p> I have 2 <span style="color: red; font-weight: bold"> red </span> shirts. </p>

<p> I have 2 <span style="color: red; font-family: 'Courier New'"> red </span> shirts.
</p>

<p> I have 2 <span style="color: red; font-size: 30px"> red </span> shirts. </p>

<p> I have 2 <span style="color: red; font-style: italic"> red </span> shirts. </p>

</body>
</html>
```

In this example, we used a span tag to modify the "red" text's color, size, and font style.

 # Output:

I have 2 **red** shirts.

I have 2 `red` shirts.

I have 2 red shirts.

I have 2 *red* shirts.

```
<span style="background-color: green; color: yellow"> red </span>
```

red

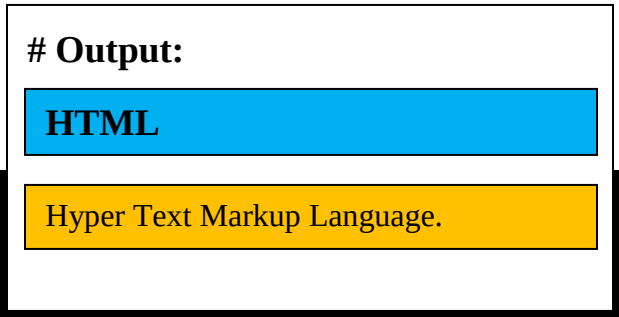
HTML Colors

- HTML colors are specified using predefined color names, or `RGB`, `HEX`, `HSL`, `RGBA`, `HSLA` values.

Background Color

Example

 # Output:




```
<!DOCTYPE html>
<html>
<body>
<h1 style="background-color: DodgerBlue;"> HTML </h1>
<p style="background-color: Orange;"> Hyper Text Markup Language. </p>
</body>
</html>
```

Text Color

Example

 # Output:

Albert Einstein

Elsa Einstein

Elsa Einstein...

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color: Red;"> Albert Einstein </h1>
<p style="color: Green;"> Elsa Einstein </p>
<p style="color: Blue;"> Elsa Einstein... </p>
</body>
</html>
```

Border Color

Example

```
<!DOCTYPE html>
<html>
<body>
<h1 style="border:2px solid Red;"> Albert </h1>
<h1 style="border:4px solid Blue;"> Elsa </h1>
<h1 style="border:6px solid Green;"> John </h1>
</body>
</html>
```

 # Output:

Albert

Elsa

John

Color Values

- In HTML, colors can also be specified using `RGB` values, `HEX` values, `HSL` values, `RGBA` values, and `HSLA` values:

Example

```
<!DOCTYPE html>
<html>
<body>
<h1 style="background-color: rgb(255, 99, 71);"> rgb(255, 99, 71) </h1>
<h1 style="background-color: #ff6347;"> #ff6347 </h1>
<h1 style="background-color: hsl(9, 100%, 64%);"> hsl(9, 100%, 64%) </h1>
<h1 style="background-color: rgba(255, 99, 71, 0.5);"> rgba(255, 99, 71, 0.5) </h1>
<h1 style="background-color: hsla(9, 100%, 64%, 0.5);"> hsla(9, 100%, 64%, 0.5) </h1>
</body>
</html>
```

Color	HTML Code
Alice Blue	#F0F8FF
Antique White	#FAEBD7
Aqua	#00FFFF
Aquamarine	#7FFFD4
Azure	#F0FFFF
Beige	#F5F5DC
Bisque	#FFE4C4
Black	#000000
Blanched Almond	#FFEBCD
Blue	#0000FF
Blue Violet	#8A2BE2
Brown	#A52A2A
Burly Wood	#DEB887
Cadet Blue	#5F9EA0
Chartreuse	#7FFF00
Chocolate	#D2691E
Coral	#FF7F50
Cornflower Blue	#6495ED

JavaScript Program:

```
<strong>
<script>
document.write("Hello World!");
</script>
</strong>
```

Output:

Hello World!

Boldfaced by the web browser

Dynamic HTML

(REAL Interactivity)

Corn silk	#FFF8DC
Crimson	#DC143C
Cyan	#00FFFF
Dark Blue	#00008B
Dark Cyan	#008B8B
Dark Goldenrod	#B8860B
Dark Gray	#A9A9A9
Dark Grey	#A9A9A9
Dark Green	#006400
Dark Khaki	#BDB76B
Dark Magenta	#8B008B
Dark Olive Green	#556B2F
Dark orange	#FF8C00
Dark Orchid	#9932CC
Dark Red	#8B0000
Dark Salmon	#E9967A
Dark Sea Green	#8FBC8F
Dark Slate Blue	#483D8B
Dark Slate Gray	#2F4F4F

Synchronous

- Until the current line of code completes, the following one cannot run.
- The current function must be executed before the next one can be executed.

The string "3" is converted to the number 3.

```
<script>
console.log(parseInt("3") + parseInt("2"));

// Output: 5
</script>
```

The string "2" is converted to the number 2.

```
<body onload="alert('Hi');">
```

Execute a JavaScript code immediately after a web page has been loaded

The **onload event** is triggered when the web page finishes loading in the web browser.

Beige	#F5F5DC
Dark Slate Grey	#2F4F4F
Dark Turquoise	#00CED1
Dark Violet	#9400D3
Deep Pink	#FF1493
Deep Sky Blue	#00BFFF
Dim Gray	#696969
Dim Grey	#696969
Dodger Blue	#1E90FF
Fire Brick	#B22222
Floral White	#FFFAF0
Forest Green	#228B22
Fuchsia	#FF00FF
Gainsboro	#DCDCDC
Ghost White	#F8F8FF
Gold	#FFD700
Goldenrod	#DAA520
Gray	#808080
Grey	#808080

```
<script>
/* Display a text "Hi" once after 4 second
 using setTimeout method
*/
function myfunc() {
    console.log('Hi');
}
setTimeout(myfunc, 4000);
</script>
```

The **setTimeout()** function creates a timer that waits 4 second.

When the timer expires, **setTimeout() method** calls the **myfunc()** function

The timer delay is 4000 milliseconds, which is 4 second

4 second delay

Hi

Green	#008000
Green Yellow	#ADFF2F
Honeydew	#F0FFF0
Hot Pink	#FF69B4
Indian Red	#CD5C5C
Indigo	#4B0082
Ivory	#FFFFF0
Khaki	#F0E68C
Lavender	#E6E6FA
Lavender Blush	#FFF0F5
Lawn Green	#7CFC00
Lemon Chiffon	#FFFACD
Light Blue	#ADD8E6
Light Coral	#F08080
Light Cyan	#E0FFFF
Light Golden Rod Yellow	#FAFAD2
Light Gray	#D3D3D3
Light Grey	#D3D3D3
Light Green	#90EE90

JavaScript + Cookie

↓

Persistent data storage on the client!

The type of the script code is always "**text/javascript**" for JavaScript code

```
<script type="text/javascript" src="gtag.js"></script>
```

The name of the file containing the script code, usually ending with **.js**.

The use of **external script** by placing the name of the script file in the **src (source)** attribute of a **<script>** tag

JavaScript + Broken cookie

↓

Big issue!

```
<script>
console.log(Math.round(Math.PI));
// Output: 3
console.log(Math.ceil(Math.PI));
// Output: 4
</script>
```

Light Pink	#FFB6C1
Light Salmon	#FFA07A
Light Sea Green	#20B2AA
Light Sky Blue	#87CEFA
Yellow Green	#9ACD32
Yellow	#FFFF00
White Smoke	#F5F5F5
White	#FFFFFF
Wheat	#F5DEB3
Violet	#EE82EE
Light Slate Gray	#778899
Light Slate Grey	#778899
Light Steel Blue	#B0C4DE
Light yellow	#FFFFE0
Lime	#00FF00
Lime Green	#32CD32
Linen	#FAF0E6
Magenta	#FF00FF
Maroon	#800000

```
<script>

console.log(`50/2 = ${50/2}`);

// Output: 50/2 = 25



console.log(- (15 - 6));

// Output: -9



console.log(Math.min(4, 6) * 50);

// Output: 200



console.log(Math.max(4, 6) * 50);

// Output: 300

</script>



<script>

let x = [11, 12, 13];

x.push(14);

x.push(15);

console.log(x);

// Output: [ 11, 12, 13, 14, 15 ]

</script>
```

Turquoise	#40E0D0
Tomato	#FF6347
Thistle	#D8BFD8
Teal	#008080
Tan	#D2B48C
Medium Aqua Marine	#66CDAA
Steel Blue	#4682B4
Medium Blue	#0000CD
Medium Orchid	#BA55D3
Spring Green	#00FF7F
Snow	#FFFAFA
Medium Purple	#9370D8
Medium Sea Green	#3CB371
Medium Slate Blue	#7B68EE
Medium Spring Green	#00FA9A
Slate Gray	#708090
Slate Grey	#708090
Silver	#C0C0C0
Sienna	#A0522D

```
<script>
if (false != true) {
console.log("Albert");
if (11 < 22) {
console.log("Einstein");
}
}
</script>
```

Output:

Albert

Einstein

```
<script>
console.log(5 * null);
// Output: 0
console.log(false == 0);
// Output: true
console.log(true == 1);
// Output: true
console.log(null == undefined);
// Output: true
</script>
```

Medium Turquoise	#48D1CC
Medium Violet Red	#C71585
Midnight Blue	#191970
Olive	#808000
Mint Cream	#F5FFFA
Misty Rose	#ffe4e1
Moccasin	#FFE4B5
Peru	#CD853F
Plum	#DDA0DD
Navajo White	#FFDEAD
Navy	#000080
Old Lace	#FDF5E6
Olive Drab	#6B8E23
Orange	#FFA500
Orange Red	#FF4500
Orchid	#DA70D6
Pink	#FFC0CB
Salmon	#FA8072
Pale Golden Rod	#EEE8AA

```
<script>

console.log(true ? 1 : 0);

// Output: 1

console.log(false ? 1 : 0);

// Output: 0


if (2 + 2 == 4) console.log("2+2=4");

// Output: 2+2=4


console.log("A", "B", 5);

// Output: A B 5

</script>


<script>

function myfunc(name) {

console.log("Hi " + name);

}

myfunc("Albert");

console.log("How are you?");

</script>

# Output:
```

Hi Albert

How are you?

Slate Blue	#6A5ACD
Sky Blue	#87CEEB
Sea Shell	#FFF5EE
Pale Green	#98FB98
Pale Turquoise	#AFEEEE
Pale Violet Red	#D87093
Sea Green	#2E8B57
Sandy Brown	#F4A460
Papaya Whip	#FFEFD5
Peach Puff	#FFDAB9
Powder Blue	#B0E0E6
Purple	#800080
Red	#FF0000
Rosy Brown	#BC8F8F
Royal Blue	#4169E1
Saddle Brown	#8B4513

```
<script>
/*
This program serves as an example of a break statement. The first value higher than or equal to 21 and divisible by 2 is obtained.
*/

for(var i = 21; ; i = i + 1) {
if (i % 2 == 0) {
console.log(i);
break;
}
}
// Output: 22
</script>
```

```
<script>
function repeat(n, action) {
for (var x = 1; x < n; x++) {
action("Albert Einstein");
}
}
repeat(3, console.log);
</script>
```

This program demonstrates a function that calls **console.log()** 2 times.

Output:

```
Albert Einstein
Albert Einstein
```

```
<script>



/*

An object can't have new attributes added to it

when it is frozen using the Object.freeze() method.

This method prohibits the alteration of already-existing

characteristics, values, and properties of the object.

*/



<!-- Freezing the object using object.freeze() method -->

var x = Object.freeze({value: 15});

<!-- Updating the value of the frozen object -->

x.value = 100;

<!-- Displaying the value of the frozen object -->

console.log(x.value); // Output: 15



</script>
```

HTML CSS

Inline CSS

- An **inline CSS** is used to apply a unique style to a single **HTML** element.
- An **inline CSS** uses the style attribute of an **HTML** element.
- This example sets the text color of the **<h1>** element to blue.

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color: blue;"> HTML </h1>
</body>
</html>
```

 # Output:

HTML

Internal CSS

- An internal **CSS** is used to define a style for a single **HTML** page.
- An internal **CSS** is defined in the **<head>** section of an **HTML** page, within a **<style>** element.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {color: blue;}
p  {color: red;}
</style>
</head>
<body>
<h1> HTML </h1>
<p> HyperText Markup Language. </p>
</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

External CSS

- An external style sheet is used to define the style for many **HTML** pages.
- **With an external style sheet, we can change the look of an entire web site, by changing one file!**
- To use an external style sheet, add a link to it in the **<head>** section of the **HTML** page.

Example

```
body {
  background-color: orange;
}
h1 {
  color: blue;
}
p {
  color: green;
}
```

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="styles.css">
</head>
<body>

<h1> HTML </h1>
<p> HyperText Markup Language. </p>

</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

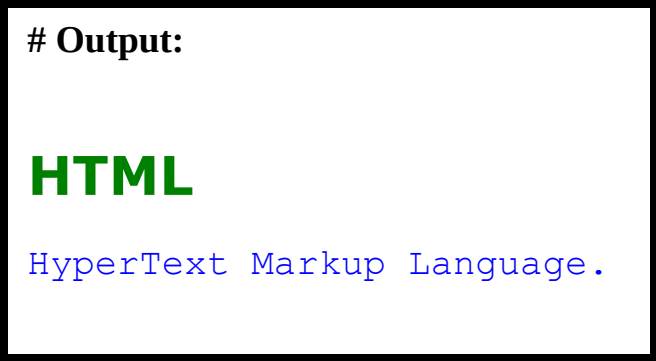
CSS Fonts

- The **CSS** color property defines the text color to be used.
- The **CSS** font-family property defines the font to be used.
- The **CSS** font-size property defines the text size to be used.

Example

```
<!DOCTYPE html>
<html>
<head>

<style>
h1 {
  color: green;
  font-family: verdana;
  font-size: 200%;
}
p {
  color: blue;
  font-family: courier;
  font-size: 130%;
}
</style>

</head>

<body>

<h1> HTML </h1>
<p> HyperText Markup Language. </p>

</body>
</html>
```

 # Output:



CSS Border

- The CSS border property defines a border around an **HTML** element.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  border: 3px solid green;
}
</style>
</head>
<body>
<h1> HTML </h1>
<p> HyperText Markup Language. </p>
</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

CSS Padding

- The CSS padding property defines a **padding** (space) between the text and the border.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  border: 3px solid green;
  padding: 40px;
}
</style>
</head>
<body>
<h1> HTML </h1>
<p> HyperText Markup Language. </p>
</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

CSS Margin

- The CSS margin property defines a **margin** (space) outside the border.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  border: 4px solid green;
  margin: 50px;
}
</style>
</head>

<body>

<h1> HTML </h1>

<p> HyperText Markup Language. </p>
<p> HyperText Markup Language. </p>

</body>
</html>
```

 # Output:

HTML

HyperText Markup Language.

HyperText Markup Language.

The id Attribute

- To define a specific style for one special element, add an **id** attribute to the element.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#p01 {
  color: blue;
}
</style>
</head>
<body>
<p> Albert Einstein. </p>
<p id="p01"> Elsa Einstein. </p>
</body>
</html>
```

 # Output:

Albert Einstein.

Elsa Einstein.

The class Attribute

- To define a style for special types of elements, add a class attribute to the element:

Example

```
<!DOCTYPE html>
```

```
<html>
<head>
<style>
.p01 {
  color: blue;
}
</style>
</head>
<body>
<p> Albert Einstein. </p>
<p class="p01"> Elsa Einstein. </p>
</body>
</html>
```

 # Output:

Albert Einstein.

Elsa Einstein.

HTML Links - The target Attribute

The target attribute specifies where to open the linked document.

The target attribute can have one of the following values:

- **_blank** → Opens the linked document in a new window
- **_self** → Opens the linked document in the same window
- **_parent** → Opens the linked document in the parent frame
- **_top** → Opens the linked document in the full body of the window

Example

```
<!DOCTYPE html>
<html>
```

```
<body>

<a href="https://www.google.com/" target="_blank"> Visit GOOGLE! </a>

<a href="https://www.amazon.com/" target="_top"> Visit Amazon! </a>

</body>

</html>
```

If **target="_blank"**, the link will open in a new browser window.

HTML Links - Image as a Link

It is common to use images as links:

Example

```
<!DOCTYPE html>

<html>

<body>

<a href="https://en.wikipedia.org/wiki/Alan_Turing" target="_blank">

<img src="Alan-Turing.jpg" alt="Alan Turing" style="width:100px;height:100px;border:0;">

</a>

</body>

</html>
```

Put the <img> tag within the <a> tag to use an image as a link.

Button as a Link

- To use an **HTML** button as a link, we have to add some JavaScript code.
- JavaScript allows us to specify what happens at certain events, such as a click of a button.

Example

```
<!DOCTYPE html>
<html>
<body>
<button onclick="document.location = 'https://www.google.com/'"> GOOGLE </button>
</body>
</html>
```

 # Output:

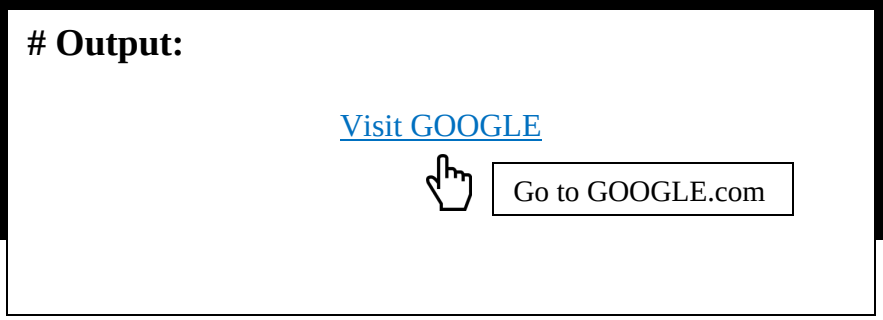
Link Titles

- The **title** attribute specifies extra information about an element. The information is most often shown as a**tooltip** text when the mouse moves over the element.

Example

```
<!DOCTYPE html>
<html>
<body>
<a href="https://www.google.com" title="Go to GOOGLE.com" target="_blank"> Visit  GOOGLE </a>
</body>
</html>
```

 # Output:



HTML JavaScript

- JavaScript makes **HTML** pages more dynamic and interactive.

Example

The use of unique HTML tag properties like "onclick" to incorporate JavaScript code directly within the HTML tag.

```
<!DOCTYPE html>
<html lang="en">
<body>

<button type="button" onclick="alert('How are you?')"> Click Me </button>

</body>
</html>
```

This program will display an alert notification upon clicking the button element.

The HTML <script> Tag

- The **<script>** tag is used to define a client-side script (**JavaScript**).
- The **<script> element** either contains script statements, or it points to an external script file through the "**src**" attribute.
- Common uses for JavaScript are image manipulation, form validation, and dynamic changes of content.
- To select an **HTML** element, JavaScript most often uses the **document.getElementById() method**.
- This JavaScript example writes "Albert Einstein" into an **HTML** element with id="mytxt":

Example

```
<!DOCTYPE html>
```

```
<html>

<body>

<p id="mytxt"></p>

<script>

document.getElementById("mytxt").innerHTML = "Albert Einstein";

</script>

</body>

</html>
```

 # Output:

Albert Einstein

HTML Link Bookmarks

- **HTML** bookmarks are used to allow readers to jump to specific parts of a Web page.
- Bookmarks can be useful if a webpage is very long.

Example

```
<!DOCTYPE html>

<html>

<body>

<p> <a href="#C4"> Jump to Chapter 4 </a> </p>

<p> <a href="#C10"> Jump to Chapter 10 </a> </p>


<h2> Chapter 1 </h2>

<p> This chapter explains absorption </p>


<h2> Chapter 2 </h2>

<p> This chapter explains absorption spectrum </p>
```

The use of "href" attribute to link to the bookmark

```
<h2> Chapter 3 </h2>
<p> This chapter explains activation energy </p>
<h2 id="C4"> Chapter 4 </h2>
<p> This chapter explains adaptive radiation </p>

<h2> Chapter 5 </h2>
<p> This chapter explains aerobiology </p>

<h2> Chapter 6 </h2>
<p> This chapter explains agrobiology </p>

<h2> Chapter 7 </h2>
<p> This chapter explains amino acid </p>

<h2> Chapter 8 </h2>
<p> This chapter explains arachnology </p>

<h2> Chapter 9 </h2>
<p> This chapter explains artificial selection </p>

<h2 id="C10"> Chapter 10 </h2>
<p> This chapter explains asexual reproduction </p>

<h2> Chapter 11 </h2>
<p> This chapter explains autoimmunity </p>

<h2> Chapter 12 </h2>
<p> This chapter explains bacteriophage </p>

<h2> Chapter 13 </h2>
```

The use of "id" attribute to define bookmarks in a page

```
<p> This chapter explains behavioral ecology </p>


<h2> Chapter 14 </h2>

<p> This chapter explains binary fission </p>


<h2> Chapter 15 </h2>

<p> This chapter explains biocatalysis </p>


<h2> Chapter 16 </h2>

<p> This chapter explains biogeography </p>


<h2> Chapter 17 </h2>

<p> This chapter explains biomechanics </p>


<h2> Chapter 18 </h2>

<p> This chapter explains biomedical research </p>


<h2> Chapter 19 </h2>

<p> This chapter explains biotechnology </p>


<h2> Chapter 20 </h2>

<p> This chapter explains carbon fixation </p>


<h2> Chapter 21 </h2>

<p> This chapter explains cell biology </p>


<h2> Chapter 22 </h2>

<p> This chapter explains chemical equilibrium </p>


<h2> Chapter 23 </h2>

<p> This chapter explains chemosynthesis </p>
```

```
</body>

</html>
```

HTML Link Colors

Example

```
<!DOCTYPE html>

<html>

<head>

<style>

a {

text-decoration: none;

color: blue;

}

a:hover {

color: red;

background-color: yellow;

text-decoration: underline;

}

</style>

</head>

<body>


<a href="https://www.google.com" target="_blank"> GOOGLE </a>


</body>

</html>
```

 # Output:

GOOGLE

GOOGLE

(On hover)

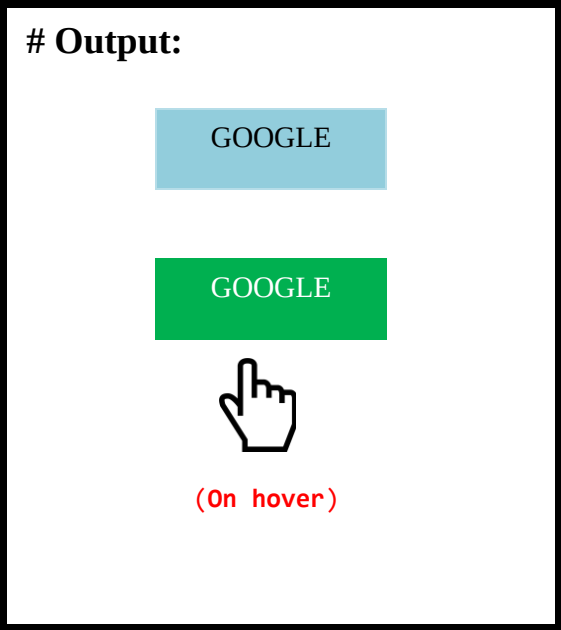
Link Buttons

- A link can also be styled as a button, by using CSS:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
a {
background-color: #89CFF0;
color: black;
padding: 15px 25px;
text-decoration: none;
display: inline-block;
}
a:hover {
background-color: green;
color: white;

}
</style>
</head>

<body>
<a href="https://www.google.com" target="_blank"> GOOGLE </a>
</body>

</html>
```

 # Output:



HTML Symbols

! !	" " "	# #
$ $	% %	& & &	' '	((
))	* * *	+ +	, ,	. .
/ /	: :	; ;	< < <	= =
> > >	? ?	@ @	[[[	\ \
]]]	^ ^	_ _	` ` `	{ { {

|				} } }		¡ ¡	¢ ¢
£ £	¤ ¤	¥ ¥	¦ ¦	§ §
¨ ¨ ¨ ¨ ¨	© © ©	ª ª	« «	¬ ¬
	® ® ® ®	¯ ¯ ‾ ¯	º °	± ± ± ±
² ²	³ ³	´ ´ ´	µ µ	¶ ¶
· · · ·	¸ ¸ ¸	¹ ¹	º º	» »
¼ ¼	½ ½ ½	¾ ¾	¿ ¿	À À

Á Á	Â Â	Ã Ã	Ä Ä	Å Å
Æ Æ	Ç Ç	È È	É É	Ê Ê
Ë Ë	Ì Ì	Í Í	Î Î	Ï Ï
Ð Ð	Ñ Ñ	Ò Ò	Ó Ó	Ô Ô
Õ Õ	Ö Ö	× ×	Ø Ø	Ù Ù
Ú Ú	Û Û	Ü Ü	Ý Ý	Þ Þ
ß ß	à à	á á	â â	ã ã

ä ä	å å	æ æ	ç ç	è è
é é	ê ê	ë ë	ì ì	í í
î î	ï ï	ð ð	ñ ñ	ò ò
ó ó	ô ô	õ õ	ö ö	÷ ÷ ÷
ø ø	ù ù	ú ú	û û	ü ü
ý ý	þ þ	ÿ ÿ	Ā Ā	ā ā
Ă Ă	ă ă	Ą Ą	ą ą	Ć Ć

ć ć	Ĉ Ĉ	ĉ ĉ	Ċ Ċ	ċ ċ
Č Č	č č	Ď Ď	ď ď	Đ Đ
đ đ	Ē Ē	ē ē	Ė Ė	ė ė
Ę Ę	ę ę	Ě Ě	ě ě	Ĝ Ĝ
ĝ ĝ	Ğ Ğ	ğ ğ	Ġ Ġ	ġ ġ
Ģ Ģ	Ĥ Ĥ	ĥ ĥ	Ħ Ħ	ħ ħ
Ĩ Ĩ	ĩ ĩ	Ī Ī	ī ī	Į Į

į į	İ İ	ı ı ı	IJ Ĳ	ij ĳ
Ĵ Ĵ	ĵ ĵ	Ķ Ķ	ķ ķ	ĸ ĸ
Ĺ Ĺ	Í ĺ	Ļ Ļ	ļ ļ	Ľ Ľ
ľ ľ	L· Ŀ	l· ŀ	Ł Ł	ł ł
Ń Ń	ń ń	Ņ Ņ	ņ ņ	Ň Ň
ň ň	ʼn ŉ	Ŋ Ŋ	ŋ ŋ	Ō Ō
ō ō	Ő Ő	ő ő	Œ Œ	œ œ

Ŕ	ŕ	Ŗ	ŗ	Ř
Ŕ	ŕ	Ŗ	ŗ	Ř
ř	Ś	ś	Ŝ	ŝ
ř	Ś	ś	Ŝ	ŝ
Ş	ş	Š	š	Ţ
Ş	ş	Š	š	Ţ
ţ	Ť	ť	Ŧ	ŧ
ţ	Ť	ť	Ŧ	ŧ
Ũ	ũ	Ū	ū	Ŭ
Ũ	ũ	Ū	ū	Ŭ
ŭ	Ů	ů	Ű	ű
ŭ	Ů	ů	Ű	ű
Ų	ų	Ŵ	ŵ	Ŷ
Ų	ų	Ŵ	ŵ	Ŷ

ŷ ŷ	Ÿ Ÿ	Ź Ź	ź ź	Ż Ż
ż ż	Ž Ž	ž ž	ƒ ƒ	Ƶ Ƶ
g´ ǵ	J ȷ	ˆ ˆ	ˇ ˇ ˇ	˘ ˘ ˘
˙ ˙ ˙	˚ ˚	˛ ˛	˜ ˜ ˜	˝ ˝ ˝
‿ ̑	_ _	Α Α	Β Β	Γ Γ
Δ Δ	Ε Ε	Ζ Ζ	Η Η	Θ Θ
Ι Ι	Κ Κ	Λ Λ	Μ Μ	Ν Ν

Ξ Ξ	Ο Ο	Π Π	Ρ Ρ	Σ Σ
Τ Τ	Υ Υ	Φ Φ	Χ Χ	Ψ Ψ
Ω Ω	α α	β β	γ γ	δ δ
ε ϵ ϵ ε	ζ ζ	η η	θ θ	ι ι
κ κ	λ λ	μ μ	ν ν	ξ ξ
ο ο	π π	ρ ρ	ς ς ς ς	σ σ
τ τ	υ υ υ	φ φ ϕ ϕ	χ χ	ψ ψ

ω ω	ϑ ϑ ϑ ϑ	ϒ ϒ ϒ	ϕ ϕ	ϖ ϖ ϖ
Ϝ Ϝ	ϝ ϝ ϝ	ϰ ϰ ϰ	ϱ ϱ ϱ	ϵ ε ϵ
Э ϶ ϶	Ё Ё	Ђ Ђ	Ѓ Ѓ	Є Є
S Ѕ	I І	Ї Ї	J Ј	Љ Љ
Њ Њ	Ћ Ћ	Ќ Ќ	Ў Ў	Џ Џ
А А	Б Б	В В	Г Г	Д Д
Е Е	Ж Ж	З З	И И	Й Й

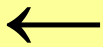
К	Л	М	Н	О
К	Л	М	Н	О
П	Р	С	Т	У
П	Р	С	Т	У
Ф	Х	Ц	Ч	Ш
Ф	Х	Ц	Ч	Ш
Щ	Ъ	Ы	Ь	Э
Щ	Ъ	Ы	Ь	Э
Ю	Я	а	б	в
Ю	Я	а	б	в
г	д	е	ж	з
г	д	е	ж	з
и	й	к	л	м
и	й	к	л	м

н н	о о	п п	р р	с с
т т	у у	ф ф	х х	ц ц
ч ч	ш ш	щ щ	ъ ъ	ы ы
ь ь	э э	ю ю	я я	ё ё
ђ ђ	ѓ ѓ	є є	ѕ ѕ	і і
ї ї	ј ј	љ љ	њ њ	ћ ћ
ќ ќ	ў ў	џ џ		

		‌	‍	‎
‏	‐ ‐ ‐	– –	— —	― ―
‖ ‖ ‖	‘ ‘ ‘	’ ’ ’ ’	‚ ‚ ‚	“ “ “
” ” ” ”	„ „ „	† †	‡ ‡ ‡	• • •
‥ ‥	… … …	‰ ‰	‱ ‱	′ ′
″ ″	‴ ‴	‵ ‵ ‵	‹ ‹	› ›

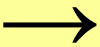
‾ ‾	‸ ⁁	⁃ ⁃	⁄ ⁄	⁏ ⁏
⁗ ⁗			⁡ ⁡	⁢ ⁢
⁣ ⁣	€ €	⃛ ⃛ ⃛	⃜ ⃜	ℂ ℂ ℂ
℅ ℅	ℊ ℊ	ℋ ℋ ℋ ℋ	ℌ ℌ ℌ	ℍ ℍ ℍ
ℎ ℎ	ℏ ℏ ℏ ℏ ℏ	ℐ ℐ ℐ	ℑ ℑ ℑ ℑ ℑ	ℒ ℒ ℒ ℒ
ℓ ℓ	ℕ ℕ ℕ	№ №	℗ ℗	℘ ℘ ℘
ℙ ℙ ℙ	ℚ ℚ ℚ	ℛ ℛ ℛ	ℜ ℜ ℜ ℜ ℜ	ℝ ℝ ℝ

℞ ℞	™ ™ ™	ℤ ℤ ℤ	Ω Ω	℧ ℧
ℨ ℨ ℨ	℩ ℩	Å Å	ℬ ℬ ℬ ℬ	ℭ ℭ ℭ
ℯ ℯ	ℰ ℰ ℰ	ℱ ℱ ℱ	ℳ ℳ ℳ ℳ	ℴ ℴ ℴ ℴ
א ℵ ℵ	ב ℶ	ג ℷ	ד ℸ	ⅅ ⅅ ⅅ
ⅆ ⅆ ⅆ	ⅇ ⅇ ⅇ ⅇ	ⅈ ⅈ ⅈ	⅓ ⅓	⅔ ⅔
⅕ ⅕	⅖ ⅖	⅗ ⅗	⅘ ⅘	⅙ ⅙
⅚ ⅚	⅛ ⅛	⅜ ⅜	⅝ ⅝	⅞ ⅞



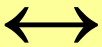
← ←
← ←
←

↑ ↑
↑
↑



→ →
→ →
→

↓ ↓
↓
↓



↔
↔
↔

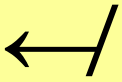
↕ ↕
↕

↖
↖
↖

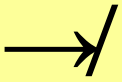
↗
↗
↗

↘ ↘
↘

↙ ↙
↙



↚ ↚



↛ ↛

↝
↝

↞
↞

↟

↠
↠

↡

↢
↢

↣
↣

↤
↤

↥
↥

↦ ↦
↦

↧
↧

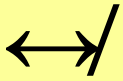
↩
↩

↪
↪

↫
↫

↬
↬

↭
↭



↮
↮

↰ ↰

↱ ↱

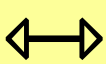
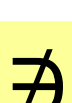
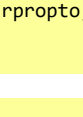
↲

↳

↵

↶
↶

 ↷ ↷	 ↺ ↺	 ↻ ↻	 ↼ ↼ ↼	 ↽ ↽ ↽
 ↾ ↾ ↾	 ↿ ↿ ↿	 ⇀ ⇀ ⇀	 ⇁ ⇁ ⇁	 ⇂ ⇂ ⇂
 ⇃ ⇃ ⇃	 ⇄ ⇄ ⇄	 ⇅ ⇅	 ⇆ ⇆ ⇆	 ⇇ ⇇
 ⇈ ⇈	 ⇉ ⇉	 ⇊ ⇊	 ⇋ ⇋ ⇋	 ⇌ ⇌ ⇌
 ⇍ ⇍	 ⇎ ⇎	 ⇏ ⇏	 ⇐ ⇐ ⇐	 ⇑ ⇑ ⇑
 ⇒ ⇒ ⇒ ⇒	 ⇓ ⇓ ⇓	 ⇔ ⇔ ⇔ ⇔	 ⇕ ⇕ ⇕	 ⇖
 ⇗	 ⇘	 ⇙	 ⇚ ⇚	 ⇛ ⇛

 ⇝	 ⇤ ⇤	 ⇥ ⇥	 ⇵ ⇵	 ⇽
 ⇾	 ⇿	 ∀ ∀	 ∁ ∁	 ∂ ∂
 ∃ ∃	 ∄ ∄ ∄	 ∅ ∅ ∅ ∅	 ∇ ∇	 ∈ ∈ ∈ ∈
 ∉ ∉ ∉	 ∋ ∋ ∋ ∋	 ∌ ∌ ∌	 ∏ ∏	 ∐ ∐
 ∑ ∑	−	 ∓ ∓ ∓	 ∔ ∔	 ∖ ∖ ∖ ∖ ∖
∗	∘ ∘	 √ √	∝ ∝ ∝ ∝ ∝	∞
∟	∠ ∠	∡ ∡	 ∢	∣ ∣ ∣ ∣

∤ ∤ ∤ ∤ ∤	∥ ∥ ∥ ∥ ∥ ∥	∦ ∦ ∦ ∦ ∦ ∦	∧ ∧ ∧	∨ ∨ ∨
∩ ∩	∪ ∪	∫ ∫ ∫	∬ ∬	∭ ∭ ∭
∮ ∮ ∮ ∮	∯ ∯ ∯	∰ ∰	∲ ∱	∳ ∲ ∲
∳ ∳ &CounterClockwiseContourI	∴ ∴ ∴ ∴	∵ ∵ ∵ ∵	∶ ∶	∷ ∷ ∷
∸ ∸ ∸	∺ ∺	≑ ∻	∼ ∼ ∼ ∼ ∼	∽ ∽ ∽
∾ ∾ ∾	∿ ∿	≀ ≀ ≀ ≀	≁ ≁ ≁	≃ ≂ ≂ ≂
≃ ≃ ≃ ≃	≄ ≄ ≄ ≄	≅ ≅ ≅	≆ ≆	≇ ≇ ≇

≈	≉	≊	≋	≌
≈ ≈ ≈ ≈ ≈ ≈	≉ ≉ ≉	≊ ≊	≋	≌ ≌
≍	≎	≏	≐	≑
≍ ≍	≎ ≎ ≎	≏ ≏ ≏	≐ ≐ ≐	≑ ≑
≒	≓	≔	≕	≖
≒ ≒	≓ ≓	≔ ≔ ≔	≕ ≕	≖ ≖
≗	≙	≚	≜	≟
≗ ≗	≙	≚	≜ ≜	≟ ≟
≠	≡	≢	≤	≥
≠ ≠	≡ ≡	≢ ≢	≤ ≤	≥ ≥ ≥
≦	≧	≨	≩	≪
≦ ≦ ≦	≧ ≧ ≧	≨ ≨	≩ ≩	≪ ≪ ≪
≫	≬	≭	≮	≯
≫ ≫ ≫	≬ ≬	≭	≮ ≮ ≮	≯ ≯ ≯

≰ ≰ ≰	 ≱ ≱ ≱	 ≲ ≲ ≲	≳ ≳ ≳	≴ ≴
 ≵ ≵	 ≶ ≶ ≶	 ≷ ≷ ≷	≸ ≸	≹ ≹
≺ ≺ ≺	 ≻ ≻ ≻	 ≼ ≼ ≼	≽ ≽ ≽	≾ ≾ ≾
 ≿ ≿ ≿	 ⊀ ⊀ ⊀	 ⊁ ⊁ ⊁	⊂ ⊂	⊃ ⊃ ⊃
 ⊄	 ⊅	 ⊆ ⊆ ⊆	⊇ ⊇ ⊇	⊈ ⊈ ⊈
 ⊉ ⊉ ⊉	 ⊊ ⊊	 ⊋ ⊋	⊍	⊎ ⊎
 ⊏ ⊏ ⊏	 ⊐ ⊐ ⊐	 ⊑ ⊑ ⊑	⊒ ⊒ ⊒	⊓ ⊓

 ⊔ ⊔	 ⊕ ⊕	 ⊖ ⊖	 ⊗ ⊗	 ⊘
 ⊙ ⊙	 ⊚ ⊚	 ⊛ ⊛	 ⊝ ⊝	 ⊞ ⊞
 ⊟ ⊟	 ⊠ ⊠	 ⊡ ⊡	 ⊢ ⊢	 ⊣ ⊣
 ⊤ ⊤	 ⊥ ⊥ ⊥ ⊥	 ⊧	 ⊨ ⊨	 ⊩
 ⊪	 ⊫	 ⊬	 ⊭	 ⊮
 ⊯	 ⊰	 ⊲ ⊲ ⊲	 ⊳ ⊳ ⊳	 ⊴ ⊴ ⊴
 ⊵ ⊵ ⊵	 ⊶	 ⊷	 ⊸ ⊸	 ⊹

⊺ ⊺	 ⊻	⊽	 ⊾	 ⊿
 ⋀ ⋀ ⋀	 ⋁ ⋁ ⋁	 ⋂ ⋂ ⋂	 ⋃ ⋃ ⋃	 ⋄ ⋄ ⋄
⋅	 ⋆ ⋆	 ⋇ ⋇	 ⋈	 ⋉
 ⋊	 ⋋ ⋋	 ⋌ ⋌	 ⋍ ⋍	 ⋎ ⋎
 ⋏ ⋏	 ⋐ ⋐	 ⋑ ⋑	 ⋒	 ⋓
 ⋔ ⋔	 ⋕	 ⋖ ⋖	 ⋗ ⋗	 ⋘
 ⋙ ⋙	 ⋚ ⋚ ⋚	 ⋛ ⋛ ⋛	 ⋞ ⋞	 ⋟ ⋟

⪵
⋠
⋠

⪶
⋡
⋡

⊈
⋢
⋢

⊉
⋣
⋣

≨
⋦

≩
⋧

⋨
⋨ ⋨

⋩
⋩ ⋩

⋪
⋪
⋪
⋪

⋫
⋫
⋫
⋫

⋬
⋬
⋬
⋬

⋭
⋭
⋭
⋭

⋮
⋮

⋯
⋯

⋰
⋰

⋱
⋱

∈
⋲

⋹
⋳

⋵
⋴

⋵̇
⋵

⋶
⋶

⋷
⋷

⋹̸
⋹

∋
⋺

⋻
⋻

∌
⋼

⋾
⋽

⋽
⋾

⊼
⌅ ⌅

⩞
⌆
⌆

⌈
⌈ ⌈

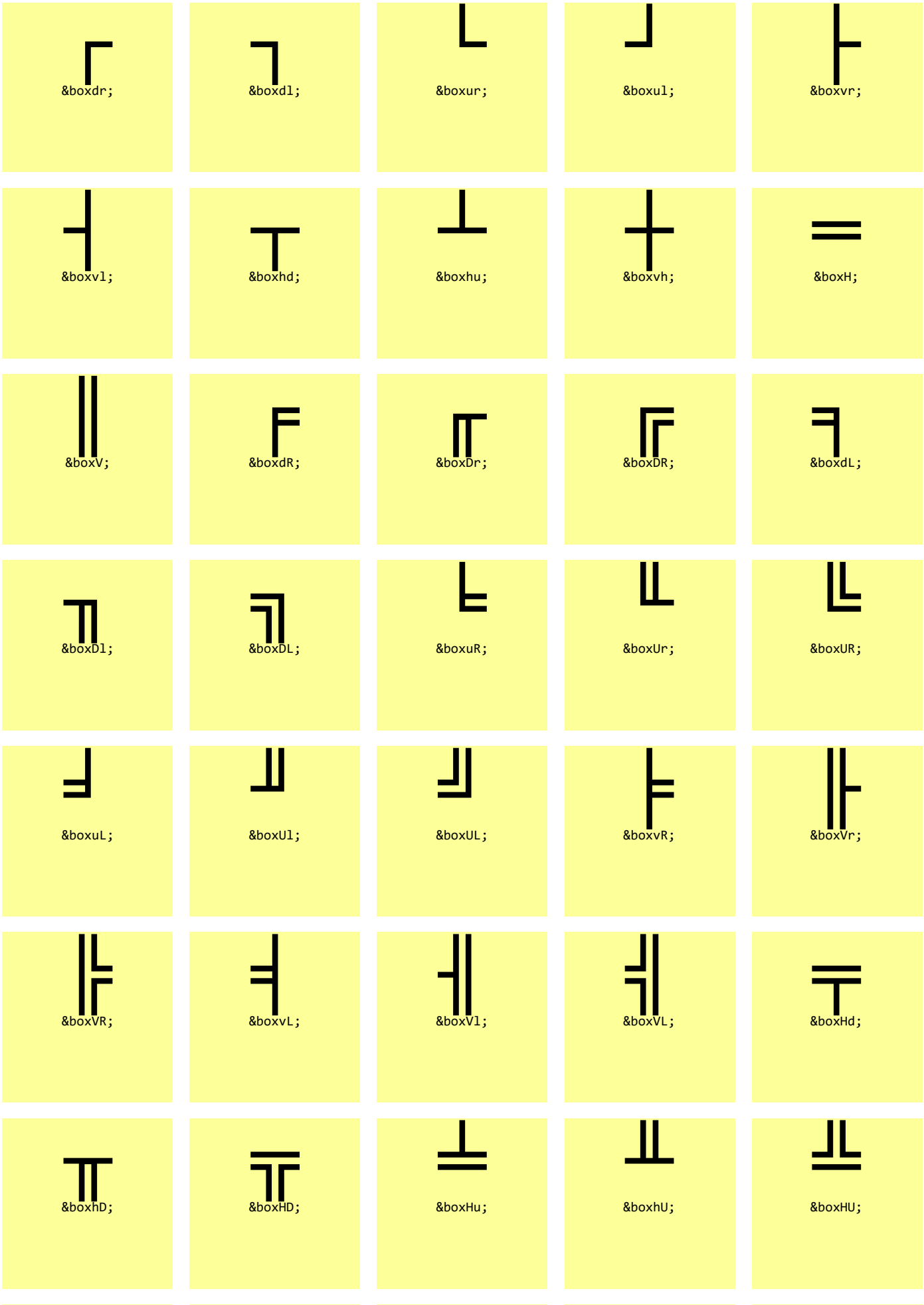
⌉
⌉ ⌉

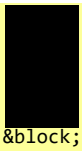
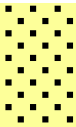
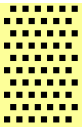
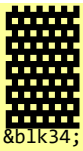
⌊
⌊ ⌊

⌋
⌋ ⌋

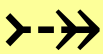
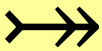
⌐
⌌

 ⌍	 ⌎	 ⌏	 ⌐	 ⌒
 ⌓	 ⌕	 ⌖	 ⌜ ⌜	 ⌝ ⌝
 ⌞ ⌞	 ⌟ ⌟	 ⌢ ⌢	 ⌣ ⌣	 ⌭
 ⌮	 ⌶	 ⌽	 ⌿	 ⍼
 ⎰ ⎰	⎱ ⎱	 ⎴ ⎴	 ⎵ ⎵	 ⎶
 ⏜	 ⏝	 ⏞	 ⏟	 ⏢
 ⏧	 ␣	 Ⓢ Ⓢ	─ ─	 │



 ╪	 ╫	 ╬	 ▀	 ▄
 █	 ░	 ▒	 ▓	 □ □ □
 ▪ ▪ ▪ ▪	 ▫	 ▭	 ▮	 ▱
 △ △	 ▴ ▴	 ▵ ▵	 ▸ ▸	 ▹ ▹
 ▽ ▽	 ▾ ▾	 ▿ ▿	 ◂ ◂	 ◃ ◃
 ◊ ◊	 ○	 ◬	 ◯ ◯	 ◸
 ◹	 ◺	 ◻	 ◼	 ★ ★

☆ ☆	☎ ☎	♀ ♀	♂ ♂	♠ ♠ ♠
♣ ♣ ♣	♥ ♥ ♥	♦ ♦ ♦	♪ ♪	♭ ♭
♮ ♮ ♮	♯ ♯	✓ ✓ ✓	✗ ✗	✠ ✠ ✠
✶ ✶	| ❘	⦇ ❲	⦈ ❳	⟦ ⟦ ⟦
⟧ ⟧ ⟧	⟨ ⟨ ⟨ ⟨	⟩ ⟩ ⟩ ⟩	⟪ ⟪	⟫ ⟫
⟦ ⟬	⟧ ⟭	⟵ ⟵ ⟵ ⟵	⟶ ⟶ ⟶ ⟶	⟷ ⟷ ⟷ ⟷
⟸ ⟸ ⟸ ⟸	⟹ ⟹ ⟹ ⟹	⟺ ⟺ ⟺ &DoubleLongLeftRightArrow	⟼ ⟼ ⟼	⇝ ⟿

 ⤂	 ⤃	 ⤄	 ⤅	 ⤌
 ⤍ ⤍	 ⤎	 ⤏ ⤏	 ⤐ ⤐	 ⤑
 ⤒	 ⤓	 ⤖	 ⤙	 ⤚
 ⤛	 ⤜	 ⤝	 ⤞	 ⤟
 ⤠	 ⤣	 ⤤	 ⤥ ⤥	 ⤦ ⤦
 ⤧	 ⤨ ⤨	 ⤩ ⤩	 ⤪	 ⤳
 ⤵	 ⤶	 ⤷	 ⤸	 ⤹

 ⤼	 ⤽	 ⥅	 ⥈	 ⥉
 ⥊	 ⥋	 ⥎	 ⥏	 ⥐
 ⥑	 ⥒	 ⥓	 ⥔	 ⥕
 ⥖	 ⥗	 ⥘	 ⥙	 ⥚
 ⥛	 ⥜	 ⥝	 ⥞	 ⥟
 ⥠	 ⥡	 ⥢	 ⥣	 ⥤
 ⥥	 ⥦	 ⥧	 ⥨	 ⥩

⥢ ⥪	⥧ ⥫	⥤ ⥬	⥭ ⥭	⥮ ⥮ ⥮
⥯ ⥯ ⥯	⥰ ⥰	⥱ ⥱	⥲ ⥲	⥳ ⥳
⥴ ⥴	⥵ ⥵	⥶ ⥶	⥸ ⥸	⥹ ⥹
⥺ ⥻	⥼ ⥼	⥽ ⥽	⥾ ⥾	⥿ ⥿
⦅ ⦅	⦆ ⦆	⦋ ⦋	⦌ ⦌	⦍ ⦍
⦎ ⦎	⦏ ⦏	⦐ ⦐	⦑ ⦑	⦒ ⦒
⦓ ⦓	⦔ ⦔	⦕ ⦕	⦖ ⦖	⦚ ⦚

⦜

⦝

⦤

⦥

⦦

⦧

⦨

⦩

⦪

⦫

⦬

⦭

⦮

⦯

⦰

⦱

⦲

⦳

⦴

⦵

⦶

⦷

⦹

⦻

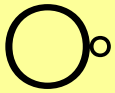
⦼

⦾

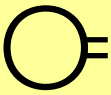
⦿

⧀

⧁



⧂



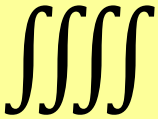
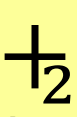
⧃

⧄

⧅

⧉

⧍

 ⧎	 ⧏	 ⧐	 ∽̱	 ⧜
 ⧝	 ⧞	 ⧣	 ⧤	 ⧥
 ⧫ ⧫	 ⧴	 ⧶	 ⨀ ⨀	 ⨁ ⨁
 ⨂ ⨂	 ⨄ ⨄	 ⨆ ⨆	 ⨌ ⨌	 ⨍
 ⨐	 ⨑	 ⨒	 ⨓	 ⨔
 ⨕	 ⨖	 ⨗	 ⨢	 ⨣
 ⨤	 ⨥	 ⨦	 ⨧	 ⨩

 ⨪	 ⨭	 ⨮	 ⨯	 ⨰
 ⨱	 ⨳	 ⨴	 ⨵	 ⨶
 ⨷	 ⨸	 ⨹	 ⨺	 ⨻
 ⨼ ⨼	 ⨿	 ⩀	 ⩂	 ⩃
 ⩄	 ⩅	 ⩆	 ⩇	 ⩈
 ⩉	 ⩊	 ⩋	 ⩌	 ⩍
 ⩐	 ⩓	 ⩔	 ⩕	 ⩖

⩗

⩘

⩚

⩛

⩜

⩝

⩟

⩦

⩪

⩭

⩮

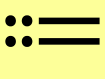
⩯

⩰

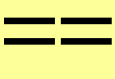
⩱

⩲

⩳



⩴



⩵

⩷ ⩷

⩸

⩹

⩺

⩻

⩼

⩽ ⩽ ⩽

⩾ ⩾ ⩾

⩿

⪀

⪁

⪂

⪃

⪄

⪅ ⪅

⪆ ⪆

⪇ ⪇


⪈ ⪈


⪉ ⪉


⪊ ⪊

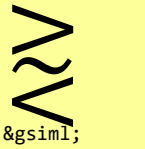

⪋ ⪋


⪌ ⪌


⪍


⪎


⪏


⪐


⪑


⪒


⪓


⪔

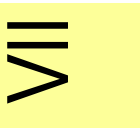

⪕ ⪕

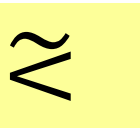

⪖ ⪖


⪗

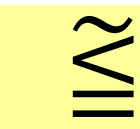

⪘

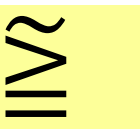

⪙


⪚


⪝

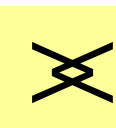

⪞

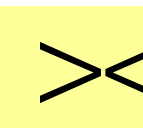

⪟

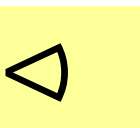

⪠

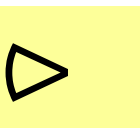

⪡

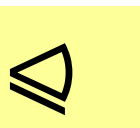

⪢


⪤

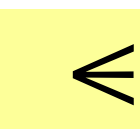

⪥

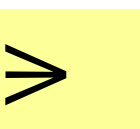

⪦

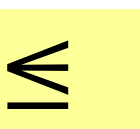

⪧

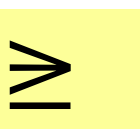

⪨


⪩


⪪


⪫


⪬


⪭

⪮

⪯ ⪯
⪯

⪰ ⪰
⪰

⪳

⪴

⪵ ⪵

⪶ ⪶

⪷ ⪷

⪸ ⪸

⪹ ⪹

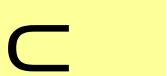
⪺ ⪺

⪻

⪼

⪽

⪾



⪿

⫀

⫁

⫂

⫃

⫄

⫅ ⫅

⫆ ⫆

⫇

⫈

⫋ ⫋

⫌ ⫌

⫏

⫐

⫑

⫒

⫓

⫔

⫕

⫖

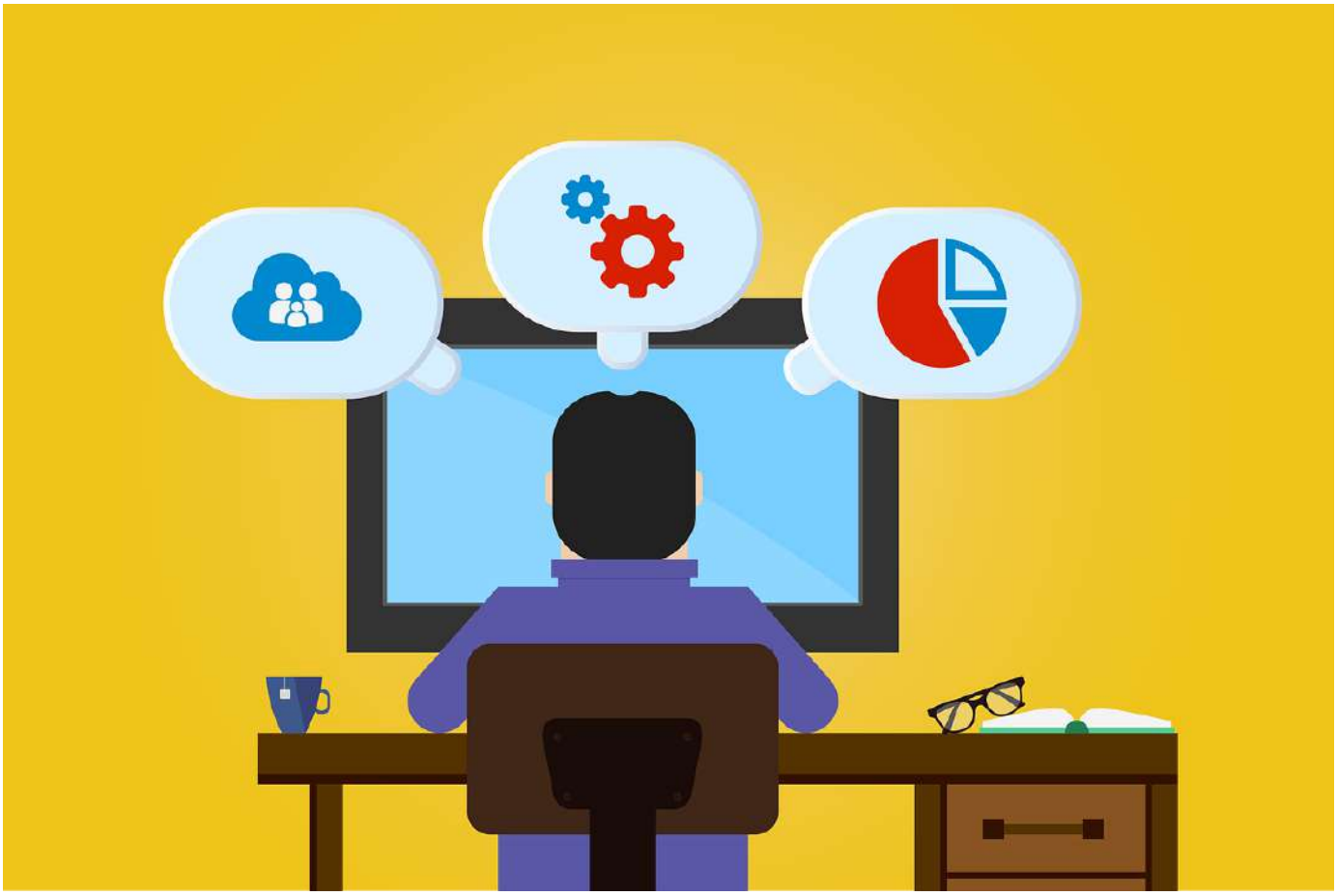
⫗ ⫗	⫘ ⫘	⫙ ⫙	⫚ ⫚	⫛ ⫛
⫤ ⫤ ⫤	⫦ ⫦	⫧ ⫧	⫨ ⫨	⫩ ⫩
⫫ ⫫	⫬ ⫬	⫭ ⫭	⫮ ⫮	⫯ ⫯
⫰ ⫰	⫱ ⫱	⫲ ⫲	⫳ ⫳	⫽ ⫽
ﬀ ﬀ	ﬁ ﬁ	ﬂ ﬂ	ﬃ ﬃ	ﬄ ﬄ
𝒜 𝒜	𝒞 𝒞	𝒟 𝒟	𝒢 𝒢	𝒥 𝒥
𝒦 𝒦	𝒩 𝒩	𝒪 𝒪	𝒫 𝒫	𝒬 𝒬

$\mathcal{S}$ 𝒮	$\mathcal{T}$ 𝒯	$\mathcal{U}$ 𝒰	$\mathcal{V}$ 𝒱	$\mathcal{W}$ 𝒲
$\mathcal{X}$ 𝒳	$\mathcal{Y}$ 𝒴	$\mathcal{Z}$ 𝒵	$\mathcal{a}$ 𝒶	$\mathcal{b}$ 𝒷
$\mathcal{c}$ 𝒸	$\mathcal{d}$ 𝒹	$\mathcal{f}$ 𝒻	$\mathcal{h}$ 𝒽	$\mathcal{i}$ 𝒾
$\mathcal{j}$ 𝒿	$\mathcal{k}$ 𝓀	$\mathcal{l}$ 𝓁	$\mathcal{m}$ 𝓂	$\mathcal{n}$ 𝓃
$\mathcal{p}$ 𝓅	$\mathcal{q}$ 𝓆	$\mathcal{r}$ 𝓇	$\mathcal{s}$ 𝓈	$\mathcal{t}$ 𝓉
$\mathcal{u}$ 𝓊	$\mathcal{v}$ 𝓋	$\mathcal{w}$ 𝓌	$\mathcal{x}$ 𝓍	$\mathcal{y}$ 𝓎
$\mathcal{z}$ 𝓏	$\mathfrak{A}$ 𝔄	$\mathfrak{B}$ 𝔅	$\mathfrak{D}$ 𝔇	$\mathfrak{E}$ 𝔈

𝔉 𝔉	𝔊 𝔊	𝔍 𝔍	𝔎 𝔎	𝔏 𝔏
𝔐 𝔐	𝔑 𝔑	𝔒 𝔒	𝔓 𝔓	𝔔 𝔔
𝔖 𝔖	𝔗 𝔗	𝔘 𝔘	𝔙 𝔙	𝔚 𝔚
𝔛 𝔛	𝔜 𝔜	𝔞 𝔞	𝔟 𝔟	𝔠 𝔠
𝔡 𝔡	𝔢 𝔢	𝔣 𝔣	𝔤 𝔤	𝔥 𝔥
𝔦 𝔦	𝔧 𝔧	𝔨 𝔨	𝔩 𝔩	𝔪 𝔪
𝔫 𝔫	𝔬 𝔬	𝔭 𝔭	𝔮 𝔮	𝔯 𝔯

𝔰	𝔱	𝔲	𝔳	𝔴
𝔰	𝔱	𝔲	𝔳	𝔴
𝔵	𝔶	𝔷	𝔄	𝔅
𝔵	𝔶	𝔷	𝔸	𝔹
𝔇	𝔈	𝔉	𝔊	𝔍
𝔻	𝔼	𝔽	𝔾	𝕀
𝔍	𝔎	𝔏	𝔐	𝔒
𝕁	𝕂	𝕃	𝕄	𝕆
𝔖	𝔗	𝔘	𝔙	𝔚
𝕊	𝕋	𝕌	𝕍	𝕎
𝔛	𝔜	𝔞	𝔟	𝔠
𝕏	𝕐	𝕒	𝕓	𝕔
𝔡	𝔢	𝔣	𝔤	𝔥
𝕕	𝕖	𝕗	𝕘	𝕙

𝕚 𝕚	𝕛 𝕛	𝕜 𝕜	𝕝 𝕝	𝕞 𝕞
𝕟 𝕟	𝕠 𝕠	𝕡 𝕡	𝕢 𝕢	𝕣 𝕣
𝕤 𝕤	𝕥 𝕥	𝕦 𝕦	𝕧 𝕧	𝕨 𝕨
𝕩 𝕩	𝕪 𝕪	𝕫 𝕫		



```
<!DOCTYPE html>
<html>
<body>

<p> &Pi; </p>
<p> &Sigma; </p>
<p> &Omega; </p>

</body>
</html>
```

 # Output:

Π

Σ

Ω

```
<script>

console.log(/xyz/.test("vwxyz"));
// Output: true

console.log(/xyz/.test("vxwyz"));
// Output: false

</script>
```

This program will provide us a **Boolean output** indicating whether the string contains a match to the expression's pattern

```
<script>

console.log(encodeURIComponent("How?"));
// Output: How%3F

console.log(decodeURIComponent("How%3F"));
// Output: How?

</script>
```

```
<script>

for(let x = 1; x < 5; x++) {
console.log(x + " *^@^*");
}

</script>
```

 # Output:

```
1 *^@^*
2 *^@^*
3 *^@^*
4 *^@^*
```

```
<script>

let x = "Albert";
console.log("Hi " + x);
if(x.length > 5) {
console.log("Elsa");
}

</script>
```

 # Output:

```
Hi Albert
Elsa
```

```
<script>

let x = "Albert";
document.write("Hi " + x);
if (x.length > 7) {
document.write("Elsa");
}

</script>
```

 # Output:

```
Hi Albert
```

```
<script>
let x = true;
let y = true;
let z = true;
if (x) {
console.log("Albert Einstein");
} else if (y) {
console.log("Elsa Einstein");
} else if (z) {
console.log("Hans Albert Einstein");
} else {
console.log("David Einstein");
}
</script>
```

 # Output:

```
Albert Einstein
```

```
<script>
let x = false;
let y = true;
let z = true;
if (x) {
console.log("Albert Einstein");
} else if (y) {
console.log("Elsa Einstein");
} else if (z) {
console.log("Hans Albert Einstein");
} else {
console.log("David Einstein");
}
</script>
```

 # Output:

```
Elsa Einstein
```

```
<script>
let x = false;
let y = false;
let z = true;
if (x) {
console.log("Albert Einstein");
} else if (y) {
console.log("Elsa Einstein");
} else if (z) {
console.log("Hans Albert Einstein");
} else {
console.log("David Einstein");
}
</script>
```

 # Output:

```
Hans Albert Einstein
```

```
<script>
let x = false;
let y = false;
let z = false;
if (x) {
console.log("Albert Einstein");
} else if (y) {
console.log("Elsa Einstein");
} else if (z) {
console.log("Hans Albert Einstein");
} else {
console.log("David Einstein");
}
</script>
```

 # Output:

```
David Einstein
```

Document object model (DOM):

The way that HTML elements are arranged and maintained by web browsers

```
<script>

var x = 'Albert, Elsa, David';

console.log(x.split(/\s*,\s*/));

// Output: [ 'Albert', 'Elsa', 'David' ]


var i = '|x|y|z|'.split(/\|/);

console.log(i);

// Output: [ '', 'x', 'y', 'z', '' ]


</script>
```

```
<script>

let a = 12, b = 13;

console.log(false == (a > b));

// Output: true

// false is equal to false


console.log((a == 12) && (b == 13));

// Output: true

// Both comparisons are true

</script>
```

```
<script>

let a = 12, b = 13;
console.log((a > 13) || (b < 13));
// Output: false
// Neither comparison is true

console.log("fadé" === "fad\u00e9");
// Output: true

console.log("two" > "three");
// Output: true
// "tw" is alphabetically greater than "th"

</script>
```

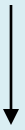
```
<script>
console.log(Math.pow(4, 1/3)); // The cube root of 4
// Output: 1.5874010519681994

console.log(Math.pow(4,23)); // 4 to the power 23
// Output: 70368744177664

// String converts to a number before comparing
console.log("0" == 0);
// Output: true

// Boolean converts to number before comparing
console.log(0 == false); // Output: true
console.log(1 == true); // Output: true

</script>
```

```
<script>

// This is a single-line comment.
/* This is a further comment. */ // And now we have another comment.
/*
 * This is still another comment.
 * It has numerous lines.
 */


// The two values are treated as equal
console.log(null == undefined); // Output: true


// Integers cannot start with "."
console.log(parseInt(".2")); // Output: NaN

// Integers cannot start with "$"
console.log(parseInt("$54.20")); // Output: NaN

// Addition after converting Boolean to number
console.log(true + true); // Output: 2
// 1 + 1 = 2


</script>
```

JavaScript comments are an effective approach to prevent code execution while testing new features because they are not actually executed. You can detect issues using this technique, which involves gradually eliminating comments until we reach the incorrect code.

```
<script>


console.log(3 + undefined ) // Output: NaN
// 3 + NaN = NaN

console.log("one" < 4); // Output: false
// "one" converted to NaN. Result is false.


</script>
```

```
a == 0 && b == 0
```



True if, and only if a and b are both 0

```
(a == 0 && b == 0) || !(c == 0)
```

a and b are both zero or c is non-zero

```
<script>


let a = .3 - .2;
let b = .2 - .1;
console.log(a == b); // false: the 2 values are not the same
console.log(a == .1); // false: .3 -.2 is not equal to .1
console.log(b == .1); // true: .2 -.1 is equal to .1

</script>
```

```
<script>

let x = 0; // Regular zero
let y = -0; // Negative zero
console.log(x === y); // true: zero and negative zero are equal
console.log(1/x === 1/y); // false: infinity and -infinity are not equal

</script>
```

- [] → An empty array
- [3+2, 5+4] → A Two element array. First element is 5 and second is 9

```
<script>
x = y = 11;
z = 12;
if (x == y)
if (y == z)
 console.log("Albert Einstein");
else
 console.log("Elsa Einstein");
 // Output: Elsa Einstein
</script>
```

```
<script>


x = [11, 12, 13, 14, 15]; // Start with a 5-element array

x.length = 3

console.log(x); // x is now [11, 12, 13]

x.length = 0

console.log(x); // Delete all elements. x is []


</script>
```

```
<script>

var [a, b] = [11, 12]; // Same as var a=11, b=12

[a, b] = [a+1, b+1]; // Same as a = a + 1, b = b+1

[a, b] = [b, a]; // Swap the value of the 2 variables

console.log([a, b]); // Output: [13,12]


</script>
```

```
<script>

var [a, b] = [11]; // a = 11, b = undefined
console.log([a, b]); // Output: [ 11, undefined ]


var[a, b] = [11, 12, 13]; // a = 11, b = 12
console.log([a, b]); // Output: [ 11, 12 ]


var[,a,,b] = [11, 12, 13, 14]; // a = 12, b = 14
console.log([a, b]); // Output: [ 12, 14 ]


var [a, [b, c]] = [11, [12, 12.5], 13]; // a = 11, b = 12
console.log([a, b]); // Output: [ 11, 12 ]


</script>
```

```
<script>

let i = [];
for(x=1; x<10; x++) {
 if (x % 2 === 0)
 i.push(x*x);
}
console.log(i);

// Output: [ 4, 16, 36, 64 ]

</script>
```

```
<script>

var x = 589;
var y = Number("55");
console.log(x + y); // Output: 644


const x = true;
const i = x ? 1 : 0;
console.log(i); // Output: 1


const a = 15;
const b = 13 - -a;
console.log(b); // Output: 28


</script>
```

```
<script>
var i = 2; // starting condition
while(i > 1 && i < 10) {
 i = i + 2; // 2 steps forward
 i = i - 1; // 1 step back
 console.log(i);
}
</script>
```

 # Output:

```
3
4
5
6
7
8
9
10
```

```
<script>

const i = true;
let a = 0;
const b = i || a++;
console.log(b);
// Output: true

const x = "ALBERT EINSTEIN";
console.log(x.toLowerCase().startsWith("al"))
// Output: true

</script>
```

```
<script>
/* The eval() method will return
the result of the last expression */

console.log(eval('13+14;15+16'));
// Output: 31
</script>
```

```
<script>

const x = false;
const i = x ? "Albert" : "Elsa";
console.log(i); // Output: Elsa


</script>
```

Aggregation:

The ability to enclose one object inside another object

```
<script>
const i = [4, 16, 18, 36];
console.log(i.every(x => x%2===0));
// Output: true
// no odd numbers
console.log(i.every(x => Number.isInteger(Math.sqrt(x))));
// Output: false
// 18 is not perfect square
</script>
```

```
<script>


const x = new Date(1988, 2, 11);

const y = new Date(2022, 3, 28);

console.log(x > y); // Output: false

console.log(x < y); // Output: true

const z = y - x;

console.log(z);

// Output: 1077062400000 milliseconds

const w = z/1000/60/60/24;

console.log(w);

 // Output: 12466 days


</script>
```

Application scaffolding:

A method that enables a programmer to specify and create a fundamental application that can generate, retrieve, update, and remove objects

```
<script>


console.log(JSON.parse('{ "Albert": 26, "James": 36 }'));
// Output: { Albert: 26, James: 36 }

console.log(JSON.parse('5496'));
// Output: 5496

console.log(JSON.parse('[11, 12, 13]'));
// Output: [ 11, 12, 13 ]


</script>
```

HTML Video

- The **HTML** `<video>` element specifies a standard way to embed a video in a web page.

Example

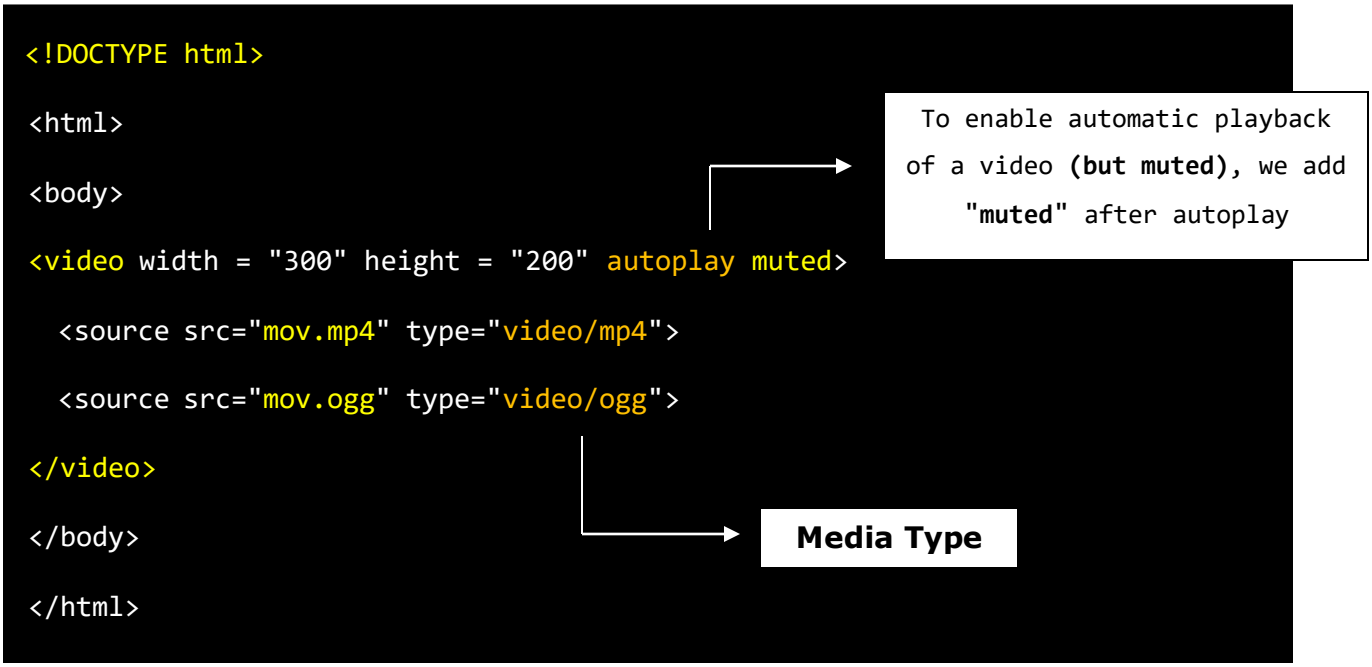
```
<!DOCTYPE html>
<html>
<body>
<video width = "300" height = "200" controls>
  <source src="mov.mp4" type="video/mp4">
  <source src="mov.ogg" type="video/ogg">
</video>
</body>
</html>
```

Video controls like play, pause, and volume are added through the **controls attribute**

```
<!DOCTYPE html>
<html>
<body>
<video width = "300" height = "200" autoplay>
  <source src="mov.mp4" type="video/mp4">
  <source src="mov.ogg" type="video/ogg">
</video>
</body>
</html>
```

We employ the **"autoplay" attribute** in order to play a video automatically

File Format



The web browser support for the **different video formats** is:

Browser	MP4	WebM	Ogg
Edge	YES	YES	YES
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO
Opera	YES	YES	YES

```
<!DOCTYPE html>
<html>
<body>
<video width = "300" height = "200" autoplay loop>
  <source src="mov.mp4" type="video/mp4">
  <source src="mov.ogg" type="video/ogg">
</video>
</body>
</html>
```

When a **video file** is finished playing, the **"loop" attribute** specifies that it will restart every time

HTML Audio

- The **HTML** **<audio>** element specifies a standard way to embed audio in a web page.

Example

```
<!DOCTYPE html>
<html>
<body>
<audio controls>
  <source src="music.ogg" type="audio/ogg">
  <source src="music.mp3" type="audio/mpeg">
</audio>
</body>
</html>
```

Audio controls like play, pause, and volume are added through the **controls attribute**

To play an **audio** file in HTML, we use the **<audio>** element

```
<!DOCTYPE html>
<html>
<body>
<audio controls autoplay>
  <source src="music.ogg" type="audio/ogg">
  <source src="music.mp3" type="audio/mpeg">
</audio>
</body>
</html>
```

We employ the **"autoplay" attribute** in order to play an audio automatically

```
<!DOCTYPE html>
<html>
<body>
<audio controls autoplay muted>
  <source src="music.ogg" type="audio/ogg">
  <source src="music.mp3" type="audio/mpeg">
</audio>
</body>
</html>
```

To enable automatic playback of a audio **(but muted)**, we add **"muted"** after autoplay

- **<audio>** tag defines audio content
- **<video>** tag defines video content

The web browser support for the **different audio formats** is:

Browser	MP3	WAV	OGG
Edge/IE	YES	YES*	YES*
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO
Opera	YES	YES	YES

HTML YouTube Videos

To play a video on a web page, do the following:

- Upload the video to YouTube
- Take a note of the "**video URL**"
- Define an **<iframe>** element in your web page
- Let the **src** attribute point to the video **URL**
- Use the width and height attributes to specify the dimension of the video player.

```
<!DOCTYPE html>
<html>
<body>
<iframe width="420" height="315" src="https://www.youtube.com/watch?v=3og4ONN38_0">
</iframe>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>

<embed width="100%" height="200px" src="index.html">

</body>
</html>
```

The **<embed>** element does not have a closing tag. It is used to include **"index.html"** in HTML Document

```
<!DOCTYPE html>
<html>
<body>

<!-- Float the image to the left -->
<img src="alan.png" alt="Alan" style="float: left; width: 100px; height: 100px;">

<!-- Float the image to the right -->
<img src="alan.png" alt="Alan" style="float: right; width: 100px; height: 100px;">

</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>
<!-- Unordered List with Circle Bullets -->
<ul style="list-style-type: circle;">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ul>
</body>
</html>
```

 # Output:

- Albert
- Elsa
- David

```html
<!DOCTYPE html>
<html>
<body>

<!-- Unordered List with Disc Bullets -->

<ul style="list-style-type: disc;">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ul>

</body>
</html>
```

 # Output:

- Albert
- Elsa
- David

```html
<!DOCTYPE html>
<html>
<body>

<!-- Unordered List with Square Bullets -->

<ul style="list-style-type: square;">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ul>

</body>
</html>
```

 # Output:

- Albert
- Elsa
- David

```
<!DOCTYPE html>
<html>
<body>

<!-- Unordered List without Bullets -->

<ul style="list-style-type: none;">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ul>

</body>
</html>
```

 # Output:

Albert

Elsa

David

```
<!DOCTYPE html>
<html>
<body>

<!-- Ordered List with Numbers -->

<ol type="1">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ol>

</body>
</html>
```

 # Output:

1. Albert
2. Elsa
3. David

```html
<!DOCTYPE html>
<html>
<body>

<!-- Ordered List with Letters -->

<ol type="A">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ol>

</body>
</html>
```

 # Output:

A. Albert
B. Elsa
C. David

```html
<!DOCTYPE html>
<html>
<body>

<!-- Ordered List with Lowercase Letters -->

<ol type="a">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ol>

</body>
</html>
```

 # Output:

a. Albert
b. Elsa
c. David

```html
<!DOCTYPE html>
<html>
<body>

<!-- Ordered List with Roman Numbers -->

<ol type="I">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ol>

</body>
</html>
```

 # Output:

I. Albert
II. Elsa
III. David

```html
<!DOCTYPE html>
<html>
<body>

<!-- Ordered List with Lowercase Roman Numbers -->

<ol type="i">
  <li> Albert </li>
  <li> Elsa </li>
  <li> David </li>
</ol>

</body>
</html>
```

 # Output:

i. Albert
ii. Elsa
iii. David

```html
<!DOCTYPE html>
<html>
<body>

<ul>
  <li> Alan Turing </li>
  <li> Albert Einstein
    <ul>
      <li> Elsa Einstein </li>
      <li> David Einstein </li>
    </ul>
  </li>
  <li> Charles Darwin </li>
</ul>

</body>
</html>
```

A Nested List

(List inside list)

 # Output:

- Alan Turing
- Albert Einstein
  - Elsa Einstein
  - David Einstein
- Charles Darwin

```html
<!DOCTYPE html>
<html>
<body>
<!-- Styling a <div> element -->
<div style="background-color: green; color: white; padding: 20px;">
  <h1> Albert Einstein </h1>
</div>

</body>
</html>
```

 # Output:

Albert Einstein

HTML Tables

HTML Table Tags

Tag	Description
<table>	Defines a table
<th>	Defines a header cell in a table
<tr>	Defines a row in a table
<td>	Defines a cell in a table

Example

```
<table>
  <tr>
    <td> Albert </td>
  </tr>
</table>
```

1 Column

 # Output:

Albert

```
<table>
  <tr>
    <td> Albert </td>
    <td> John </td>
    <td> Mary </td>
  </tr>
</table>
```

1 Row and 3 Columns

 # Output:

Albert John Mary

```
<table>
  <tr>
    <td> Albert </td>
    <td> John </td>
    <td> Mary </td>
  </tr>
    <tr>
    <td> Elsa </td>
    <td> Alan </td>
    <td> David </td>
  </tr>
    <tr>
    <td> Charles </td>
    <td> James </td>
    <td> William </td>
  </tr>
</table>
```

3 Rows and 3 Columns

 # Output:

Albert	John	Mary
Elsa	Alan	David
Charles	James	William

```
<!DOCTYPE html>
<html>
<head>
<style>
table, tr, td {
  border: 2px solid black;
}
</style>
</head>
```

A table with borders

The use of **"CSS border attribute"** to add a border to the table

```
<body>

<table>

<tr>
    <td> Albert </td>
    <td> John </td>
    <td> Mary </td>
  </tr>

<tr>
    <td> Elsa </td>
    <td> Alan </td>
    <td> David </td>
  </tr>

<tr>
    <td> Charles </td>
    <td> James </td>
    <td> William </td>
  </tr>

</table>

</body>
</html>
```

```
<script>

console.log(escape("Albert Einstein"));
// Output: Albert%20Einstein

console.log(unescape("Albert%20Einstein"));
// Output: Albert Einstein

</script>
```

 # Output:

Albert	John	Mary
Elsa	Alan	David
Charles	James	William

2 basic groups of data types:

- **Reference type:** These are complicated data types, primarily consisting of dates and strings.
- **Primitive type:** These types of data contain numerical information.

```
<!DOCTYPE html>
<html>
<head>
<style>
table, tr, td {
  border: 2px solid black;
  border-collapse: collapse;
}
</style>
</head>
<body>
<table>
<tr>
    <td> Albert </td>
    <td> John </td>
    <td> Mary </td>
  </tr>
<tr>
    <td> Elsa </td>
    <td> Alan </td>
    <td> David </td>
  </tr>
<tr>
    <td> Charles </td>
    <td> James </td>
    <td> William </td>
  </tr>
</table>
</body>
</html>
```

A table with collapsed borders

The **"CSS border-collapse attribute"** must be included if we wish the borders to merge into one border.

 # Output:

Albert	John	Mary
Elsa	Alan	David
Charles	James	William

```
<head>
<style>
table, tr, td {
  border: 2px solid black;
  border-collapse: collapse;
}
td {
  padding: 15px;
}
</style>
</head>
<body>
<table>
<tr>
    <td> Albert </td>
    <td> John </td>
    <td> Mary </td>
  </tr>
<tr>
    <td> Elsa </td>
    <td> Alan </td>
    <td> David </td>
  </tr>
<tr>
    <td> Charles </td>
    <td> James </td>
    <td> William </td>
  </tr>
</table>
</body>
```

A table with cell padding

The space between the content of a cell and its borders is specified by **cell padding.**

 # Output:

Albert	John	Mary
Elsa	Alan	David
Charles	James	William

```
<head>
<style>
table, th, td {
  border: 2px solid black;
  border-collapse: collapse;
}
th, td {
  padding: 15px;
}
</style>
</head>
<body>
<table>
<tr>
    <th> Name </th>
    <th> Age </th>
  </tr>
<tr>
    <td> Elsa </td>
    <td> 50 </td>
   </tr>
<tr>
    <td> Charles </td>
    <td> 75 </td>

  </tr>
</table>

</body>
```

A table with headings

 # Output:

Name	Age
Elsa	50
Charles	75

```
<!DOCTYPE html>
<html>

<head>
<style>
table, th, td {
  border: 2px solid black;
  border-collapse: collapse;
}
th, td {
  padding: 15px;
}
</style>
</head>

<body>
<table>
<tr>
    <th> Name </th>
    <th colspan="2"> Phone number </th>
  </tr>

<tr>
    <td> Elsa </td>
    <td> 080-65960971 </td>
    <td> 080-68950991 </td>
   </tr>

</table>

</body>
</html>
```

The **"colspan attribute"** can be used to make a cell span multiple columns.

 # Output:

Name	Phone number	
Elsa	080-65960971	080-68950991

CSS

(Cascading Style Sheets)

Cascading Style Sheets, or **CSS**, is a language that specifies how **HTML** elements should be displayed, including the layout, colours, and fonts of an **HTML** Document. Along with **HTML** and **JavaScript**, **CSS** is a key component of the **World Wide Web** and a simplistic design language made to make the process of generating appealing web pages easier.

CSS Syntax

World Wide Web Consortium (W3C)

Creator of CSS

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  color: red;
  text-align: center;
}
</style>
</head>

<body>
<h1> Albert Einstein </h1>

</body>
</html>
```

 # Output:

Albert Einstein

- **h1** is a **selector** in **CSS** (it points to the **HTML** element you want to style: **<h1>**).
- **color** is a property, and **red** is the property value
- **text-align** is a property, and **center** is the property value

CSS Selectors

- **CSS** selectors are used to "find" (or select) the **HTML** elements we want to style.

The CSS element Selector

- The **`element selector`** selects **HTML** elements based on the element name.

Example

- Here, all **`<h2>`** elements on the page will be **`center-aligned`** with a **`red`** text color.

```
<!DOCTYPE html>
<html>
<head>
<style>
h2 {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

<h2> Albert Einstein </h2>
<h2> Elsa Einstein </h2>
<h2> David Einstein </h2>

</body>
</html>
```

 # Output:

Albert Einstein

Elsa Einstein

David Einstein

The CSS id Selector

- The **`id selector`** uses the id attribute of an **HTML** element to select a specific element.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#my1 {
  text-align: center;
  color: red;
}

</style>
</head>
<body>

<h1 id="my1"> Albert Einstein </h1>

</body>
</html>
```

 # Output:

Albert Einstein

The CSS class Selector

- The **`class selector`** selects **HTML** elements with a **`specific class attribute`**.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
```

```
.my1 {
  text-align: center;
  color: red;
}
</style>
</head>
<body>
<p class="my1"> Elsa Einstein </p>
</body>
</html>
```

 # Output:

Elsa Einstein

The CSS Universal Selector

- The **`universal selector`** (*) selects all **HTML** elements on the page.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
* {
  text-align: center;
  color: blue;
}
</style>
</head>
<body>

<h1> Albert Einstein </h1>
<p> Elsa Einstein </p>
<p> David Einstein </p>

</body>
</html>
```

 # Output:

Albert Einstein

Elsa Einstein

David Einstein

The CSS Grouping Selector

- The **grouping selector** selects all the **HTML** elements with the same style definitions.
- Look at the following **CSS** code (the **h1**, **h2**, and **p** elements have the same style definitions):

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
h1, h2, p {
  text-align: center;
  color: red;
}
</style>
</head>

<body>

<h1> Albert Einstein </h1>
<h2> Elsa Einstein </h2>
<p> David Einstein </p>

</body>
</html>
```

 # Output:

Albert Einstein

Elsa Einstein

David Einstein

CSS Comments

- **Comments** are used to explain the code and may help when we edit the source code at a later time.
- **Comments** are ignored by web browsers.
- A **CSS** comment is placed inside the **<style>** element and starts with /* and ends with */:

Example

```
<!DOCTYPE html>

<html>

<head>

<style>

/* This is a single-line comment */

p {

  color: blue; /* Set text color to blue */

}

/* This is

a multi-line

Comment */

</style>

</head>

<body>

<p> Albert Einstein </p>

</body>

</html>
```

It's a good idea to include **comments** in the code because they can make it easier to understand if someone reads it or if it is later reviewed.

 # Output:

Albert Einstein

CSS Backgrounds

CSS background-color

- The **background-color property** specifies the background color of an element.

Example

```
<!DOCTYPE html>
<html>
<head>

<style>
body {
  background-color: yellow;
}
</style>

</head>
<body>

<h1> Albert Einstein </h1>

</body>
</html>
```

 # Output:

Albert Einstein

Opacity / Transparency

The "**opacity or transparency**" of an **HTML** element is defined by its opacity property. It takes values between 0.0 and 1.0. The more opacity, the lower the value.

opacity 1	opacity 0.6	opacity 0.3	opacity 0.1

```
<!DOCTYPE html>
<html>
<head>
<style>

div {
 /* Blue background with 60% opacity */
  background-color: blue;
  opacity: 0.6;

}

</style>
</head>
<body>

<div>
  <h1> Albert Einstein </h1>
</div>

</body>
</html>
```

 # Output:

CSS background-image

- The **`background-image property`** specifies an **image** to use as the background of an element.
- By default, the image is repeated so it covers the entire element.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url("img.png");
}
</style>
</head>

<body>

<h1> Albert Einstein </h1>

</body>
</html>
```

 # Output:

CSS background-repeat: repeat-x

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url("img.png");
  background-repeat: repeat-x;
}
</style>
</head>
```

Set "background-repeat: repeat-x" to repeat a background image horizontally.

```
<body>

<h1> Albert Einstein </h1>

</body>
</html>
```

CSS background-repeat: repeat-y

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url("img.png");
  background-repeat: repeat-y;
}
</style>
</head>
<body>

<h1> Albert Einstein </h1>

</body>
</html>
```

Set "background-repeat: repeat-y" to repeat a background image vertically.

CSS background-repeat: no-repeat

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url("img.png");
  background-repeat: no-repeat;
}
</style>
</head>
<body>


<h1> Albert Einstein </h1>


</body>
</html>
```

The background-image is not repeated

CSS background-position

- The **background-position property** is used to specify the position of the background image.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url("img.png");
  background-repeat: no-repeat;
  background-position: right top;
}
</style>
</head>

<body>
<h1> Albert Einstein </h1>
</body>
</html>
```

The **background image** is placed in the top-right corner of the **HTML** Document

CSS background - Shorthand property

Example

```
<style>
body {
background: #ffffff url("img.png") no-repeat right top; margin-right: 300px;
}
</style>
```

All the background properties in one declaration

CSS Borders

CSS Border Style

- The **border-style property** specifies what kind of border to display.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>

p.dotted {border-style: dotted;} /* Defines a dotted border  */
p.dashed {border-style: dashed;} /* Defines a dashed border  */
p.solid {border-style: solid;}  /*  Defines a solid border  */
p.double {border-style: double;} /* Defines a double border  */
p.groove {border-style: groove;} /* Defines a 3D grooved border */
p.ridge {border-style: ridge;}  /* Defines a 3D ridged border */
p.inset {border-style: inset;} /* Defines a 3D inset border */
p.outset {border-style: outset;} /* Defines a 3D outset border */
p.none {border-style: none;}  /* Defines no border */
p.hidden {border-style: hidden;} /* Defines a hidden border */
p.mix {border-style: dotted dashed solid double;}

</style>
</head>
<body>

<p class="dotted"> Albert Einstein </p>

<p class="dashed"> Albert Einstein </p>
```

```
<p class="solid"> Albert Einstein </p>
<p class="double"> Albert Einstein </p>
<p class="groove"> Albert Einstein </p>
<p class="ridge"> Albert Einstein </p>
<p class="inset"> Albert Einstein </p>
<p class="outset"> Albert Einstein </p>
<p class="none"> Albert Einstein </p>
<p class="hidden"> Albert Einstein </p>
<p class="mix"> Albert Einstein </p>

</body>
</html>
```

 # Output:

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

CSS Border Width

- The **`border-width property`** specifies the width of the four borders.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p.b1 {
  border-style: solid;
  border-width: 6px;
}

p.b2 {
  border-style: solid;
  border-width: medium;
}

p.b3 {
  border-style: dotted;
  border-width: 10px;
}

p.b4 {
  border-style: dotted;
  border-width: thick;
}

p.b5 {
  border-style: double;
  border-width: 4px;
}
```

In the **Java programming language**, the word **"Final"** has several different meanings:

- **Final Class:** This class is not inheritable.
- **Final Variable:** Once a value has been assigned to a final variable, it cannot be changed.
- **Final Method:** This Method is irrevocable.

```
p.b6{
  border-style: double;
  border-width: thick;
}
</style>
</head>
<body>

<p class="b1"> Albert Einstein </p>
<p class="b2"> Albert Einstein </p>
<p class="b3"> Albert Einstein </p>
<p class="b4"> Albert Einstein </p>
<p class="b5"> Albert Einstein </p>
<p class="b6"> Albert Einstein </p>

</body>
</html>
```

 # Output:

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

Albert Einstein

CSS Border Color

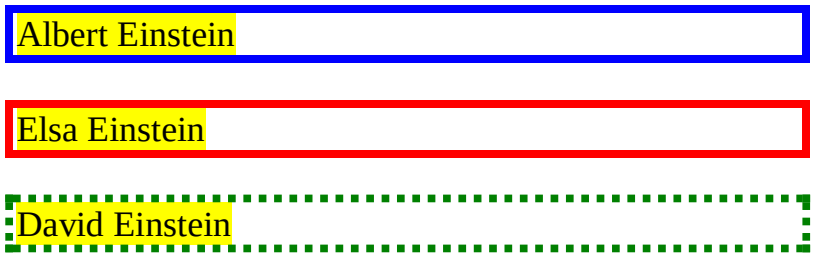
- The **`border-color property`** is used to set the color of the four borders.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p.b1 {
  border-style: solid;
  border-color: blue;
}

p.b2 {
  border-style: solid;
  border-color: red;
}

p.b3 {
  border-style: dotted;
  border-color: green;
}
</style>
</head>
<body>

<p class="b1"> Albert Einstein </p>
<p class="b2"> Elsa Einstein </p>
<p class="b3"> David Einstein </p>

</body>
</html>
```

A programmer can add new variables at runtime, and variable types are changed at runtime, which is why the **JavaScript programming language** is referred to be a **dynamically typed language**.

 # Output:



Specific Side Colors

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p.b1 {
  border-style: solid;
  border-color: green orange red yellow;
  /* green top, orange right, red bottom and yellow left */
}
</style>
</head>
<body>
<p class="b1"> Albert Einstein </p>
</body>
</html>
```

Output:

Albert Einstein

Did you know?

The end of **World War II** was significantly influenced by computer codes. **Alan Turing**, an English computer scientist, used his expertise in cryptology and mathematics to successfully decrypt the ENIGMA code machine used by the Nazis.

HEX Values

- The color of the border can be specified using a **hexadecimal value** (HEX):

Example

```
p.b1 {
  border-style: solid;
  border-color: #ff0000; /* red */
}
```

RGB Values

- The color of the border can be specified using a **RGB value** (RGB):

Example

```
p.b1 {
  border-style: solid;
  border-color: rgb(255, 0, 0); /* red */
}
```

HSL Values

- The color of the border can be specified using a **HSL value** (HSL):

Example

```
p.b1 {
  border-style: solid;
  border-color: hsl(0, 100%, 50%); /* red */
}
```

CSS Rounded Borders

- The border-radius property is used to add rounded borders to an **HTML** element:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p.b1 {
  border: 2px solid blue;
  padding: 5px;
}

p.b2 {
  border: 2px solid blue;
  border-radius: 12px;
  padding: 5px;
}
</style>
</head>
<body>

<p class="b1"> Albert Einstein </p>
<p class="b2"> Elsa Einstein </p>


</body>
</html>
```

Did you know?

The International Space Station uses **ADA**, a coding language that was created in 1980. This programming language gained acceptance as an **international standard** in 1995.

The **PHP** programming language was not intended for usage in widespread enterprise applications. **PHP** was developed by **Rasmus Lerdorf** solely for use in managing his own online project. But over time, **PHP** rose to become one of the most widely used programming languages available.

 # Output:

Albert Einstein

Albert Einstein

CSS Margins

- The CSS margin properties are used to create space around **HTML** elements, outside of any defined borders.

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  border: 1px solid black;
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 50px;
  background-color: yellow;
}
</style>
</head>
<body>

<div> Elsa Einstein was the second wife and cousin of Albert Einstein. </div>

</body>
</html>
```

Margin - Shorthand Property

```
margin: 100px 100px 150px 50px;
```

- top margin is 100px
- right margin is 100px
- bottom margin is 150px
- left margin is 50px

This ** ** element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 50px.

 # Output:

Elsa Einstein was the second wife and cousin of Albert Einstein.

CSS Padding

- The CSS padding properties are used to generate space around an **HTML** element's content, inside of any defined borders.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  border: 1px solid black;
  background-color: yellow;
  padding-top: 60px;
  padding-right: 40px;
  padding-bottom: 60px;
  padding-left: 70px;
}
</style>
</head>
<body>

<div> Elsa Einstein was the second wife and cousin of Albert Einstein. </div>

</body>
</html>
```

Padding - Shorthand Property

```
padding: 60px 40px 60px 70px;
```

- top padding is 60px
- right padding is 40px
- bottom padding is 60px
- left padding is 70px

This ** ** element has a top padding of 60px, a right padding of 40px, a bottom padding of 60px, and a left padding of 70px.

 # Output:

Elsa Einstein was the second wife and cousin of Albert Einstein.

CSS Height and Width

- The height and width properties are used to set the height and width of an **HTML** element.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  height: 100px;
  width: 40%;
  background-color: orange;
}
</style>
</head>
<body>

<div> Albert and Elsa Einstein. </div>

</body>
</html>
```

 # Output:

Albert and Elsa Einstein.

This ** ** element has a height of 100px and a width of 40%.

```
<style>
div {
  height: 100px;
  width: 400px;
  background-color: orange;
}
</style>
```

The ** ** element has a height of 100px and a width of 400px.

CSS Text

Text Alignment

- The **`text-align property`** is used to set the horizontal alignment of a text.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  text-align: center;
}

h2 {
  text-align: left;
}

h3 {
  text-align: right;
}
</style>
</head>

<body>
<h1> Albert Einstein </h1>
<h2> Elsa Einstein </h2>
<h3> David Einstein </h3>
</body>
</html>
```

#Output:

Albert Einstein

Elsa Einstein

David Einstein

The three headings are aligned center, left and right.

When the **text-align** property is set to "justify", every line is widened to the same width, and the left and right margins are aligned evenly.

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  border: 1px solid black;
  padding: 20px;
  width: 200px;
  height: 100px;
  text-align: justify;
  background-color: yellow;
}
</style>
</head>
<body>

<div>
Elsa Einstein was the second wife and cousin of Albert Einstein
</div>

</body>
</html>
```

 # Output:

Elsa Einstein was the second wife and cousin of Albert Einstein

```
<style>
div {
  border: 1px solid black;
  padding: 20px;
  width: 200px;
  height: 100px;
  background-color: yellow;
}
</style>
```

→

 # Output:

Elsa Einstein was the second wife and cousin of Albert Einstein

Text Decoration

- **The `text-decoration property`** is used to set or remove decorations from text.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  text-decoration: overline;
}

h2 {
  text-decoration: line-through;
}

h3 {
  text-decoration: underline;
}

p.mytxt {
  text-decoration: overline underline;
}
</style>
</head>

<body>
<h1> Albert Einstein </h1>
<h2> Elsa Einstein </h2>
<h3> David Einstein </h3>
<p class="mytxt"> Elsa Einstein was the second wife and cousin of Albert Einstein. </p>
</body>
</html>
```

 # Output:

Albert Einstein

~~Elsa Einstein~~

David Einstein

Elsa Einstein was the second wife and cousin of Albert Einstein.

Text Transformation

- The text-transform property is used to specify uppercase and lowercase letters in a text.

Example

```
<!DOCTYPE html>
<html>
<head>

<style>
h1.uppercase {
  text-transform: uppercase;
}

h1.lowercase {
  text-transform: lowercase;
}

h1.capitalize {
  text-transform: capitalize;
}
</style>

</head>

<body>

<h1 class="uppercase"> albert einstein </h1>
<h1 class="lowercase"> ALBERT EINSTEIN </h1>
<h1 class="capitalize"> albert einstein </h1>

</body>
</html>
```

 # Output:

ALBERT EINSTEIN

albert einstein

Albert Einstein

Text Indentation

- The **`text-indent property`** is used to specify the indentation of the first line of a text.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  text-indent: 60px;
}
</style>
</head>

<body>

<p> Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time. Einstein is best known for developing the theory of relativity, but he also made important contributions to the development of the theory of quantum mechanics. </p>

</body>
</html>
```

 # Output:

Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time. Einstein is best known for developing the theory of relativity, but he also made important contributions to the development of the theory of quantum mechanics.

Letter Spacing

- The **`letter-spacing property`** is used to specify the space between the characters in a text.

Example

```
<!DOCTYPE html>
<html>
<head>

<style>
h1 {
  letter-spacing: 7px;
}

h2 {
  letter-spacing: -3px;
}
</style>

</head>
<body>

<h1> Albert Einstein </h1>
<h2> Albert Einstein </h2>

</body>
</html>
```

 # Output:

A l b e r t E i n s t e i n

AlbertEinstein

Line Height

- The **`line-height property`** is used to specify the space between lines.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p.l1 {
  line-height: 0.8;
}

p.l2 {
  line-height: 1.5;
}
</style>
</head>
<body>

<p class="l1">
Albert Einstein was a German-born theoretical physicist. <br>
Best known for developing the theory of relativity. <br>
</p>

<p class="l2">
Albert Einstein was a German-born theoretical physicist. <br>
Best known for developing the theory of relativity.  <br>
</p>

</body>
</html>
```

 # Output:

Albert Einstein was a German-born theoretical physicist.
Best known for developing the theory of relativity.

Albert Einstein was a German-born theoretical physicist.

Best known for developing the theory of relativity.

Word Spacing

- The **word-spacing property** is used to specify the space between the words in a text.

```
<!DOCTYPE html>
<html>
<head>
<style>

p.l1 {
  word-spacing: 12px;
}

p.l2 {
  word-spacing: -3px;
}

</style>
</head>
<body>

<p class="l1"> Elsa Einstein was the second wife and cousin of Albert Einstein. </p>

<p class="l2"> Elsa Einstein was the second wife and cousin of Albert Einstein. </p>

</body>
</html>
```

 # Output:

Elsa Einstein was the second wife and cousin of Albert Einstein.

ElsaEinsteinwasthe secondwifeandcousinofAlbertEinstein

Text Shadow

- The **text-shadow property** adds shadow to text.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  text-shadow: 2px 2px blue;
}
</style>
</head>
<body>

<h1> Albert </h1>

</body>
</html>
```

Font Awesome Text Icons

- fa-500px
- fa-adjust
- fa-adn
- fa-align-center
- fa-align-justify
- fa-align-left
- fa-align-right
- fa-amazon
- fa-ambulance
- fa-anchor
- fa-android
- fa-angellist
- fa-angle-double-down
- fa-angle-double-left
- fa-angle-double-right
- fa-angle-double-up
- fa-angle-down
- fa-angle-left
- fa-angle-right
- fa-angle-up
- fa-apple
- fa-archive
- fa-area-chart
- fa-arrow-circle-down
- fa-arrow-circle-left
- fa-arrow-circle-o-down
- fa-arrow-circle-o-left
- fa-arrow-circle-o-right
- fa-arrow-circle-o-up
- fa-arrow-circle-right
- fa-arrow-circle-up
- fa-arrow-down
- fa-arrow-left
- fa-arrow-right
- fa-arrow-up
- fa-arrows
- fa-arrows-alt
- fa-arrows-h
- fa-arrows-v
- fa-asterisk
- fa-at
- fa-automobile
- fa-backward
- fa-balance-scale
- fa-ban
- fa-bank
- fa-bar-chart
- fa-bar-chart-o
- fa-barcode
- fa-bars
- fa-battery-0
- fa-battery-1
- fa-battery-2
- fa-battery-3
- fa-battery-4
- fa-battery-empty
- fa-battery-full
- fa-battery-half
- fa-battery-quarter
- fa-battery-three-quarters
- fa-bed
- fa-beer
- fa-behance
- fa-behance-square
- fa-bell
- fa-bell-o
- fa-bell-slash
- fa-bell-slash-o
- fa-bicycle
- fa-binoculars

 fa-birthday-cake

 fa-bitbucket

 fa-bitbucket-square

 fa-bitcoin

 fa-black-tie

 fa-bluetooth

 fa-bluetooth-b

 fa-bold

 fa-bolt

 fa-bomb

 fa-book

 fa-bookmark

 fa-bookmark-o

 fa-briefcase

 fa-btc

 fa-bug

 fa-building

 fa-building-o

 fa-bullhorn

 fa-bullseye

 fa-bus

 fa-buysellads

 fa-cab

 fa-calculator

 fa-calendar

 fa-calendar-check-o

 fa-calendar-minus-o

 fa-calendar-o

 fa-calendar-plus-o

 fa-calendar-times-o

 fa-camera

 fa-camera-retro

 fa-car

 fa-caret-down

 fa-caret-left

 fa-caret-right

 fa-caret-square-o-down

 fa-caret-square-o-left

 fa-caret-square-o-right

 fa-caret-square-o-up

 fa-caret-up

 fa-cart-arrow-down

 fa-cart-plus

 fa-cc

 fa-cc-amex

 fa-cc-diners-club

 fa-cc-discover

 fa-cc-jcb

 fa-cc-mastercard

 fa-cc-paypal

 fa-cc-stripe

 fa-cc-visa

 fa-certificate

 fa-chain

 fa-chain-broken

 fa-check

 fa-check-circle

 fa-check-circle-o

 fa-check-square

 fa-check-square-o

 fa-chevron-circle-down

 fa-chevron-circle-left

 fa-chevron-circle-right

 fa-chevron-circle-up

 fa-chevron-down

 fa-chevron-left

 fa-chevron-right

 fa-chevron-up

 fa-child

 fa-chrome

 fa-circle

 fa-circle-o

 fa-circle-o-notch

 fa-circle-thin

 fa-clipboard

 fa-clock-o

 fa-clone

 fa-close

 fa-cloud

 fa-cloud-download

 fa-cloud-upload

 fa-cny

 fa-code

 fa-code-fork

 fa-codepen

 fa-codiepie

 fa-coffee

 fa-cog

 fa-cogs

 fa-columns

 fa-comment

 fa-comment-o

 fa-commenting

 fa-commenting-o

 fa-comments

 fa-comments-o

 fa-compass

 fa-compress

 fa-connectdevelop

 fa-contao

 fa-copy

 fa-copyright

 fa-creative-commons

 fa-credit-card

 fa-credit-card-alt

 fa-crop

 fa-crosshairs

 fa-css3

 fa-cube

 fa-cubes

 fa-cut

 fa-cutlery

 fa-dashboard

 fa-dashcube

 fa-database

 fa-dedent

 fa-delicious

 fa-desktop

 fa-deviantart

 fa-diamond

 fa-digg

 fa-dollar

 fa-dot-circle-o

 fa-download

 fa-dribbble

 fa-dropbox

 fa-drupal

 fa-edge

 fa-edit

 fa-eject

 fa-ellipsis-h

 fa-ellipsis-v

 fa-empire

 fa-envelope

 fa-envelope-o

 fa-envelope-square

 fa-eraser

 fa-eur

 fa-euro

 fa-exchange

 fa-exclamation

 fa-exclamation-circle

 fa-exclamation-triangle

 fa-expand

 fa-expeditedssl

 fa-external-link

 fa-external-link-square

 fa-eye

 fa-eye-slash

 fa-eyedropper

 fa-facebook

 fa-facebook-f

 fa-facebook-official

 fa-facebook-square

 fa-fast-backward

 fa-fast-forward

 fa-fax

 fa-feed

 fa-female

 fa-fighter-jet

 fa-file

 fa-file-archive-o

 fa-file-audio-o

 fa-file-code-o

 fa-file-excel-o

 fa-file-image-o

 fa-file-movie-o

 fa-file-o

 fa-file-pdf-o

 fa-file-photo-o

 fa-file-picture-o

 fa-file-powerpoint-o

 fa-file-sound-o

 fa-file-text

 fa-file-text-o

 fa-file-video-o

 fa-file-word-o

 fa-file-zip-o

 fa-files-o

 fa-film

 fa-filter

 fa-fire

 fa-fire-extinguisher

 fa-firefox

 fa-flag

 fa-flag-checkered

 fa-flag-o

 fa-flash

 fa-flask

 fa-flickr

 fa-floppy-o

 fa-folder

 fa-folder-o

 fa-folder-open

 fa-folder-open-o

 fa-font

 fa-fonticons

 fa-fort-awesome

 fa-forumbee

 fa-forward

 fa-foursquare

 fa-frown-o

 fa-futbol-o

 fa-gamepad

 fa-gavel

 fa-gbp

 fa-ge

 fa-gear

 fa-gears

 fa-genderless

 fa-get-pocket

 fa-gg

 fa-gg-circle

 fa-gift

 fa-git

 fa-git-square

 fa-github

 fa-github-alt

 fa-github-square

 fa-gittip

 fa-glass

 fa-globe

 fa-google

 fa-google-plus

 fa-google-plus-square

 fa-google-wallet

 fa-graduation-cap

 fa-gratipay

 fa-group

 fa-h-square

 fa-hacker-news

 fa-hand-grab-o

 fa-hand-lizard-o

 fa-hand-o-down

 fa-hand-o-left

 fa-hand-o-right

 fa-hand-o-up

 fa-hand-paper-o

 fa-hand-peace-o

 fa-hand-pointer-o

 fa-hand-rock-o

 fa-hand-scissors-o

 fa-hand-spock-o

 fa-hand-stop-o

 fa-hashtag

 fa-hdd-o

 fa-header

 fa-headphones

 fa-heart

 fa-heart-o

 fa-heartbeat

 fa-history

 fa-home

 fa-hospital-o

 fa-hotel

 fa-hourglass

 fa-hourglass-1

 fa-hourglass-2

 fa-hourglass-3

 fa-hourglass-end

 fa-hourglass-half

 fa-hourglass-o

 fa-hourglass-start

 fa-houzz

 fa-html5

 fa-i-cursor

 fa-ils

 fa-image

 fa-inbox

 fa-indent

 fa-industry

 fa-info

 fa-info-circle

 fa-inr

 fa-instagram

 fa-institution

 fa-internet-explorer

 fa-intersex

 fa-ioxhost

 fa-italic

 fa-joomla

 fa-jpy

 fa-jsfiddle

 fa-key

 fa-keyboard-o

 fa-krw

 fa-language

 fa-laptop

 fa-lastfm

 fa-lastfm-square

 fa-leaf

 fa-leanpub

 fa-legal

 fa-lemon-o

 fa-level-down

 fa-level-up

 fa-life-bouy

 fa-life-buoy

 fa-life-ring

 fa-life-saver

 fa-lightbulb-o

 fa-line-chart

 fa-link

 fa-linkedin

 fa-linkedin-square

 fa-linux

 fa-list

 fa-list-alt

 fa-list-ol

 fa-list-ul

 fa-location-arrow

 fa-lock

 fa-long-arrow-down

 fa-long-arrow-left

 fa-long-arrow-right

 fa-long-arrow-up

 fa-magic

 fa-magnet

 fa-mail-forward

 fa-mail-reply

 fa-mail-reply-all

 fa-male

 fa-map

 fa-map-marker

 fa-map-o

 fa-map-pin

 fa-map-signs

 fa-mars

 fa-mars-double

 fa-mars-stroke

 fa-mars-stroke-h

 fa-mars-stroke-v

 fa-maxcdn

 fa-meanpath

 fa-medium

 fa-medkit

 fa-meh-o

 fa-mercury

 fa-microphone

 fa-microphone-slash

 fa-minus

 fa-minus-circle

 fa-minus-square

 fa-minus-square-o

 fa-mixcloud

 fa-mobile

 fa-mobile-phone

 fa-modx

 fa-money

 fa-moon-o

 fa-mortar-board

 fa-motorcycle

 fa-mouse-pointer

 fa-music

 fa-navicon

 fa-neuter

 fa-newspaper-o

 fa-object-group

 fa-object-ungroup

 fa-odnoklassniki

 fa-odnoklassniki-square

 fa-opencart

 fa-openid

 fa-opera

 fa-optin-monster

 fa-outdent

 fa-pagelines

 fa-paint-brush

 fa-paper-plane

 fa-paper-plane-o

 fa-paperclip

 fa-paragraph

 fa-paste

 fa-pause

 fa-pause-circle

 fa-pause-circle-o

 fa-paw

 fa-paypal

 fa-pencil

 fa-pencil-square

 fa-pencil-square-o

 fa-percent

 fa-phone

 fa-phone-square

 fa-photo

 fa-picture-o

 fa-pie-chart

 fa-pied-piper

 fa-pied-piper-alt

 fa-pinterest

 fa-pinterest-p

 fa-pinterest-square

 fa-plane

 fa-play

 fa-play-circle

 fa-play-circle-o

 fa-plug

 fa-plus

 fa-plus-circle

 fa-plus-square

 fa-plus-square-o

 fa-power-off

 fa-print

 fa-product-hunt

 fa-puzzle-piece

 fa-qq

 fa-qrcode

 fa-question

 fa-question-circle

 fa-quote-left

 fa-quote-right

 fa-ra

 fa-random

 fa-rebel

 fa-recycle

 fa-reddit

 fa-reddit-alien

 fa-reddit-square

 fa-refresh

 fa-registered

- fa-remove
- fa-renren
- fa-reorder
- fa-repeat
- fa-reply
- fa-reply-all
- fa-retweet
- fa-rmb
- fa-road
- fa-rocket
- fa-rotate-left
- fa-rotate-right
- fa-rouble
- fa-rss
- fa-rss-square
- fa-rub
- fa-ruble
- fa-rupee
- fa-safari
- fa-save
- fa-scissors
- fa-scribd
- fa-search
- fa-search-minus
- fa-search-plus
- fa-sellsy
- fa-send
- fa-send-o
- fa-server
- fa-share
- fa-share-alt
- fa-share-alt-square
- fa-share-square
- fa-share-square-o
- fa-shekel
- fa-sheqel
- fa-shield
- fa-ship
- fa-shirtsinbulk
- fa-shopping-bag
- fa-shopping-basket
- fa-shopping-cart
- fa-sign-in
- fa-sign-out
- fa-signal
- fa-simplybuilt
- fa-sitemap
- fa-skyatlas
- fa-skype
- fa-slack
- fa-sliders
- fa-slideshare
- fa-smile-o
- fa-soccer-ball-o
- fa-sort
- fa-sort-alpha-asc
- fa-sort-alpha-desc
- fa-sort-amount-asc
- fa-sort-amount-desc
- fa-sort-asc
- fa-sort-desc
- fa-sort-down
- fa-sort-numeric-asc
- fa-sort-numeric-desc
- fa-sort-up
- fa-soundcloud
- fa-space-shuttle
- fa-spinner
- fa-spoon
- fa-spotify

 fa-square

 fa-square-o

 fa-stack-exchange

 fa-stack-overflow

 fa-star

 fa-star-half

 fa-star-half-empty

 fa-star-half-full

 fa-star-half-o

 fa-star-o

 fa-steam

 fa-steam-square

 fa-step-backward

 fa-step-forward

 fa-stethoscope

 fa-sticky-note

 fa-sticky-note-o

 fa-stop

 fa-stop-circle

 fa-stop-circle-o

 fa-street-view

 fa-strikethrough

 fa-stumbleupon

 fa-stumbleupon-circle

 fa-subscript

 fa-subway

 fa-suitcase

 fa-sun-o

 fa-superscript

 fa-support

 fa-table

 fa-tablet

 fa-tachometer

 fa-tag

 fa-tags

 fa-tasks

 fa-taxi

 fa-television

 fa-tencent-weibo

 fa-terminal

 fa-text-height

 fa-text-width

 fa-th

 fa-th-large

 fa-th-list

 fa-thumb-tack

 fa-thumbs-down

 fa-thumbs-o-down

 fa-thumbs-o-up

 fa-thumbs-up

 fa-ticket

 fa-times

 fa-times-circle

 fa-times-circle-o

 fa-tint

 fa-toggle-down

 fa-toggle-left

 fa-toggle-off

 fa-toggle-on

 fa-toggle-right

 fa-toggle-up

 fa-trademark

 fa-train

 fa-transgender

 fa-transgender-alt

 fa-trash

 fa-trash-o

 fa-tree

 fa-trello

 fa-tripadvisor

 fa-trophy

 fa-truck

 fa-try

 fa-tty

 fa-tumblr

 fa-tumblr-square

 fa-turkish-lira

 fa-tv

 fa-twitch

 fa-twitter

 fa-twitter-square

 fa-umbrella

 fa-underline

 fa-undo

 fa-university

 fa-unlink

 fa-unlock

 fa-unlock-alt

 fa-unsorted

 fa-upload

 fa-usb

 fa-usd

 fa-user

 fa-user-md

 fa-user-plus

 fa-user-secret

 fa-user-times

 fa-users

 fa-venus

 fa-venus-double

 fa-venus-mars

 fa-viacoin

 fa-video-camera

 fa-vimeo

 fa-vimeo-square

 fa-vine

 fa-vk

 fa-volume-down

 fa-volume-off

 fa-volume-up

 fa-warning

 fa-wechat

 fa-weibo

 fa-weixin

 fa-whatsapp

 fa-wheelchair

 fa-wifi

 fa-wikipedia-w

 fa-windows

 fa-won

 fa-wordpress

 fa-wrench

 fa-xing

 fa-xing-square

 fa-y-combinator

 fa-y-combinator-square

 fa-yahoo

 fa-yc

 fa-yc-square

 fa-yelp

 fa-yen

 fa-youtube

 fa-youtube-play

 fa-youtube-square

```
<!DOCTYPE html>

<html>

<head>

<!-- To use the Font Awesome icons, we add the following line inside the
<head> section of our HTML Document -->

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

</head>

<body>

<i class="fa fa-angle-right"></i>

<i class="fa fa-angle-right" style="font-size: 56px;"></i>

<i class="fa fa-angle-right" style="font-size: 70px; color: blue;"></i>

</body>

</html>
```

 # Output:

```
<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

</head>

<body>

<i class="fa fa-heart fa-lg"></i>

<i class="fa fa-heart fa-2x"></i>

<i class="fa fa-heart fa-3x"></i>

<i class="fa fa-heart fa-4x"></i>

<i class="fa fa-heart fa-5x"></i>

</body>

</html>
```

The **icon sizes** can be increased in proportion to their container using the **fa-lg** (**33% increase**), **fa-2x**, **fa-3x**, **fa-4x**, or fa-5x classes.

 # Output:

```
<!DOCTYPE html>

<html>

<body>

<button onclick="window.print();"> Print this page </button>

</body>

</html>
```

JavaScript program to print the contents of the current window

```
<script>

let a = 7;

let b = 5;

let c = 9;

let s = (a + b + c)/2;

document.write(Math.sqrt(s*((s-a)*(s-b)*(s-c))));

// Output: 17.41228014936585

</script>
```

Heron's formula states that the area of a triangle whose sides have lengths a, b, and c is $\sqrt{s(s-a)(s-b)(s-c)}$ where s is the **semiperimeter** of the triangle; that is, $s = \frac{a+b+c}{2}$

JavaScript program to find the area of a triangle where lengths of the three of its sides are 7, 5, 9

```
<script>

// Prints the current website URL

document.write(document.URL);

</script>
```

```
<script>

// Display the extension of a filename

console.log("1.txt".split('.').pop()); // Output: txt

console.log("2.pdf".split('.').pop()); // Output: pdf

</script>
```

- Write a **JavaScript program** to calculate the sum of the two given numbers. If the two numbers are equal, then return 2 times their sum.

Solution:

```
<script>
let x = 10;
let y = 10;
if (x == y) {
 console.log(2 * (x + y));
}
else {
 console.log(x + y);
}
</script>
```

 # Output:

```
40
```

```
<script>
let x = 10;
let y = 20;
if (x == y) {
 console.log(2 * (x + y));
}
else {
 console.log(x + y);
}
</script>
```

 # Output:

```
30
```

```
<script>

let x = 70;

let y = 45;

console.log((x == 25 || y == 25) || (x - y == 25));

// Output: true

</script>
```

JavaScript program to check 2 given numbers and return true if one of the number is 25 or if their difference is 25

```
<script>

let x = 100;

console.log((x % 5 == 0 || x % 10 == 0));

// Output: true

</script>
```

JavaScript program to check whether a given positive number is a multiple of 5 or a multiple of 10

```
<script>

console.log("ecdafb".split("").sort().join(""));

// Output: abcdef

</script>
```

JavaScript program to sort the letters of a given string in alphabetical order

```
<script>

x=[20, 30, 40];

console.log(x[0] + x[1] + x[2]);

// Output: 90

</script>
```

JavaScript program to calculate the sum of 3 elements of a given array of numbers of length "3"

```
<script>

x=23;

console.log(x % 10 + Math.floor(x / 10));

// Output: 5

</script>
```

23 → 2 + 3 → 5

JavaScript program to add 2 digits of a given positive integer of length "2"

```
<script>

function myfunc() {

for(var i = 0; i < 2; i++) {

console.log("Albert");

}

}

var x = Date.now();

myfunc();

var y = Date.now();

console.log("Total time taken : " + (y - x) + " milliseconds");

</script>
```

JavaScript program to measure the time taken by a function **"myfunc()"** to execute

 # Output:

```
Albert

Albert

Total time taken : 4 milliseconds
```

```
<script>

console.log(/^[a-z][a-z0-9+.-]*:/.test('https://google.com'));

// Output: true

console.log(/^[a-z][a-z0-9+.-]*:/.test('/temp/files'));

// Output: false

</script>
```

JavaScript program that will return true if the given string is an absolute URL, otherwise, it will return false

```
<script>

const distance = (a0, b0, a1, b1) => Math.hypot(a1 - a0, b1 - b0);

console.log(distance(2, 3, 4, 2));

// Output: 2.23606797749979

</script>
```

JavaScript program to calculate the Euclidean distance between two points

```
<script>

let sum = 0;

for(var i = 0; i < 20; i++) {

if (i % 2 == 0 && i % 4 == 0) {

     sum = sum + i;

   }

}

console.log(sum);

// Output: 40


</script>
```

JavaScript program to sum the multiples of 2 and 4 under 20

```
<script>

console.log(''.length==0); // Output: true

console.log('albert'.length==0); // Output: false

</script>
```

JavaScript program to check whether a string is blank or not

```
<?php

# Display the PHP version and configuration information

phpinfo();

?>
```

```
<?php

# Display the current file name

echo basename($_SERVER['PHP_SELF']);

?>
```

```
<?php
# Check whether the page is HTTPS or HTTP enabled
if (!empty($_SERVER['HTTPS'])) {
  echo 'HTTPS enabled';
}
else {
echo 'HTTP enabled';
}
?>
```

```
<?php

# Redirect a user to "https://www.wikipedia.org/"

header('Location: https://www.wikipedia.org/');

?>
```

K-Means Clustering	K-Nearest Neighbor (KNN)
An unsupervised learning algorithm mainly used for **clustering**	A supervised learning algorithm mainly used for **classification** and sometimes even for **regression**

Number of clusters

Number of nearest neighbors

A field of research focuses on the creation of computer-based technologies based on the ideas of **quantum theory**

Challenges of quantum computing in the software sector:

- Locating substances capable of keeping a quantum state for prolonged enough to allow for computations
- Constructing a quantum computational model that can be expanded to carry out exceedingly challenging calculations.
- Constructing methodologies that can utilize the quantum state to carry out computations more quickly than traditional techniques

Time Series Forecasting:

A technique for forecasting future outcomes using data from the past. It involves developing patterns through historical study, using them to make judgments and influence rational decision in the future

Time series analysis	Time series forecasting
Illustrates how data evolves over time	Figure out which way the data is changing

Data collection + Data cleaning → **Data Engineer**

Data cleaning + Exploratory data analysis → **Data Analyst**

Building + deployment of model → **Machine Learning Engineer**

Data scientist

Statistical Interaction:

When 2 or more variables interact and this has an impact on a third variable, this is referred to as a statistical interaction

PYTHON is used for data cleaning because it has modules like **NumPy** and **Pandas** that facilitate it to remove incorrect data entries

Progressive rendering:

An idea that suggests presenting the most important stuff to the browser first and the less important pieces afterwards, as needed by the user

HTTP ETag:

An HTTP header used by browsers to make constrained requests to resources and validate Web cache

HTML5 Web Storage:

A web storage mechanism that, while being faster and considerably more effective than cookies, allows users to save some data locally on the user's computer

Cross-site scripting (XSS):

A hacker inserts **malicious executable scripts** into a reputable application or website's source code. Attackers frequently provide a malicious attachment to a user and encourage them to click on to start an **XSS** attack.

```
<script>
function myfunc() {
 console.log("Albert Einstein");
}
myfunc();
</script>
```

```
<script>
var myfunc=function() {
  console.log("Albert Einstein");
}
myfunc();
</script>
```

 # Output: Albert Einstein

```
<script>
myfunc();
function myfunc() {
 console.log("Albert Einstein");
}
</script>
```

Outputs **"Albert Einstein"** even when the function is declared after calling

```
<script>
function mult(x) {
  return function(y){
    return x * y;
  }
}
console.log(mult(5)(3));
// Output: 15
</script>
```

```
<script>
function myfunc(num) {
  console.log(num + 200);
}
myfunc(210);
// Output: 410
</script>
```

```
<script>

console.log(myfunc(14, 13));

function myfunc(x, y) {

  return x + y;

  console.log("Albert"); // This line of code does not get executed

}

</script>
```

We use the "**return keyword**" to return a value, the function execution is stopped as soon as the return statement is invoked

```
# Output:  27
```

A **Callback** is a **JavaScript function** that is provided as an argument or a parameter to a method. A function called the **Callback** must be run following the completion of another function.

```
<script>

console.log(2 +  "3" + "3");

// Output: 233

console.log(2 +  +"3" + "3");

// Output: 53

console.log(2 +  -"3" + "3");

// Output: -13

console.log(+"2" +  "3" + "3");

// Output: 233

</script>
```

```
<script>

console.log(11 < 12 < 13);

// Output: true
```

11 < 12 → true

true has value 1, so 1 < 13 → true

```
console.log(13 > 12 > 11);

// Output: false
```

13 > 12 → true

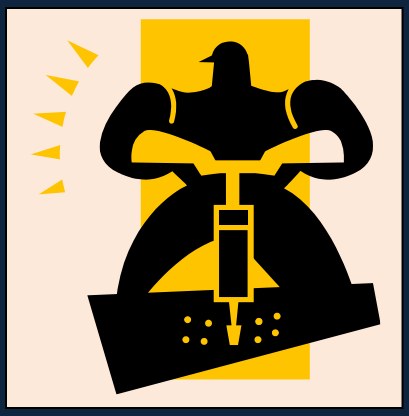
true has value 1, so 1 > 11 → false

```
</script>
```

```
<script>

console.log(typeof 25);

// Output: number

console.log(typeof typeof 25);

// Output: string

</script>
```

typeof 25 will return "number" and typeof "number" will return string

```
<script>

const myfunc=()=> {

console.log("Albert Einstein");

}

myfunc();

// Output: Albert Einstein

</script>
```



Memorization:

A method of optimization that is largely used to speed up computer programs by caching the output of complex function calls and returning it when the same inputs are received again

Constructor functions

Functions that are used in JavaScript to construct objects. When we want to build many objects with related properties and methods, we use it

```
<script>

const names = ['John', 'Albert', 'James'];

const age = ['16', '25', '36'];

console.log([...names, ...age]);

// Output: [ 'John', 'Albert', 'James', '16', '25', '36' ]

</script>
```

Factors to be taken into account to avoid overfitting:

- Train with more data
- Assessing and comparing learning algorithms by splitting data into **two sections** for learning or training a model and the other for model validation
- Defining an extremely significant number of training iterations and stop training when the model's performance on a **holdout validation dataset** stops progressing
- Utilising a variety of techniques and strategies to create models that **best match** the data

Factors to be taken into account to avoid underfitting:

- Increasing the dataset's feature count
- Training the data using more robust and elaborate models

- The type of the project
- The data or information
- The time
- The difficulty of the algorithm and the project
- Efficiency of the Algorithm and the Objective

Factors to be taken into account to choose an algorithm

```php
<?php

$x = '6';

$y = &$x;

$y = "2$y";

echo $x.", ".$y;

?>
```

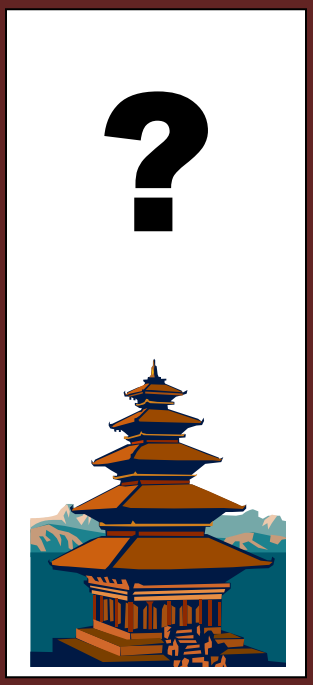
 # Output: 26, 26

```javascript
<script>

function mult(x, y, z) {

  return x * y * z;

}

console.log(mult(...[1, 2, 3]));

// Output: 6

// 1 × 2 × 3 = 6

</script>
```

```javascript
<script>

var regex = /\d+/g;

var x = "Albert Einstein's 1905 Papers";

console.log(x.match(regex));

// Output: [ '1905' ]

</script>
```

Using a **regular expression**, we can get any numbers from a string

```
<style>
  body * {
    margin: 30px;
    border: 2px solid orange;
  }
</style>
<div onclick="alert('Albert Einstein')">Albert
  <div onclick="alert('Elsa Einstein')">Elsa
    <div onclick="alert('David Einstein')">David
    </div>
  </div>
</div>
```




```
<script>
const age = 50;
age >= 60 ? console.log("Senior citizen") : console.log("Not a senior citizen");
// Output: Not a senior citizen
</script>
```

```
<script>
  document.write("Albert Einstein");
</script>
<noscript>
JavaScript is not supported by your browser.
</noscript>
```

The **<noscript>** tag is used to recognize and respond to browsers with disabled **JavaScript**

```php
<?php

# Print 2 PHP variables using single "echo" statement

$x = "Albert";

$y = "Einstein";

echo $x." ". $y;

?>
```

Output:

```
Albert Einstein
```

```php
<?php

for($i=1;$i<=5;$i++) {

for($j=1;$j<=$i;$j++) {

     echo "*";

  }

 echo "\n";

}

?>
```

Output:

```
*
**
***
****
*****
```

Data Science Pipeline

A series of steps that transforms the unstructured and perplexing data from many sources – such as surveys, reviews, comments, suggestions, feedbacks, lists of sales, polls, etc.– into a format that is easy to store and utilize for analysis

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  background-color: yellow;

  width: 200px;

  height: 65px;

  border: 1px solid;

  overflow: visible;

}

</style>

</head>

<body>


<div> Albert Einstein was a German-born theoretical physicist, widely
acknowledged to be one of the greatest and most influential physicists of all
time. </div>


</body>

</html>
```

Output:

Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time.

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  background-color: yellow;

  width: 200px;

  height: 65px;

  border: 1px solid;

  overflow: hidden;

}

</style>

</head>

<body>


<div> Albert Einstein was a German-born theoretical physicist, widely
acknowledged to be one of the greatest and most influential physicists of all
time. </div>


</body>

</html>
```

Output:

Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most

The use of "calc()" function to calculate the width and height of a <p> element

```
<!DOCTYPE html>
<html>
<head>
<style>
#mytxt {
  width: 200px;
  height: 65px;
  border: 1px solid black;
  background-color: yellow;
}
</style>
</head>
<body>
<p id ="mytxt"> Alan Mathison Turing </p>
</body>
</html>
```

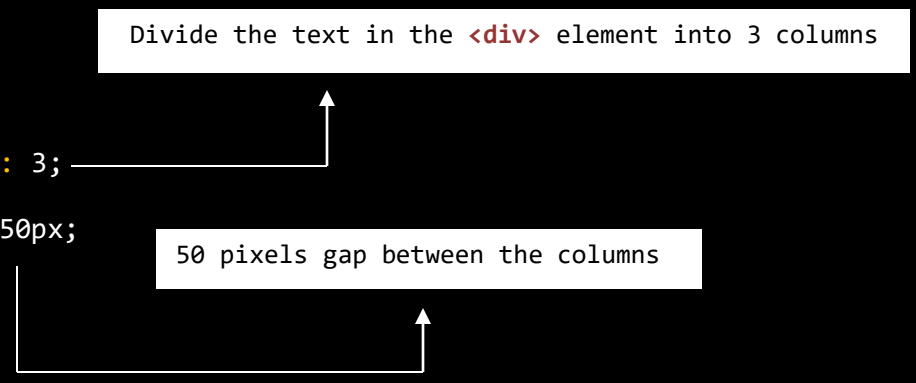
```
#mytxt {
  width: calc(400px - 200px);
  height: calc(100px - 35px);
  border: 1px solid black;
  background-color: yellow;
}
```

Output:

Alan Mathison Turing

```php
<?php

// PHP program to print letters from 'A' to 'Z'

for($i = ord('A'); $i <= ord('Z'); $i++)

 echo chr($i);

# Output: ABCDEFGHIJKLMNOPQRSTUVWXYZ

?>
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.mytxt {

  column-count: 3;

  column-gap: 50px;

}

</style>

</head>

<body>

<div class="mytxt"> Albert Einstein was a German-born theoretical
physicist, widely acknowledged to be one of the greatest and most
influential physicists of all time. Einstein is best known for developing
the theory of relativity, but he also made important contributions to the
development of the theory of quantum mechanics.

</div>

</body>

</html>
```



Output:

Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time. Einstein is best known for developing the theory of relativity, but he also made important contributions to the development of the theory of quantum mechanics.

```
<!DOCTYPE html>
<html>
<head>
<style>
.mytxt {
  column-count: 3;
  column-gap: 50px;
  column-rule-style: solid;
  column-rule-width: 5px;
  column-rule-color: green;
}
</style>
</head>
<body>
<div class="mytxt"> Albert Einstein was a German-born theoretical
physicist, widely acknowledged to be one of the greatest and most
influential physicists of all time. Einstein is best known for developing
the theory of relativity, but he also made important contributions to the
development of the theory of quantum mechanics.
</div>
</body>
</html>
```

- The color of the rule between columns is specified by the column-rule-color attribute.
- The width of the rule between columns is specified by the column-rule-width attribute.
- The style of the rule between columns is specified by the column-rule-style attribute.

Output:

Albert Einstein was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time. Einstein is best known for developing the theory of relativity, but he also made important contributions to the development of the theory of quantum mechanics.

The Prediction Effect:

In business, a prediction with lower accuracy is better than a speculation

The data effect:

There is always something worthwhile to acquire from the data since it always has a narrative to tell

The Induction Effect:

Machine learning is driven by creativity; when computer programs are used, tactics created in part through informal human creativity and imagination are successful in creating prediction models that work well on new cases

The Ensemble Effect:

Predictive models that are combined into an ensemble make up for one another's shortcomings, increasing the likelihood that the ensemble will forecast accurately as a whole over the individual models

Torch:

An open source machine learning library built on the Lua programming language that offers a variety of deep learning algorithms

How a data scientist operates?

- Recognize the business issue
- Browse and gather information
- Extract the data
- Clean the data
- Search for and add missing data
- Standardize data
- Evaluate data, look for patterns, and forecast the future
- Clearly communicate the outcome so that **"enterprise organisation"** may grasp it

Information regarding:

- What is typical?
- What can we predict?
- What is the standard?
- How likely is that?

can be obtained using statistics

An **artificial neuron** is referred to as a **Perceptron**. It is the most basic neural network that is conceivable. **Machine learning** is built on neural networks. One of the most important scientific achievements in **Modern history** is the neural network. Problems that **algorithms** can't solve can be tackled by **neural networks**.

- **Face Recognition**
- **Speech Recognition**

Deep Learning is based on neural networks.

TensorFlow.js

A well-known **JavaScript library** for machine learning is **Tensorflow**. We can train and use **machine learning** in the web browser due to **Tensorflow**. We can incorporate machine learning features into any web application using **Tensorflow**.

```
<!DOCTYPE html>
<html>
<head>
<!-- To use TensorFlow.js, we add the following script tag to our HTML Document -->
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>
</head>
<body>
<!-- Tensor Addition -->
<script>
const x = tf.tensor([[6, 5], [4, 3], [2, 1]]);
const y = tf.tensor([[1,-2], [3,-1], [1,-1]]);
// Addition of 2 tensors using x.add(y) method
const z= x.add(y);
document.write(z); // Prints the result
// Output: Tensor [[7, 3], [7, 2], [3, 0]]
</script>
</body>
</html>
```

AWS TrustedAdvisor:

A real-time, web-based suggestion service that offers suggestions to help us adhere to AWS best practices

```
<!DOCTYPE html>

<html>

<head>

<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>

</head>

<body>

<!-- Tensor Subtraction -->

<script>

const x = tf.tensor([[6, 5], [4, 3], [2, 1]]);

const y = tf.tensor([[1,-2], [3,-1], [1,-1]]);

// Subtraction of 2 tensors using x.sub(y) method

const z= x.sub(y);

document.write(z); // Prints the result

// Output: Tensor [[5, 7], [1, 4], [1, 2]]

</script>

</body>

</html>
```

Microservices architecture is a group of smaller, independently deployable services as compared to a **monolithic application**, which is constructed as a single integrated unit.

The life cycle of a **ML model** comprises 3 stages:

- Data Gathering
- Training
- Conclusion

```
<!DOCTYPE html>

<html>

<head>

<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>

</head>

<body>

<!-- Tensor Multiplication -->

<script>

const x = tf.tensor([[6, 5], [4, 3], [2, 1]]);

const y = tf.tensor([[1,-2], [3,-1], [1,-1]]);

// Multiplication of 2 tensors using x.mul(y) method

const z= x.mul(y);

document.write(z); // Prints the result

// Output: Tensor [[6 , -10], [12, -3 ], [2 , -1 ]]

</script>

</body>

</html>
```

AWS Cloud Fargate:

A serverless, pay-as-you-go compute platform that liberates us from server administration so we can concentrate on creating applications

```
<!DOCTYPE html>

<html>

<head>

<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>

</head>

<body>

<!-- Tensor Division -->

<script>

const x = tf.tensor([[6, 4], [4, 3], [2, 1]]);

const y = tf.tensor([[1,-2], [2,-1], [1,-1]]);

// Division of 2 tensors using x.div(y) method

const z= x.div(y);

document.write(z); // Prints the result

// Output: Tensor [[6, -2], [2, -3], [2, -1]]

</script>

</body>

</html>
```

As a result of **AI** technology, computers may imitate mankind in:

- Speaking
- Reasoning
- Studying
- Designing
- Knowing

```
<!DOCTYPE html>

<html>

<head>

<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>

</head>

<body>

<!-- Tensor Square -->

<script>

const x = tf.tensor([[6, 4], [4, 3], [2, 1]]);

// Square a tensor using x.square() method

const z= x.square();

document.write(z); // Prints the result

// Output: Tensor [[36, 16], [16, 9 ], [4 , 1 ]]

</script>

</body>

</html>
```

Brain.js

A JavaScript library that conceals the mathematics' difficulty to make it simple to grasp neural networks

The collection of information makes up to 80% of a ML project

- What information is needed?
- What information is readily available?
- How do we choose the information?
- How should information be gathered?
- How can the information be cleaned?
- How should the information be completely ready?
- What to do with the information?

Human vs Computing Machine

Human	Computing Machine
Intelligent but imprecise and slow	Foolish but precise and fast

Client-Server Model:

In a **client-server model**, a client communicates with and makes requests to a computer server. The server completes tasks for the client and provides information.

Compute Optimized Instances:

Designed for applications that require a lot of computing power, like some scientific simulations or high-performance web servers

Memory Optimized Instances:

Designed for applications that require a lot of storage power, like running Spark or Hadoop or doing real-time big data analytics

- **Dynamic scaling:** The ability to automatically change capacity depending on load or other metrics
- **Predictive scaling:** The ability of changing capacity in response to anticipated demand

Load Allocation:

The service that distributes incoming traffic among the resources that is accessible. It will divide up the resources according to their availability.

AWS Simple Notification:

A managed cloud service that enables us to mass-deliver messages and send notifications to our customers

Amazon Simple Queue Service:

A fully managed message queuing service that allows us to decouple and scale serverless applications, distributed systems, and microservices

How AWS Lambda Functions?

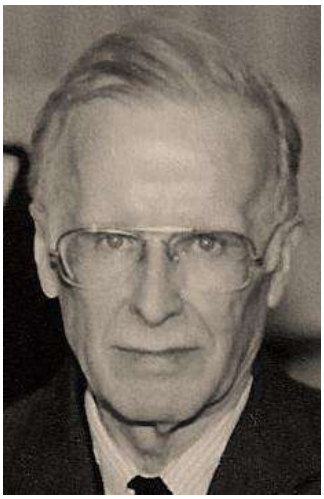
- Upload your program to Lambda
- Prepare the code to launch an event
- Program only executes when provoked or triggered
- Pay only when your program is running

Amazon Elastic Kubernetes Service:

A managed service that enables us to create Kubernetes clusters on AWS without manually installing the Kubernetes on EC2 compute instances

Did you know?

Ada Lovelace, a prominent female mathematician and the offspring of eminent poet Lord Byron, created the first computer program in human history.



John Backus created the first high-level programming language called FORTRAN for IBM in 1954. It was made available for purchase in 1957. Even now, FORTRAN is used to program scientific and mathematical tools.

30 Bash Scripts for Beginners

```
echo "Albert Einstein"

# Output: Albert Einstein
```

```
echo -n "\nAlan \t Mathison \t Turing\n"

# Output: \nAlan \t Mathison \t Turing\n

echo -e "\nAlan \t Mathison \t Turing\n"

# Output: Alan     Mathison        Turing
```

```
# multiply the numbers 2 and 3

((x=2*3))

# prints the result

echo $x

# Output: 6
```

```
# add the numbers 2 and 3

((x=2+3))

# prints the result

echo $x

# Output: 5
```

```
: '

calculate the

square value of the number: 4.

'


((y=4*4))

# prints the result

echo $y

# Output: 16
```

Multiline comment

```
x=true
i=0
while [ $x ]
do
echo $i
if [ $i -eq 4 ];
then
break
fi
((i++))
done
```

Output:

```
0
1
2
3
4
```

```
for ((i=5; i>0; i--))
do
echo -n "$i "
done
printf "\n"
# Output: 5 4 3 2 1
```

```
for ((i=0; i<5; i++))
do
echo -n "$i "
done
printf "\n"
# Output: 0 1 2 3 4
```

```
for ((i=0; i<5; i++))
do
echo "$i "
done
printf "\n"
```

Output:

```
0
1
2
3
4
```

```
echo "Please enter your name:"

read x

echo "Welcome $x to Myw3schools.com"

# Output:

Please enter your name:

Manju # entered name

Welcome Manju to Myw3schools.com
```

```
i=5

if [ $i -lt 10 ];

then

echo "i is less than 10"

else

echo "i is greater than 10"

fi

# Output: i is less than 10
```

```
i=5

if [ $i -gt 10 ];

then

echo "i is less than 10"

else

echo "i is greater than 10"

fi

# Output: i is greater than 10
```

```
echo "Enter username:"

read x

echo "Enter password:"

read y

if [[ ( $x == "manju" && $y == "123" ) ]]; then

echo "user authentication passed"

else

echo "user authentication failed"

fi
```

Output:

```
Enter username:

manju # entered username

Enter password:

123 # entered password

user authentication passed
```

```
echo "Please enter a number:"
read i
if [[ ( $i -eq 25 || $i -eq 50 ) ]]
then
echo "You guessed it right"
else
echo "You guessed it wrong"
fi
```

Output:

```
Please enter a number:
25 # entered number
You guessed it right
```

```
echo "Please enter a number:"
read i
if [ $i -eq 2 ];
then
echo "You earned the first prize"
elif [ $i -eq 4 ];
then
echo "You earned the second prize"
elif [ $i -eq 8 ];
then
echo "You earned the third prize"
else
echo "Please try again later"
fi
```

Output:

```
Please enter a number:
2 # entered number
You earned the first prize
```

```
echo "Please enter a number:"
read i
case $i in
2)
echo "You earned the first prize" ;;
4)
echo "You earned the second prize" ;;
8)
echo "You earned the third prize" ;;
*)
echo "Please try again later" ;;
esac
```

Output:

```
Please enter a number:
5 # entered number
Please try again later
```

```
read -p "Please enter a string: "  x
read -p "Please enter a string: "  y
if [ "$x" == "$y" ]
then
  echo "Both strings are equal"
else
  echo "Both strings are not equal"
fi
```

Output:

```
Please enter a string: john # entered string
Please enter a string: john # entered string
Both strings are equal
```

```
read -p "Please enter a string: "  x

read -p "Please enter a string: "  y

if [ "$x" != "$y" ]

then

  echo "Both strings are not equal"

else

  echo "Both strings are equal"

fi
```

Output:

```
Please enter a string: john # entered string

Please enter a string: alan # entered string

Both strings are not equal
```

```
a="Albert "

b=" Einstein"

echo "$a$b"

c=$a+$b

c+=

echo $c
```

Output:

```
Albert Einstein

Albert + Einstein
```

```
# The value 7 indicates the starting point from where the

# substring will start and 6 indicates the length of the substring


x="Master Python in 10 days"

i=${x:7:6}

echo $i

# Output: Python
```

```
echo "Please enter a number:"

read x

echo "Please enter a number:"

read y

((z=x*y))

echo "The product of $x and $y is: $z"
```

Output:

```
Please enter a number:

3 # entered number

Please enter a number:

6 # entered number

The product of 3 and 6 is: 18
```

```
function AL()

{

echo 'Alan Turing was a brilliant British mathematician.'

}

AL

# Output: Alan Turing was a brilliant British mathematician.
```

```
x() {

area=$(($1 * $2))

echo "Area of a rectangle is: $area"

}

x 60 10

# Output: Area of a rectangle is: 600
```

```
x() {

volume=$(($1 * $2 * $3))

echo "Volume of a rectangle of length $1, breadth $2 and height $3 is: $volume"

}

x 12 16 18

# Output: Volume of a rectangle of length 12, breadth 16 and height 18 is: 3456
```

```
x=0

while [ $x -le 4 ]

do

   echo "$x"

   let x++

done
```

Output:

```
0
1
2
3
4
```

```
# Infinite while loop

while true

do

  echo "Albert" # Press CTRL + C to exit from infinite while loop

done
```

```
function x() {

a="Howdy, $name"

echo $a

}

echo "Please enter your name:"

read name

i=$(x)

echo "$i"
```

Output:

```
Please enter your name:

manju # entered name

Howdy, manju
```

```
echo "Enter the name of the directory:"

read x

`mkdir $x`

# Output:

Enter the name of the directory:

myfiles # entered directory name

# Create a directory named 'myfiles'
```

```
for names in Sham Mary John James Lawrence

do

   echo "$names"

done
```

Output:

```
Sham

Mary

John

James

Lawrence
```

```
echo "Enter the name of the directory:"
read x
if [ -d "$x" ]
then
echo "Directory exists"
else
`mkdir $x`
echo "Directory is created"
fi
```

Output:

```
Enter the name of the directory:
myfiles # entered directory name
```

- Check for 'myfiles':
- If 'myfiles' exists:
- Prints **Directory exists** otherwise
- **'myfiles' directory** is created

```
for i in *
do
  ls -l $i
done
```

Read all files from the current directory

```
x='1.txt'
while read line; do
echo $line
done < $x
```

Read the file '1.txt'

```
echo "Enter the name of the file to remove:"

read x

rm -i $x

# Output:

Enter the name of the file to remove:

1.txt # entered file name

1.txt file is deleted
```

```
x=("Albert" "was" "a" "German-born" "theoretical physicist")

echo ${x[@]}

echo ${!x[@]}

# Output:

Albert was a German-born theoretical physicist

0 1 2 3 4
```

```
# Prints login name, current date, time, user name

# and current working directory



echo "Hello, $LOGNAME"

echo "Today's date is `date`"

echo "Username is `who i am`"

echo "Current directory is `pwd`"
```

```
for PID in $$
do
echo $PID
done
```

Prints PID of the current shell

```
id=10
x="Albert"
```

1.sh

```
source 1.sh

echo "Hi $x"
echo "Your id is $id"

# Output:
            Welcome Albert
            Your id is 10
```

```
myfunc(){
   echo $1 $2 $3 $4
}
myfunc 01. Alan Mathison Turing
# Output: 01. Alan Mathison Turing
```

```
Year=`date +%Y`

Month=`date +%m`

Day=`date +%d`

Hour=`date +%H`

Minute=`date +%M`

Second=`date +%S`

echo `date` # Output: Wed 29 Jun 2022 05:53:06 PM UTC

echo "Current Date is: $Day-$Month-$Year" # Output: Current Date is: 29-06-2022

echo "Current Time is: $Hour:$Minute:$Second" # Output: Current Time is: 17:53:06
```

```
echo "Albert" > abc.txt

 if [ $? -eq 0 ]; then

  echo "The command executed successfully"

else

  echo "Sorry, command execution failed"

fi

# Output: The command executed successfully

# The command: echo "Albert" > abc.txt executed successfully
```

```
for ((x=-4; x<0; x++))

do

  echo "$x"

done
```

Output:

```
-4

-3

-2

-1
```

```
for i in {1..10}; do echo -n "$i"; done
# Output: 12345678910


x="manju"
if grep ${x} /etc/passwd
then
  echo "Hey! string found"
else
  echo "Sorry, no matching string found"
fi
```

Check whether or not the file '/etc/passwd' contains the string 'manju'

```
echo "Please enter a number:"
read x
if [ $x -gt 0 ]
then
    echo "Positive number"
elif [ $x -lt 0 ]
then
    echo "Negative number"
else
    echo "Neither Positive nor Negative number"
fi
```

Output:

```
Please enter a number:
0 # entered number
Neither Positive nor Negative number
```

```
x=(1 2 3 4 5)
sum=0
for a in ${x[@]}
do
    sum=`expr $sum + $a`
done
echo $sum
# Output: 15
# 1 + 2 + 3 + 4 + 5 = 15
```

```
printf 'Albert'
printf '%s\n' 'Einstein'
printf -- '-%.0s' {1..15}
# Output:
AlbertEinstein
-----------------------
```

```
x="Alan+Mathison+Turing"
a=($(echo $x | tr "+" "\n"))
for i in "${a[@]}"
do
echo $i
done
# Output:
            Alan
            Mathison
            Turing
```

```
i=1
until [ $i -gt 4 ]
do
echo 'Albert'
((i++))
done
# Output:
            Albert
            Albert
            Albert
            Albert
```

```
import itertools

print(list(itertools.chain(range(0, 2), range(12, 14))))

# Output: [0, 1, 12, 13]
```

```
# ASCII value of C is 67

# ASCII value of D is 68

print({ord('C'): ord('D')})

# Output: {67: 68}
```

```
# ASCII value of A is 65

# ASCII value of B is 66

# ASCII value of C is 67

# ASCII value of D is 68

print(tuple(ord(i) for i in 'ABCD'))

# Output: (65, 66, 67, 68)
```

```
print(eval('11 + 11 == 22')) # Output: True

print(eval('11 + 11 != 22')) # Output: False

print(eval('"P" + "Q" + "R"')) # Output: PQR

print(eval('"DLBERT".translate({68: 65})')) # Output: ALBERT

print(eval('"DDD".count("D")')) # Output: 3

print(eval('["#"] * 2')) # Output: ['#', '#']
```

```
i= 2

print(eval("i * 8", {"i": i}, {})) # Output: 16



print(eval("pow(2, 4)", {}, {})) # Output: 16

print(eval("__import__('math').sqrt(16)", {}, {})) # Output: 4.0
```

```
import numpy as np

print(np.sqrt(16)) # Output: 4.0
```

```
a = 'Alan Turing'

import io

x = io.StringIO(a)

print(x.read()) # Output: Alan Turing
```

```
a = 'Alan Turing'

import io

x = io.StringIO(a)

print(x.read(5)) # Output: Alan
```

```
a = 'Alan Turing'

import io

x = io.StringIO(a)

print(x.seek(15)) # Output: 15
```

```
import sys

for x in range(3):

  sys.stdout.write('Albert\n')
```

```
for x in range(3):

  print('Albert')
```

```
import sys

for x in range(3):

  sys.stderr.write('Albert\n')
```

Output:

Albert

Albert

Albert

```
from urllib.parse import urlencode

x = {'name': 'Albert Einstein'}

print(urlencode(x)) # Output: name=Albert+Einstein
```

```
print(repr(repr(1915) + ' Papers')) # Output: '1915 Papers'

print([x for x in (11, 12)]) # Output: [11, 12]
```

```
x = ['A', 'B', 'C','D', 'E']

a = 0

for a in range(0, len(x)):

    print(x[a])
```

Output:

A

B

C

D

E

```
x = ['A', 'B', 'C','D', 'E']

a = 0

for a in range(2, len(x)):

    print(x[a])
```

Output:

```
C

D

E
```

```
x = {'A', 'B', 'C','D', 'E'}

x.add('D')

print(x) # Output: {'E', 'B', 'C', 'A', 'D'}



x = {'A', 'B', 'C','D', 'E'}

x.add('F')

print(x) # Output: {'E', 'B', 'C', 'A', 'F', 'D'}
```

```
import sys

print(sys.version_info)

print(sys.version)

# Output:

    sys.version_info(major=3, minor=7, micro=3, releaselevel='final', serial=0)

        3.7.3 (default, Mar 27 2019, 17:13:21) [MSC v.1915 64 bit (AMD64)]



print(b'unny' + b'rabbit') # Output: b'unnyrabbit'
```

```
print('bunny' == 'bunny') # Output: True

print(b'unny' == 'bunny') # Output: False

print(b'unny %s' % b'rabbit') # Output: b'unny rabbit'

print('alan %s' % 'turing') # Output: alan turing
```

```
x = 'price'

y = 12.568

print('%-10s : $%.2f' % (x, y))

# Output: price      : $12.57
```

```
book = [

 ('price: $', 8.77),

 ('price: $', 3.6),

 ('price: $', 50),

]

for i, (x, y) in enumerate(book):

 print('%d: %-12s  %.2f' % (i, x, y))
```

Output:

```
0: price: $      8.77

1: price: $      3.60

2: price: $      50.00
```

```
p = 861.45891

print("{:5.2f}".format(p)) # Output: 861.46

q = 861.45891

print("%5.2f" % q) # Output: 861.46
```

```
x = 1905

y = 'Papers'

print('{} {}'.format(x, y)) # Output: 1905 Papers



x = 1925

y = 'Dollars'

print('{:.2f} {:<10}'.format(x, y)) # Output: 1925.00 Dollars
```

```
print('%.2f%%' % 25.8) # Output: 25.80%

print('{} = {{}}'.format(1905)) # Output: 1905 = {}
```

```
x = 6

y = 4

print(f'{x:.{y}f}') # Output: 6.0000
```

```
x = ['Albert', 'Alan', 'James']

y = [len(i) for i in x]

print(y) # Output: [6, 4, 5]
```

```
for x in range(3):
 print('Hi')
else:
 print('Albert')
```

Output:

```
Hi
Hi
Hi
Albert
```

```
for x in range(3):
 print('Hi')
 if x == 1:
  break
else:
 print('Albert')
```

Output:

```
Hi
Hi
```

```
for a in []:
 print('Hi')
else:
 print('Albert')
# Output: Albert
```

```
while False:
 print('Hi')
else:
 print('Albert')
```

```
x = iter(range(2, 4))
a, b = x
print(f'{a} & {b}') # Output: 2 & 3
```

```
x = {}
y = 'Alan'
z = []
x.setdefault(y, z)
print(x)
z.append('Turing')
print(x)
```

Output:

```
{'Alan': []}
{'Alan': ['Turing']}
```

```
p = float(input("Enter the marks obtained in Practical: "))

q = float(input("Enter the marks obtained in Theory: "))

if (p >= 15 and q >= 35) or (p + q) >=50:

  print("\n You are selected")

else:

  print("\n Sorry, try next time")

# Output:

Enter the marks obtained in Practical: 26 # Entered marks

Enter the marks obtained in Theory: 36 # Entered marks

You are selected
```

```
# hello.py

print("Hello World!")



# Executes the hello.py file in the interpreter

import os

os.system('hello.py') # Output: Hello World!
```

```
p = "Alan\n"

q = "Turing"

print(p + q)
```

Output:

Alan

Turing

```
from time import sleep

from datetime import datetime

def i(x, y=datetime.now()):

 print(f'{y}: {x}')

i('Albert')

sleep(2) # wait for 2 seconds

i('Einstein')
```

Output:

```
2022-06-30 18:51:33.611028: Albert

wait for 2 seconds

2022-06-30 18:51:33.611028: Einstein
```

```
# python program to calculate the power of a number

# Assign values to x and n

x = 4

n = 3


# Method 1:

b = x ** n

print("%d to the power %d is %d" % (x,n,b)) # Output: 4 to the power 3 is 64


# Method 2:

b = pow(x,n)

print("%d to the power %d is %d" % (x,n,b)) # Output: 4 to the power 3 is 64


# Method 3:

import math

b = math.pow(x,n)

print("%d to the power %d is %5.2f" % (x,n,b)) # Output: 4 to the power 3 is 64.00
```

```
# import getpass module

import getpass

# Take password from the user

x = getpass.getpass('Please enter your password:')

# Check the password

if x == "alan@123":

   print("Authentication passed")

else:

   print("Authentication failed")



# Output:

Please enter your password: alan@123 # entered password

Authentication failed
```

```
# List the numbers greater than 14 and divisible by 5

a = [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

print([i for i in a if i > 14 if i % 5 == 0]) # Output: [15, 20]
```

```


import itertools

print(list(itertools.repeat('Alan', 4)))

# Output: ['Alan', 'Alan', 'Alan', 'Alan']
```

```
import itertools

x = itertools.cycle([4, 5])

print([next(x) for _ in range (4)])

# Output: [4, 5, 4, 5]
```

```
import itertools

a, b  = itertools.tee(['Alan', 'Turing'], 2)

print(list(a))

print(list(b))
```

Output:

```
['Alan', 'Turing']

['Alan', 'Turing']
```

```
import itertools

print(list(itertools.product([3, 4], repeat=1)))

# Output: [(3,), (4,)]

print(list(itertools.product([3, 4], repeat=2)))

# Output: [(3, 3), (3, 4), (4, 3), (4, 4)]
```

```
import itertools

print(list(itertools.product([6, 8], ['x', 'y'])))

# Output: [(6, 'x'), (6, 'y'), (8, 'x'), (8, 'y')]
```

```
# Python program to list files in a directory

# Import os module to read directory

import os

# Set the directory path

x = os.listdir('C:/Users/Manju')

# Print the content of the directory

for i in x:

   print(i)
```

```
import itertools

print(list(itertools.combinations([2, 4, 6], 1)))

# Output: [(2,), (4,), (6,)]

print(list(itertools.combinations([2, 4, 6], 2)))

# Output: [(2, 4), (2, 6), (4, 6)]

print(list(itertools.combinations_with_replacement([2, 4, 6], 1)))

# Output: [(2,), (4,), (6,)]

print(list(itertools.combinations_with_replacement([2, 4, 6], 2)))

# Output: [(2, 2), (2, 4), (2, 6), (4, 4), (4, 6), (6, 6)]

print(list(itertools.permutations([2, 4, 6], 1)))

# Output: [(2,), (4,), (6,)]

print(list(itertools.permutations([2, 4, 6], 2)))

# Output: [(2, 4), (2, 6), (4, 2), (4, 6), (6, 2), (6, 4)]
```

```
# Define the string

x = 'Albert Einstein was a German-born theoretical physicist'

# Define the search string

i = 'Albert'

print("%s appears %d times" % (i, x.count(i)))

# Output: Albert appears 1 times
```

```
a, b, *c = range(4)

print(a, b,  c) # Output: 0 1 [2, 3]



a, b, *c = range(2)

print(a, b,  c) # Output: 0 1 []
```

```
a, *b, c, d = range(6)

print(a, b,  c) # Output: 0 [1, 2, 3] 4



*a, b, c, d = range(6)

print(a, b,  c) # Output: [0, 1, 2] 3 4
```

```
x = list(range(10))

print(x)

# Output: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

x[3:6] = [60, 80]

print(x)

# Output: [0, 1, 2, 60, 80, 6, 7, 8, 9]
```

```
x = list(range(10))

print(x)

# Output: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

x[3::1] = [60, 80]

print(x)

# Output: [0, 1, 2, 60, 80]
```

```
x = list(range(10))

print(x)

# Output: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

del x[4:6]

print(x)

# Output: [0, 1, 2, 3, 6, 7, 8, 9]
```

```
print([['#'] * 2 for x in range(2)])

# Output: [['#', '#'], ['#', '#']]

print([['#'] * 2] * 2)

# Output: [['#', '#'], ['#', '#']]
```

```
x= ['#'] * 2
y = []
for i in range(2):
  y.append(x)
  print(y)
```

Output:

```
['#', '#']

['#', '#'], ['#', '#']]
```

```
x = ['A', 'ABC', 'AB', 'ABCD']

print(sorted(x))

# Output: ['A', 'AB', 'ABC', 'ABCD']

print(sorted(x, key=len))

# Output: ['A', 'AB', 'ABC', 'ABCD']

print(sorted(x, key=len, reverse=True))

# Output: ['ABCD', 'ABC', 'AB', 'A']
```

```
from collections import deque

print(deque(range(5), maxlen=5))

# Output: deque([0, 1, 2, 3, 4], maxlen=5)
```

```
x = [15, 10, '1', 5, '123', '1234', 3, 45, '12', 21]

print(sorted(x, key=int))

# Output: ['1', 3, 5, 10, '12', 15, 21, 45, '123', '1234']

print(sorted(x, key=str))

# Output: ['1', 10, '12', '123', '1234', 15, 21, 3, 45, 5]
```

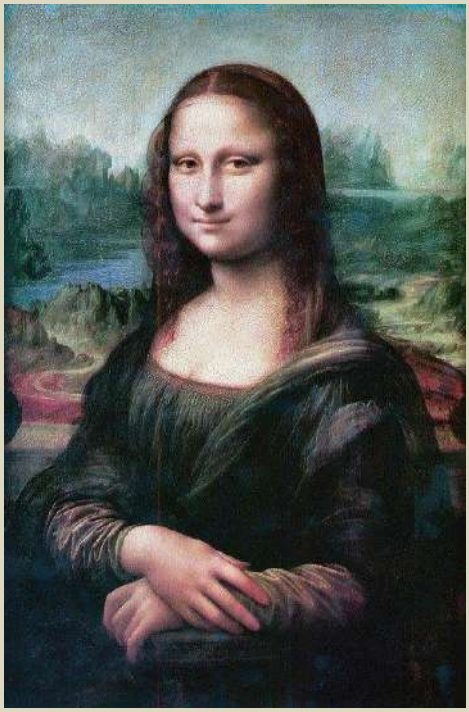
```
x= ['alan', 'albert', 'alan', 'albert']

print(set(x)) # Output: {'albert', 'alan'}

print(list(set(x))) # Output: {'albert', 'alan'}
```

```
print(frozenset(range(5))) # Output: frozenset({0, 1, 2, 3, 4})


# List a set of Latin-1 characters with codes from 100 to 150
# that have the letter 'A' in their Unicode names

from unicodedata import name

print({chr(x) for x in range(100, 150) if 'A' in name(chr(x),'')})

# Output:

{'l', 'y', 'i', '|', '}', 'r', 'g', 'h', '{', 'u', 't', 'p',

'n', 'w', 'q', 'x', 'o', 'j', 'm', 'd', 'f', 'e', 's', 'v', 'z', 'k'}
```

```
# install PyPDF2 using the command: pip install PyPDF2

import PyPDF2

i = open('1.pdf', 'rb')

x = PyPDF2.PdfFileReader(i)

print("The numbers of pages in a PDF file: ", x.numPages)

i.close()

# Output: The number of pages in a PDF file: 33
```

```
import os

print(os.listdir(b'.'))

# Output: ['1.txt', '1.pdf','names.csv', 'test.py']

print(os.listdir(b'.'))

# Output: [b'1.txt', b'1.pdf', b'names.csv', b'test.py']
```

```python
# python program to open an image

from PIL import Image
x = Image.open(r"C:\Users\Manju\1.png")
x.show()

# Output:
```



1.png

```python
from operator import methodcaller

print(methodcaller('upper')('albert')) # Output: ALBERT

print(methodcaller('lower')('ALBERT')) # Output: albert
```

```python
from operator import mul

from functools import partial

print(partial(mul, 7)(8)) # Output: 56
```

```
from operator import add

from functools import partial

print(partial(add, 7)(8)) # Output: 15
```

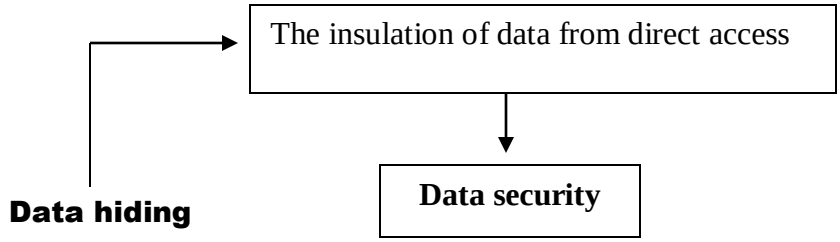
```
import math

print("The factorial of 6 is:", math.factorial(6))

# Output: The factorial of 6 is: 720
```

```
x = [14, [15, 16], (17, 18, 19)]

y = list(x)

print(y == x) # Output: True

print(y is x) # Output: False
```

```
x = {'Name': 'Albert', 'Died': 1955}

y = x

print(y is x) # Output: True

y['born'] = 1979

print(x) # Output: {'Name': 'Albert', 'Died': 1955, 'born': 1979}
```

```
def f(x, y):
   x += y
   return x


x = [12, 13]

y = [14, 15]

print(f(x, y))


# Output: [12, 13, 14, 15]
```

```
x = (21, 22, 23)

y = tuple(x)

print(y is x) # Output: True

z = x[:]

print(z is x) # Output: True
```



After providing proper username and password only, we can access our Account information.

Abstraction

Hide internal implementation and just highlight the set of services

Encapsulation	Data hiding
Deals with hiding the complexity of a program	Deals with the security of data in a program
Focuses on wrapping the complex data in order to present a simpler view for the user	Focuses on restricting the use of data − intending to assure the data security.

Encapsulation = Datahiding + Abstraction

	private	protected
Accessible from own class	✓	✓
Accessible from derived class	×	✓

Advantages of abstraction:

- Reduces Complexity
- Avoid Code Duplication

- Eases the burden of maintenance
- Increase Security and confidentiality

Single level Inheritance:

2 classes: A and B

Class B inherits from class A

Multi-level inheritance:

3 classes: A, B and C

Class C inherits from class B **while Class B** inherits from class A

Multiple Inheritance:

3 classes: A, B and C

C inherits from both class A and B

Hierarchical Inheritance:

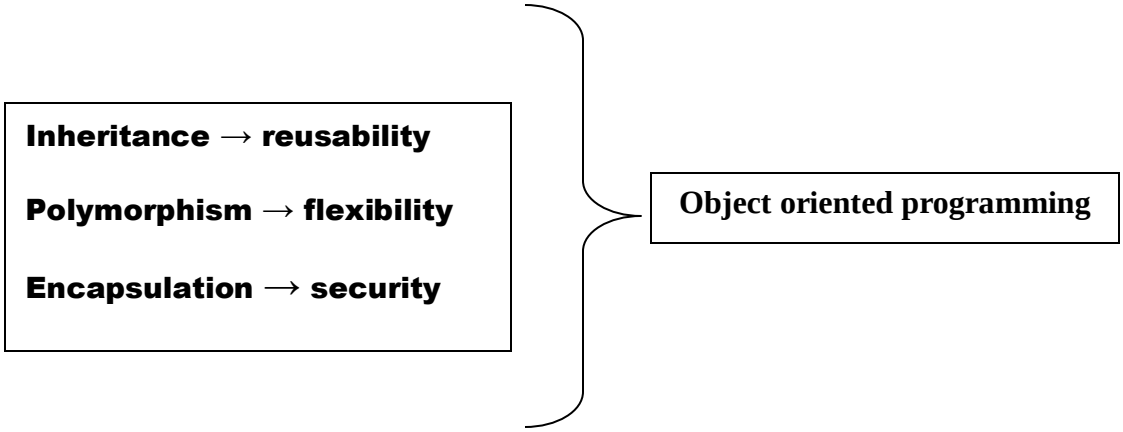
3 classes: A, B and C

Class B and C inherits from class A

Hybrid Inheritance:

4 classes: A, B, C and D

B and C inherits from A **while D inherits** from B and C

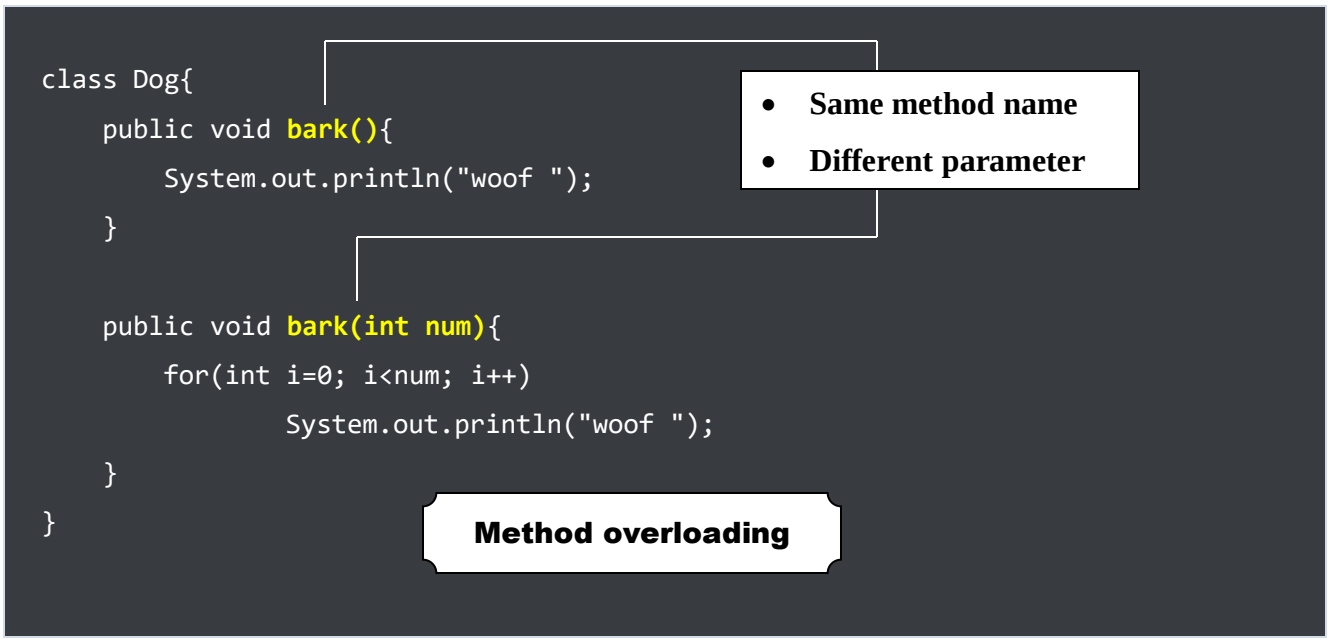


Method overloading	Method overriding
Compile time polymorphism	Run time polymorphism
may or may not require inheritance	**always need inheritance**

```
class Dog{
    public void bark(){
        System.out.println("woof ");
    }
}
class Hound extends Dog{
    public void sniff(){
        System.out.println("sniff ");
    }

    public void bark(){
        System.out.println("bowl");
    }
}
```

- **Same method name**
- **Same parameter**

Method overriding

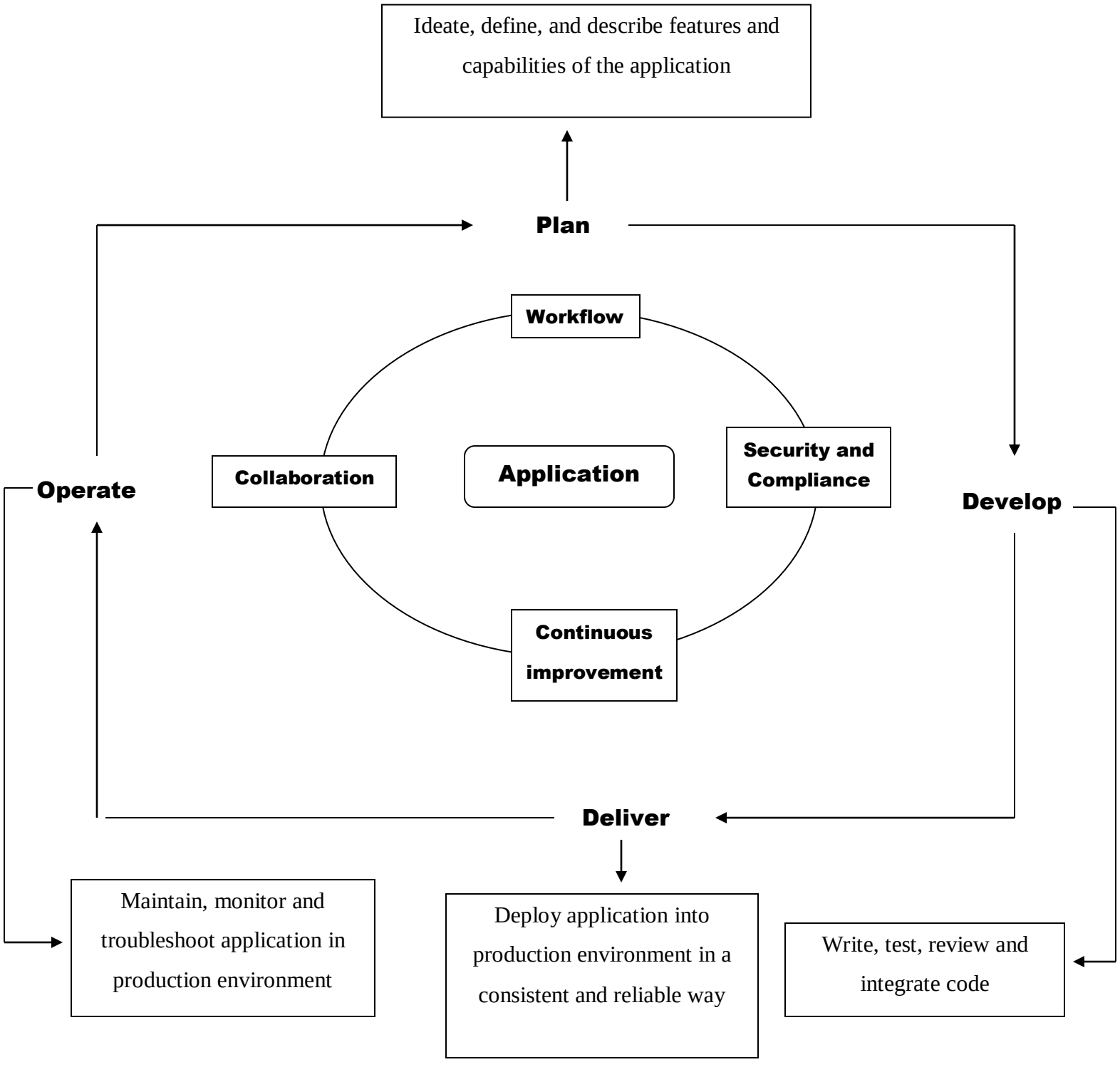


String	String buffer	String builder
Immutable (**once created cannot be changed**)	**Mutable** (**once created can be changed**)	**Mutable** (**once created can be changed**)
Consumes more memory	Consumes less memory	Consumes less memory
Not thread safe	Thread safe	Not thread safe
Synchronized (✓)	Synchronized (✓)	Synchronized (✖)
Very high performance (**Fast**)	Moderate performance (**Slow**)	Very high performance (**Fast**)

Constructor	Method
A Constructor's basic function is to create an instance of a class	A Method's basic function is to execute java code
Have no return type	Have any valid return type or no return type
Have the same name as their class	Have the name other than the class name
Constructors will start with a uppercase letter	Methods will start with a lowercase letter

DevOps isn't any single person's job. It's everyone's job.

Christophe Capel



Development (Software engineering) + Quality assurance + operations = DevOps

Development (Software engineering) + Quality assurance + operations + Security

DevSecOps

- Communication
- Collaboration
- Integration

Dev

(Software releases and updates)

Ops

(Reliability, performance and scaling)

Sec

(Confidentiality, availability and integrity)

Agile

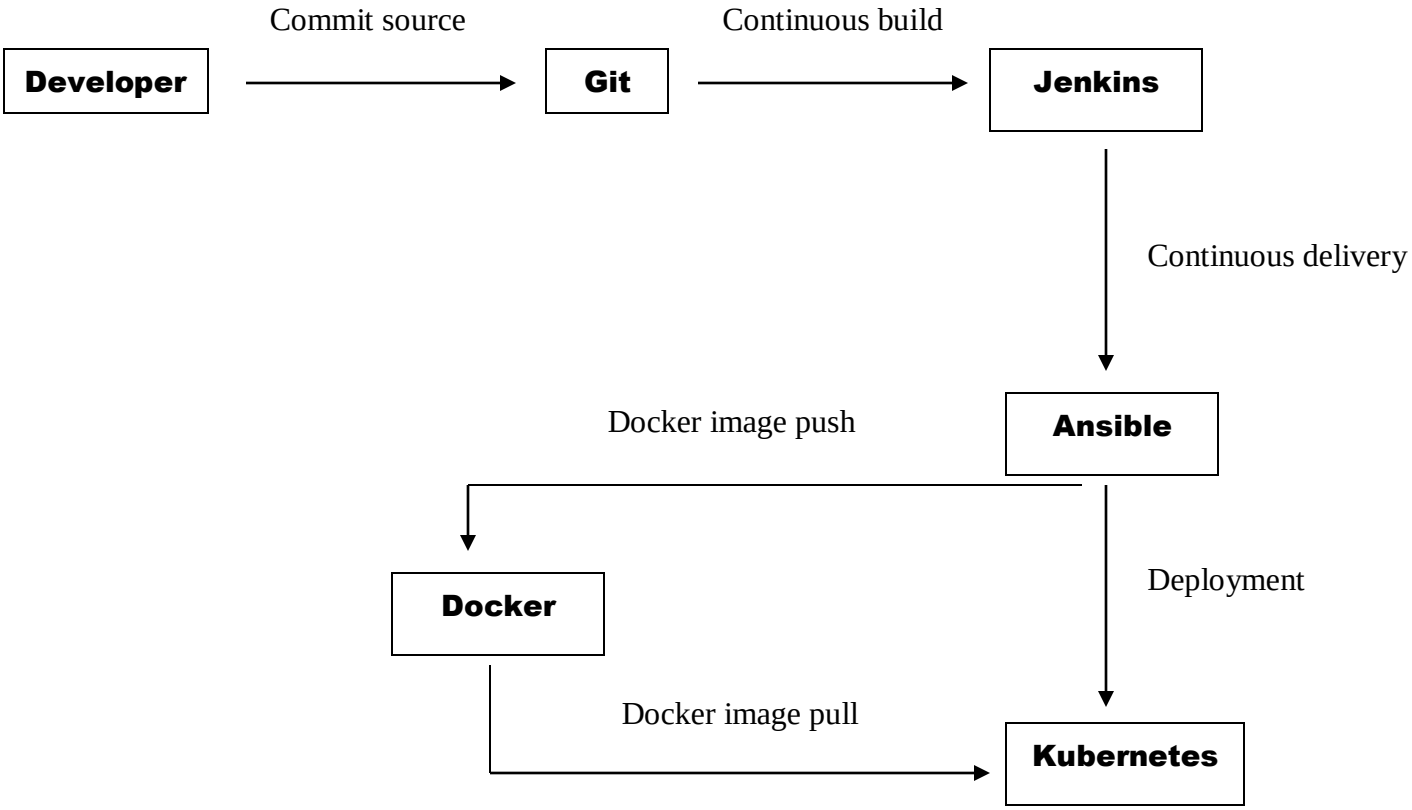
Focuses on processes highlighting change while accelerating delivery

CI/CD

Focuses on software-defined life cycles highlighting tools that emphasize automation

DevOps

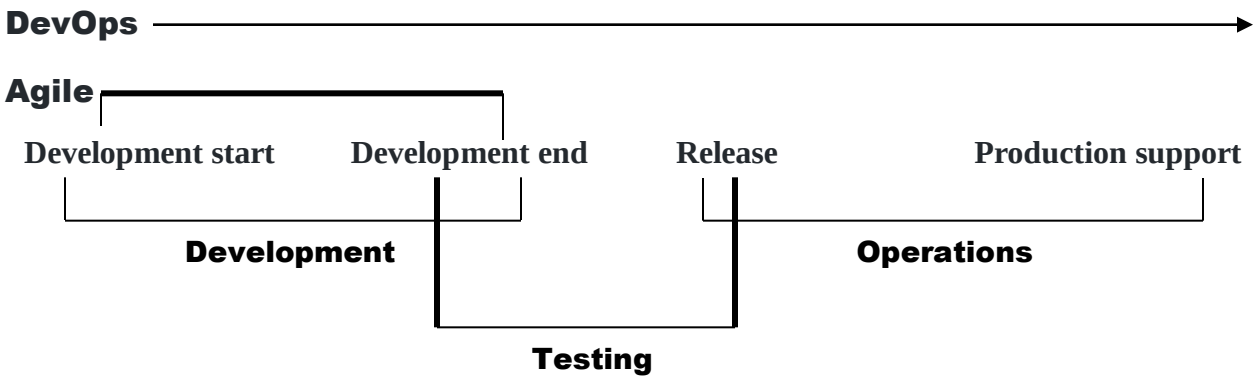
Focuses on culture highlighting roles that emphasize responsiveness



DevOps Flow

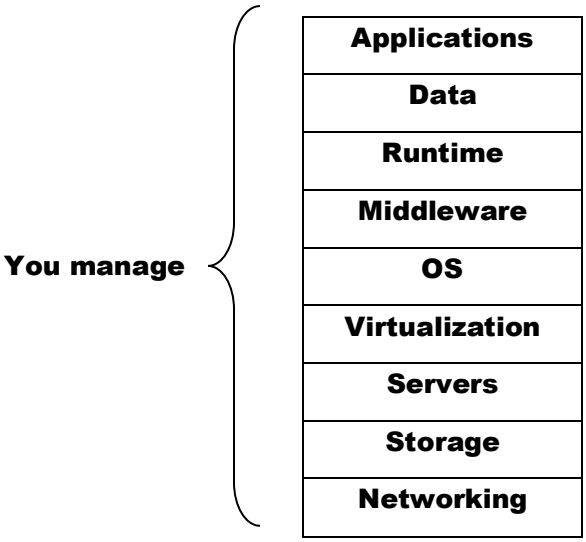
DevOps Tools

Git	Version Control System tool
Jenkins	Continuous Integration tool
Selenium	Continuous Testing tool
Puppet, Chef, Ansible	Configuration Management and Deployment tools
Nagios	Continuous Monitoring tool
Docker	Containerization tool

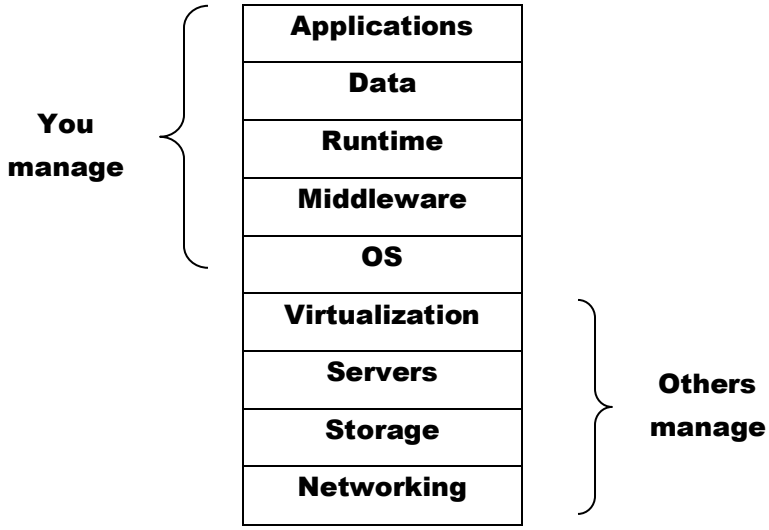


Cloud Service:

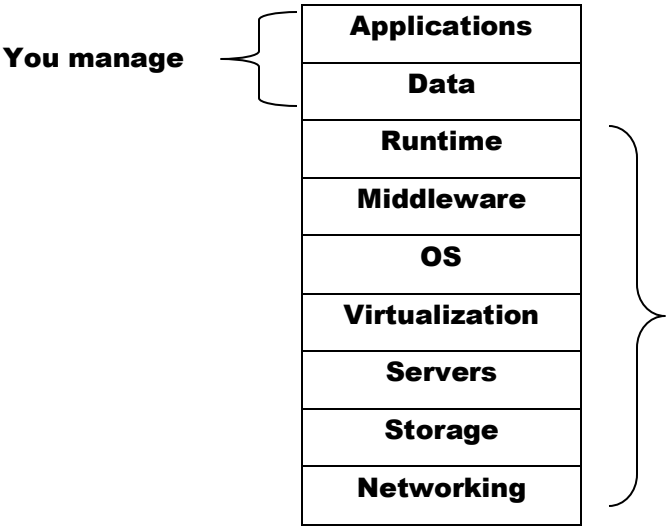
On-premise



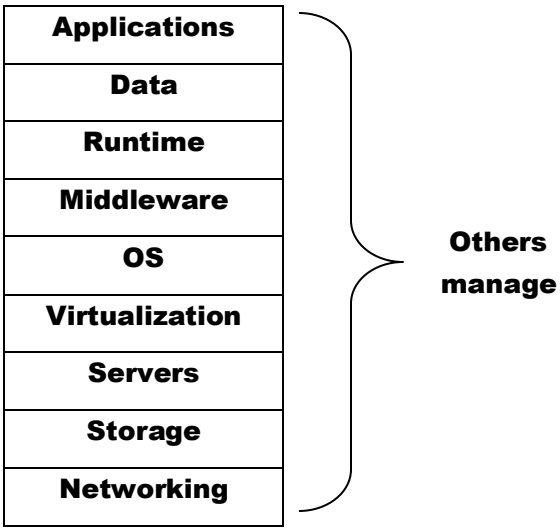
Infrastructure as a service

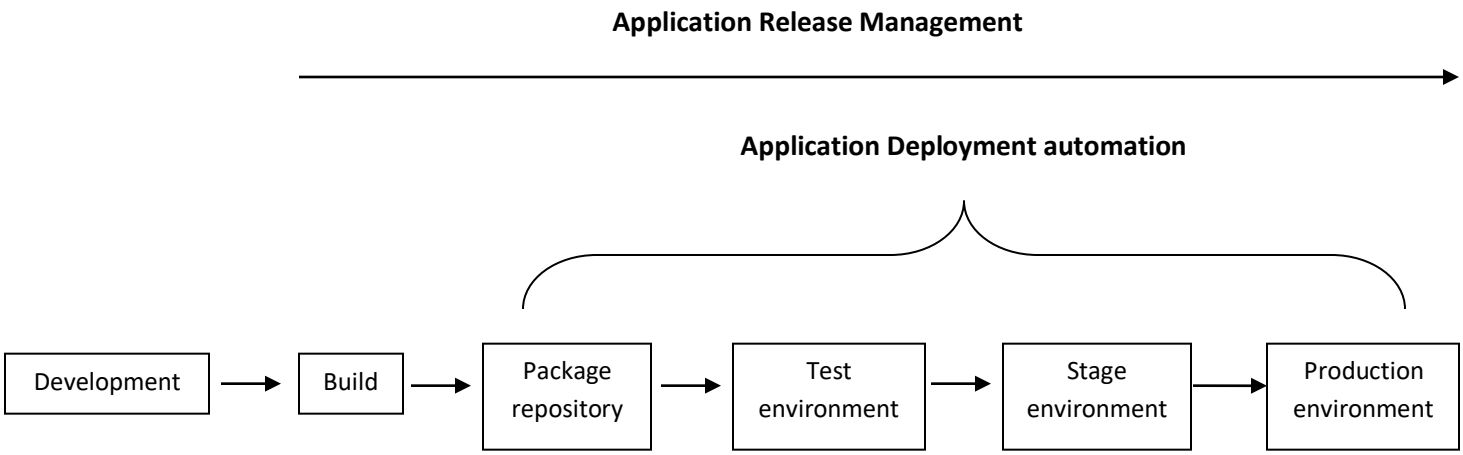


Platform as a service



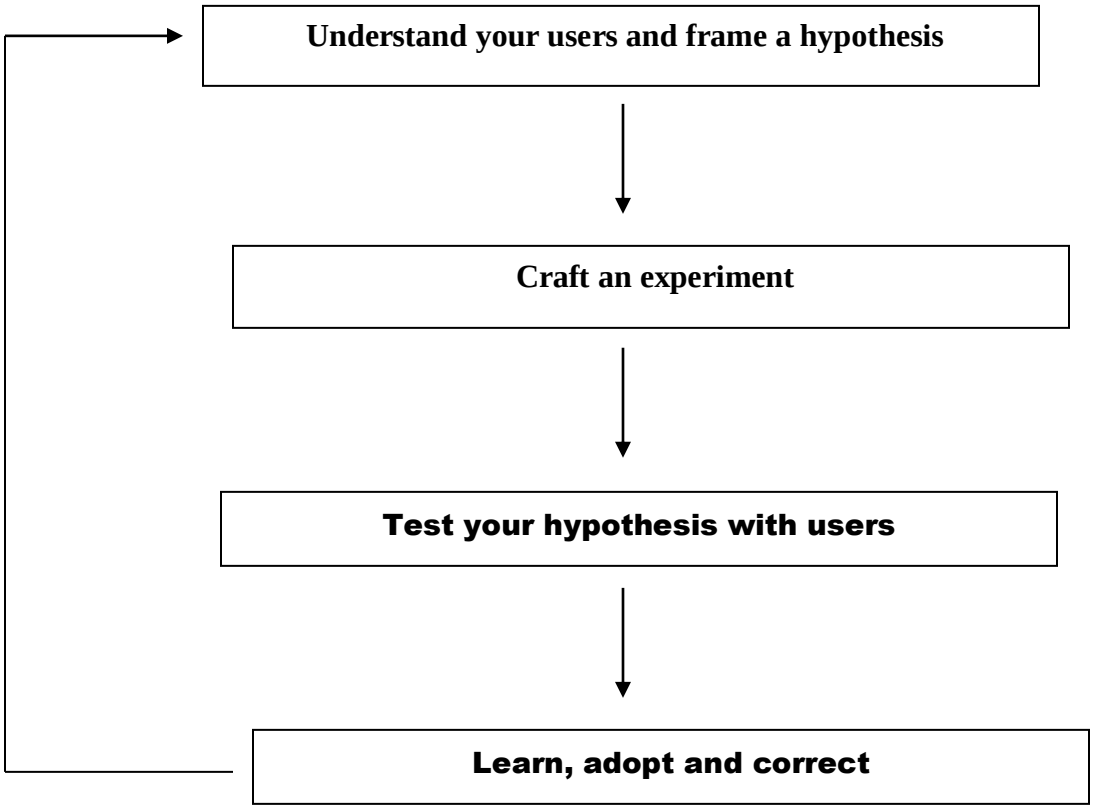
Software as a service





Stages of a typical DevOps delivery pipeline

How to Frame a Hypothesis?



```
# Basic Hello World program written in Groovy

class MyClass {
    static void main(String[] args) {

        println('Hello World');

    }
}
```

6 Build Phases in Maven:

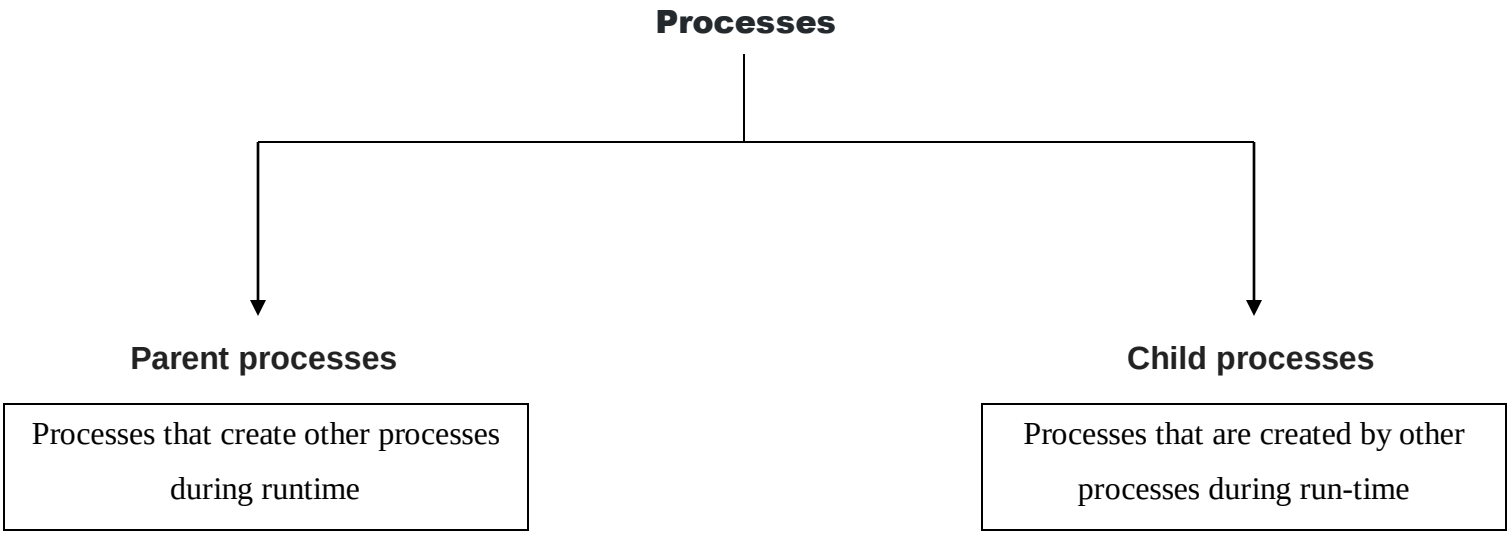
Validate → Compile → Test → Package → Install → Deploy

Command:

```
ifstat
```

Description:

Prints network interface statistics



Command:

```
pidof bash
```

Description:

Display the process IDs of a specific running program (Bash**)**

```
# Display the process ID of the current shell

echo $$

# Display the parent process ID of the current shell

echo $PPID
```

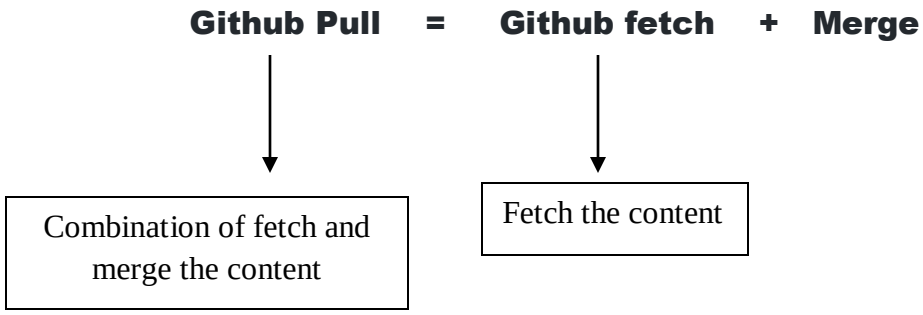
Command:

```
ps aux  | awk '{print $6/1024 " MBtt" $11}'  | sort -n
```

Description:

Display a list of most memory consuming processes

Github Fetch	Github Pull
Fetches the required information only to local repository	Fetches the required information not only to local repository but also to the workspace that you are currently working in



Docker registry	Docker repository
Service for hosting and distributing docker images	Collection of related Docker images

Selenium supports 2 types of testing:

- **Regression Testing** → retesting a product around an area where a bug was fixed.
- **Functional Testing** → testing of software features (functional points) individually.

Command:

```
ps au
```

```
ps axu
```

Description:

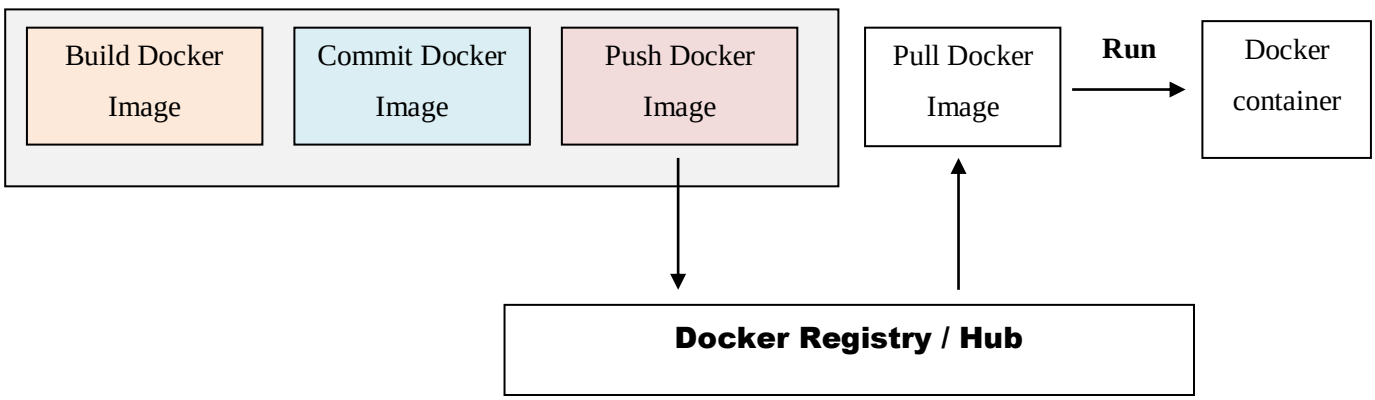
Display all processes in BSD format

Command:

```
ps aux
```

Description:

Display all processes and their status and resource usage



```
# Takes in integer x and returns the binary representation of x in a string format

x = 15

print(bin(x))
```

```
0b1111
```

```
# Takes in integer x and returns the octal representation of x in a string format

x = 10

print(oct(x))
```

```
0o12
```

```
# Takes in integer x and returns the hexadecimal representation of x in a string format

x = 100

print(hex(x))
```

```
0x64
```

```
x="python\nlanguage"
print(x)
```

```
python
language
```

```
x="python\tlanguage"
print(x)
```

```
python   language
```

Operator	Meaning	Example
in	True if value found in the sequence	**5 in x**
not in	True if value is not found in the sequence	**5 not in x**

```
list_1=[1,2,3,4,5]
list_2=[6,7,8,9]
for item in list_1:
   if item in list_2:
     print("overlapping")
else:
     print("not overlapping")
```

Output on the screen:

```
not overlapping
```

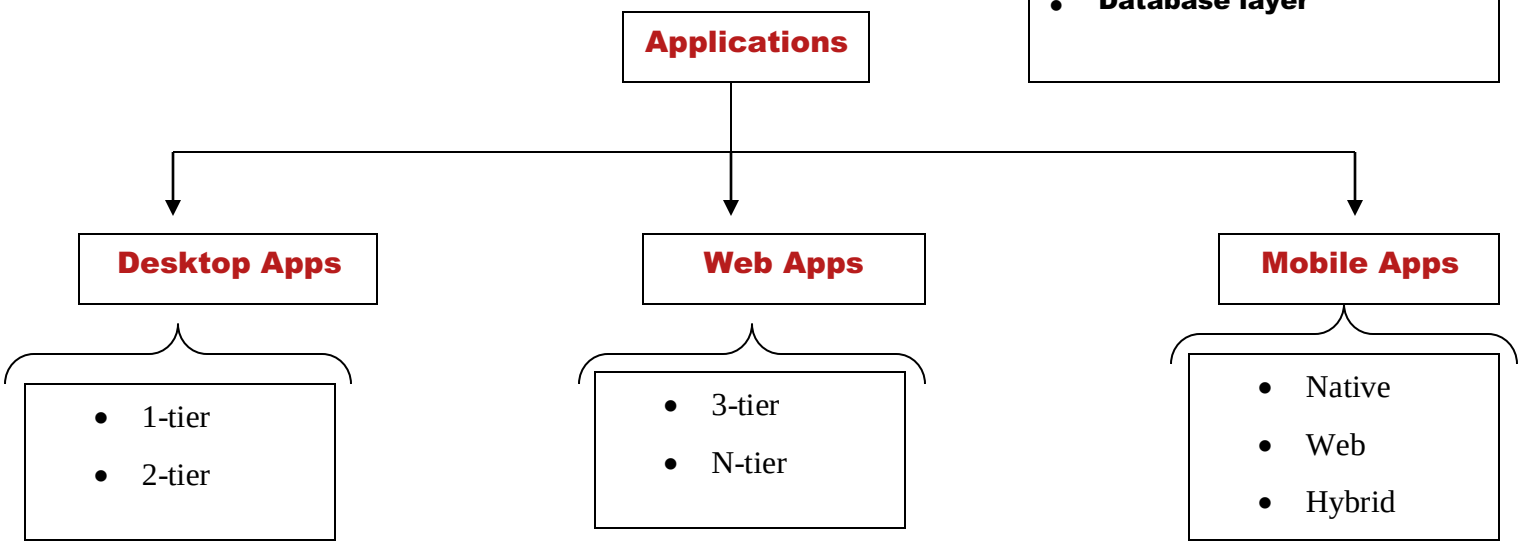
```
x = 24

y = 20

list = [10, 20, 30, 40, 50 ];


if ( x not in list ):

   print("x is NOT present in given list")

else:

   print("x is present in given list")


if ( y in list ):

   print("y is present in given list")

else:

   print("y is NOT present in given list")
```

Output on the screen:

```
x is NOT present in given list

y is present in given list
```

3-tier Applications consists of :

- **Web Layer**
- **Application layer**
- **Database layer**



Java Design Patterns:

- **Creational design patterns** → Design patterns that deal with the way of creating objects.
- **Structural design patterns** → Design patterns that deal with how classes and objects can be composed to form larger structures.
- **Behavioral design patterns** → Design patterns that deal with algorithms and the assignment of responsibilities between objects.
- **JEE Design Patterns** → Design patterns that deal with providing solutions to the Java EE-based software applications and frameworks.

- **Automation** → Reduces execution time
- **Robotic Process Automation** → Reduces the number of people and automate task just like a human being

A **Machine Learning model** is said to learn from:

- Actual problem that need to be solved
- Knowledge acquired from experience
- Performance as per expectation

Improves with experience

- Robotic Desktop automation
- Robotic Process automation

Process driven

- Machine learning
- Artificial intelligence

Data driven

Despite being a well-liked and effective programming language, Python has a drawback in the form of a slow execution rate. Python is an interpreted language; hence its execution is slower than that of compiled languages.

Robotic Process Automation:

Identify → Analyze → Design → Develop → Test → Implement

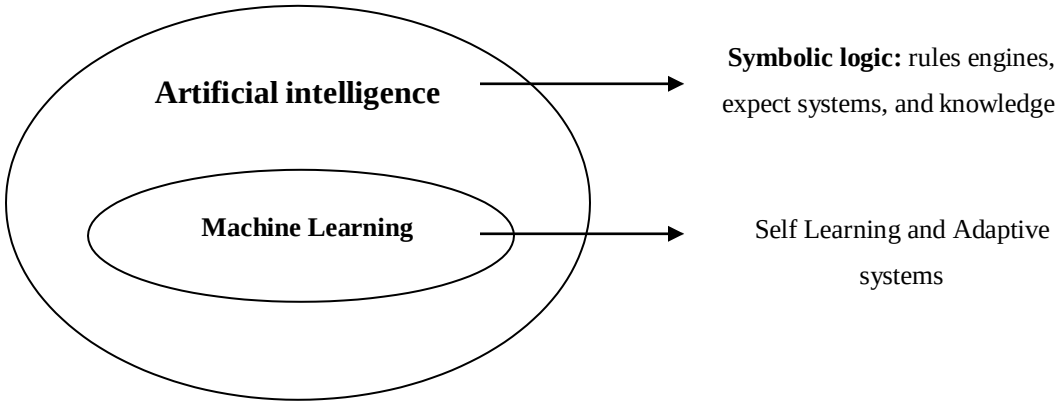
Robotic process automation + Cognitive automation + Workforce Analytics → Intelligent Automation

	Big Data Analytics	Data Science
Type of Data Processed	Structured	• Structured Data • Unstructured Data • Semi-structured Data

3 Components of Cloud Computing:

- Software-as-a-Service **(SaaS)**
- Infrastructure-as-a-Service **(IaaS)**
- Platform-as-a-Service **(PaaS)**

- **Label Encoding:** A method for transforming labels of categorical data into numeric form so they may be processed by a machine learning model
- **Eager Learning:** Putting more effort into training and less effort into prediction
- **Lazy Learning:** Putting more effort into prediction and less effort into training



	Data Mining	Statistics
Methodology	Inductive	Deductive
Variables	Large	Small
Used for	Exploration	Confirmation

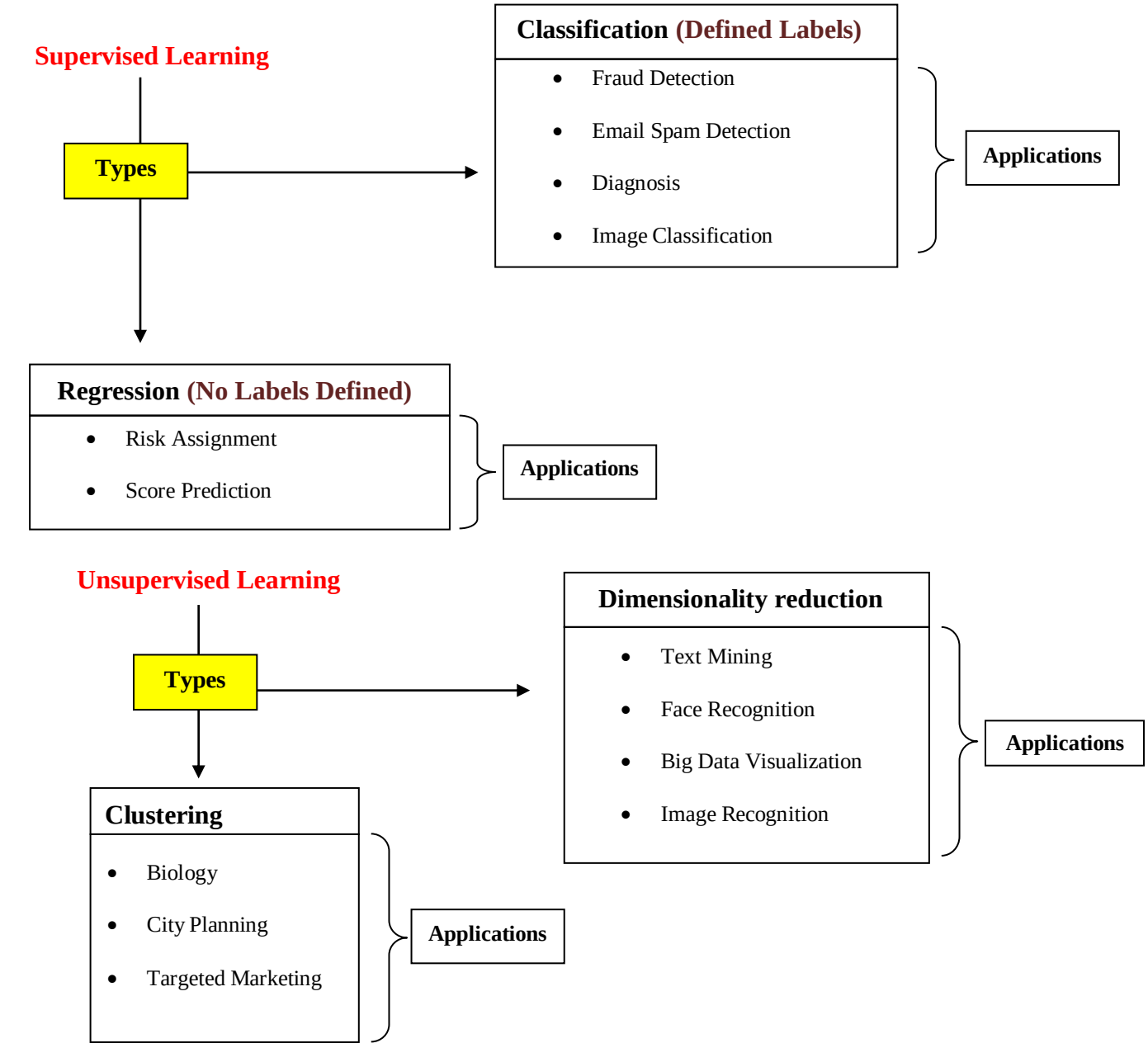
Simple regression model:

In this regression model, a single, univariate data feature is used to derive the predictions

Multiple regression model:

In this regression model, multiple features of the data are used to derive the predictions

Classification decision trees	Regression decision trees
The decision variable is categorical	The decision variable is continuous



Applications of Reinforcement Learning

An approach to **Machine Learning** whereby a software program engages with the environment and learns how to react with it

- Gaming
- Finance Sector
- Manufacturing
- Inventory
- Management
- Robot Navigation

Keras	Tensorflow
A simple Python library for Deep Learning	A set of libraries for Machine Learning
Easy to debug	Complex to debug
Used for small datasets	Used for massive datasets

A freely available distributed database system for managing and storing massive volumes of data across commodity servers

Disadvantages of Cassandra:

- No support to Data Integration
- Large outputs must be physically paged

Mean shift clustering algorithm:

A **Centroid-based technique** that supports numerous unsupervised learning application scenarios. It is one of the best algorithms for computer vision and image processing. In essence, it iteratively distributes the **data points** to the clusters by moving them toward the **cluster Centroid**, which has the maximum density of **data points**

Manual Testing	Automated Testing
Less accurate	Highly reliable
Needs lots of time	Very fast
Test cases are run once or twice	Test cases are run repeatedly
Human intervention	No Human intervention

The main phases of risk analysis are:

The process of identifying, analyzing, evaluating and treating the risks that could cause the failure of the project

- Identification of Risk
- Analyzing the Risk
- Evaluating the Risk
- Treat the Risk
- Review the Risk

- **SQL** → A query language that manages relational database management system
- **MYSQL** → A relational database management system that uses SQL

Data Science Life Cycle

Step 1: Define the Problem

Step 2: Collect the Data

Step 3: Process the Data

Step 4: Explore the Data

Step 5: Analyze the Data

Step 6: Consolidate the Result

Differential privacy

A set of **frameworks** and practices that assist in getting the useful information from the databases containing **individual's personal information** − without revealing the personal identification about individuals

Neuromorphic Computing: Computing that imitate closely the human brain or the way the human brain operates

Tensorflow architecture consists of three parts:

Open-source software library that provide an uncomplicated approach to solve complex calculations of machine learning with the help of graphs

- Data preprocessing (**Unstructured raw data** —**transformed into**→ **Structured useful and efficient data**)
- Model building
- Model training and evaluation

Quantum Computer: A machine that harnesses the properties of quantum mechanics to store data and perform computations

9 Interesting Python Facts

In python we can return multiple values:

```
def XYZ():
    p = 3
    q = 2
    return p, q

a, b = XYZ()
print(a, b)
```

Output on the screen:

```
3 2
```

Data fusion

The process of combing information from numerous data sources to produce more appropriate, authentic and useful information than that provided by any individual data source

---

Allows Negative Indexing:

```
my_list = ['apple', 'orange', 'grapes']
print(my_list[-2])
```

Output on the screen:

```
orange
```

Data consolidation

Collection and integration of information from multiple data sources into a single repository

---

Combine Multiple Strings:

```
my_list = ['I', 'Love', 'Python']
print(''.join(my_list))
```

Output on the screen:

```
ILovePython
```

Data federation

A technology that allows 2 or more databases to function as one

---

We can swap two objects in Python:

```
a = 3
b = 2

print('Before Swapping')
print(a, b)

a, b = b, a
print('After Swapping')
print(a, b)
```

Output on the screen:

```
Before Swapping

3 2

After Swapping

2 3
```

Polymorphic malware

Malicious software that gradually changes its identifiable features in order to make it difficult for anti-malware software to prevent, detect and remove it

---

We can know about Python version:

```
import sys
print("My Python version Number: {}".format(sys.version))
```

Output on the screen:

```
My Python version Number: 3.7.3 (default, Mar 27 2019, 17:13:21) [MSC v.1915 64 bit (AMD64)]
```

We can Store all values of List in new separate variables:

```
x = [1, 2, 3]
a, b, c = x
print(a)
print(b)
print(c)
```

Output on the screen:

```
1

2

3
```

AI Transparency

Understanding, explaining and interpreting how AI systems make decisions and use data

We can convert nested list into one list:

```
import itertools
x = [[1, 2], [3, 4], [5, 6]]
print(list(itertools.chain.from_iterable(x)))
```

Output on the screen:

```
[1, 2, 3, 4, 5, 6]
```

Risk-based testing: A method of software testing that is based on the process of analyzing the risks associated with the testing Project

We can transpose a Matrix:

```
import numpy as np
x = np.matrix('[5, 1; 14, 2]')
y = x.transpose()
print(y)
```

Output on the screen:

```
[[ 5 14]
 [ 1  2]]
```

Data Privacy	Data protection
↓	↓
Defines who has access to information	Legal control over access to and use of information

We can create small anonymous function:

```
x = lambda a, b, c : a + b + c
print(x(5, 6, 3))
```

Output on the screen:

```
14
```

AI Traceability

Identifying and tracing the processes involved in the production of an AI model

Responsible AI

{The ethical development and use of AI systems}

AI which is explainable, monitorable, reproducible, secure, human-centered, unbiased and justifiable

C++ Exercises

- **Intra-personal intelligence:** The ability to interpret and recognize our feelings and emotions
- **Interpersonal intelligence:** The ability to interpret and recognize other people's feelings and emotions

Question 1

Question:

Auditory Learning: Learning by hearing and listening

Write a program to print Hello World!.

Solution:

```
#include<iostream>
int main()
{
std::cout<<"Hello World!";
return 0;
}
```

Heuristic Knowledge: Practice knowledge, precise reasoning, and one's capability to analyze and guess

---

Solution:

```
#include<iostream>
using namespace std;
int main()
{
cout<<"Hello World!";
return 0;
}
```

Phonology: The study of sound patterns found in languages

Question 2

Question:

Write a program to find the area of a circle.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
float r, area;
cout<<"Enter any number:";
cin>>r;
area = 3.14 * r * r;
cout<<"The area of the circle = "<< area;
return 0;
}
```

Data Stack: A bunch of technologies and services that an enterprise organization use to store, manage and access information

---

Question 3

Hypothesis Testing

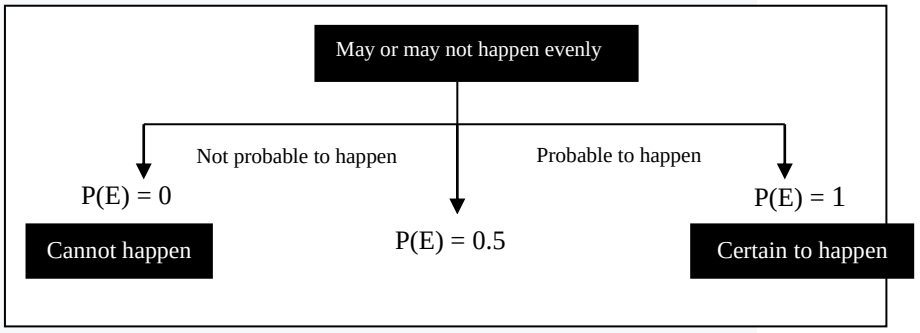
The process of checking whether a hypothesis is true or not

Question:

Write a program to find the sum of two numbers.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
float a, b, sum;
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
sum = a+ b;
cout<<"The sum of a and b = "<< sum;
return 0;
}
```



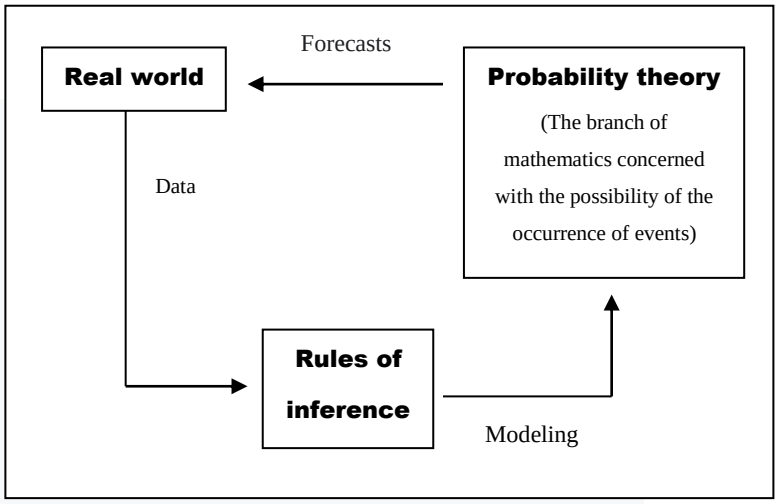
Question 4

Question:

Write a program to find the square of a number.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int a, b;
a=2;
b = a * a;
cout<<"The square of a = "<< b;
return 0;
}
```



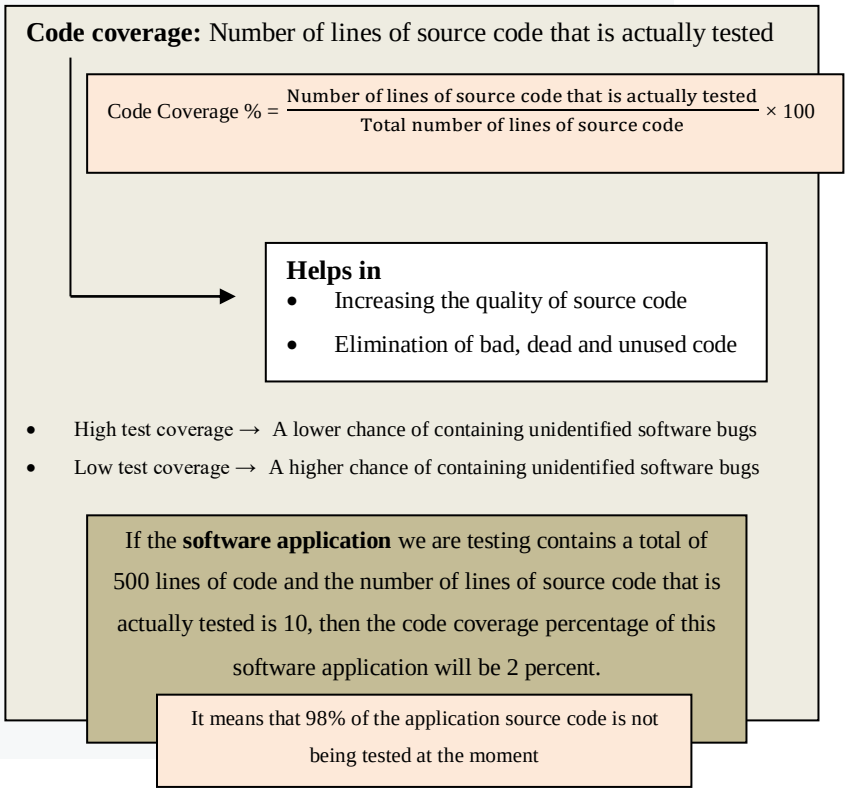
Question 5

Question:

Write a program to find the greatest of two numbers.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int a, b;
a = 2;
b = 3;
if(a>b)
{
cout<<"a is greater than b";
}
else
{
cout<<"b is greater than a";
}
return 0;
}
```



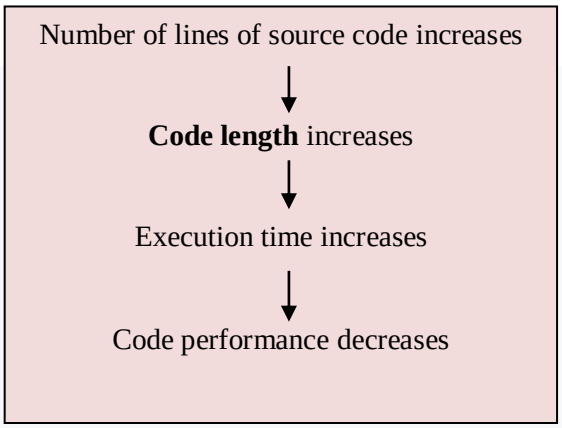
Code coverage is a measure of how much of application source code is tested while test coverage is a measure of how well the software application is tested when it comes to its requirements

Question 6

Question:

Write a program to print the average of the elements in the array.

Solution:




```
#include<iostream>
using namespace std;
int main()
{
int i, avg, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num [i];
avg = sum/5;
cout<<"Sum of the Elements in the array = "<< sum <<endl;
cout<<"Average of the elements in the array= "<< avg<<endl;
return 0;
}
```

Question 7

Question:

Write a program such that a Switch (case) allows to make a decision from the number of choices, i.e., from the number of cases.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
char ch;
cout<<"Enter any character:";
cin>>ch;
switch(ch)
{
case 'R':
cout<<"Red";
break;
case 'W':
cout<<"White";
break;
case 'Y':
cout<<"Yellow";
break;
case 'G':
cout<<"Green";
break;
default:
cout<<"Error";
break;
}
return 0;
}
```

Extreme Programming

Software development methodology

specifically designed to improve software quality and increase customer satisfaction by allowing development team to respond to changing customer requirements

Fail Fast Philosophy

If it is possible to learn from failure then the quickly the failure occurs, the quickly the learning begins

Vertical scaling	Horizontal scaling
Increasing instance size	Adding more instances

Degauss:

A methodology for de-magnetizing computer monitors

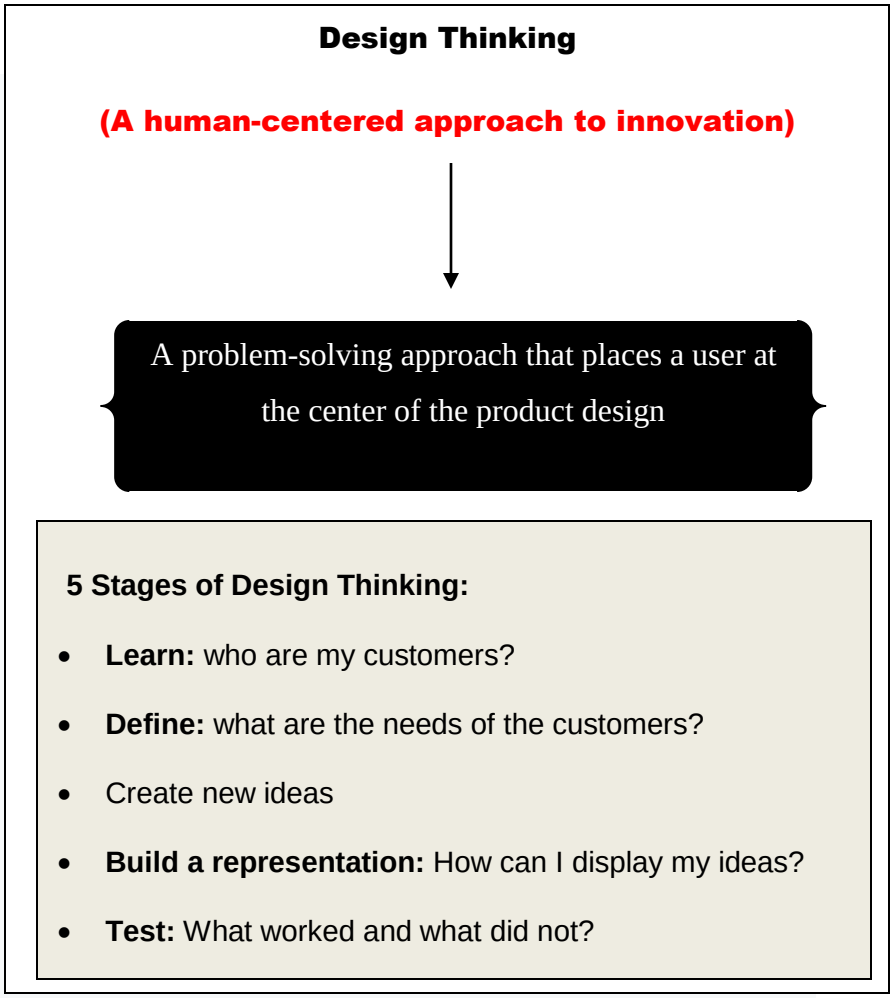
Question 8

Question:

Write a program to find the greatest of two numbers using pointers.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int x, y, *p, *q;
cout<<"Enter any integer:";
cin>> x;
cout<<"Enter any integer:";
cin>> y;
p  =  &x;
q = &y;
if(*p>*q)
{
cout<<"x is greater than y";
}
else
{
cout<<"y is greater than x";
}
return 0;
}
```



Question 9

Question:

Write a program to print the address of x and the value assigned to x.

Solution:

Dockerizing: The process of packing, deploying and running software applications using Docker containers

```
#include<iostream>
using namespace std;
int main()
{
int x, *p;
cout<<"Enter any integer:";
cin>>x;
p = &x;
cout<<"The address of the variable x = "<< p<<endl;
cout<<"The value of the variable x = "<< *p<<endl;
return 0;
}
```

Question 10

Question:

Write a program to print the first 10 numbers starting from one together with their squares and cubes.

Solution:

Linting

(Pre-code review)

The automated checking of **application source code** for programmatic and dramatic errors

```
#include<iostream>
using namespace std;
int main()
{
int i;
for( i=1; i<=10; i++)
cout<<"number = "<< i <<" its square = "<< i*i <<" its cube = "<< i*i*i<< endl;
return 0;
}
```

Question 11

Question:

Write a program:

If you enter a character M

Output must be: ch = M.

- **File locking:** A mechanism that restricts access to a file
- **Minimal Viable Product:** A product with minimum basic features which is introduced in the market just to get the attention of the consumers

---

Solution:

Product backlog

A **prioritized list of tasks** that need to be finished during the development of new product or while upgrading existing product

```
#include<iostream>
using namespace std;
int main()
{
char M;
cout<<"Enter any character:";
cin>>M;
cout<<"ch= "<< M;
```

```
return 0;
}
```

Question 12

Question:

Write a program to print the multiplication table of a number.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int n, i;
cout<<"Enter any number:";
cin>>n;
for( i=1; i<=5; i++)
cout<< n <<" * "<< i <<" = "<< n*i <<endl;
return 0;
}
```

3 principles that every developer should be aware of:

YAGNI Principle

"You Aren't Gonna Need It"

↓

Implementation of only required functionalities

KISS Principle

"Keep It Simple, Stupid"

↓

Avoid unnecessary complexity in the code

DRY Principle

"Don't Repeat Yourself"

↓

Don't write the same code repeatedly

Value Proposition

A promise of value specified by a company that answers why **customers** should choose their products or services

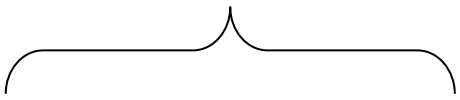
Question 13

Question:

Write a program to print the product of the first 10 digits.

Solution:

Blogosphere



All of the blogs or bloggers on the internet

```
#include<iostream>
using namespace std;
int main()
{
int i, product = 1;
for( i=1; i<=10; i++)
product = product * i;
cout<<"The product of the first 10 digits = " << product;
return 0;
}
```

Question 14

RSS

(Rich Site Summary)

An XML-based summary of a website

Question:

Write a program to print whether the given number is positive or negative.

Solution:

Netizen: A user of the internet

```
#include<iostream>
using namespace std;
int main()
{
int a;
a = -35;
if(a>0)
{
cout<<"Number is positive";
}
else
{
cout<<"Number is negative";
}
return 0;
}
```

Bits-Per-Second

A measure of the rate at which data is sent from one location to another

CDMA

(Code Division Multiple Access)

A wireless data and voice communication protocol

Question 15

Question:

Write a program to check the equivalence of two numbers.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int x, y;
cout<<"Enter any number:";
```

Cyberspace: The entire set of information resources accessible via computer networks

```
cin>>x;
cout<<"Enter any number:";
cin>>y;
if(x-y==0)
{
cout<<"The two numbers are equivalent";
}
else
{
cout<<"The two numbers are not equivalent";
}
return 0;
}
```

Imputation: The process of filling in **missing data values** in a dataset

Devalue the performance of data analytics leading to a wrong prediction

---

Question 16

Question:

Write a program to print the remainder of two numbers.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int a, b, c;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
```

Data-directed decision making: The process of using data to support making crucial business decisions

```
c = a % b;
cout<<"The remainder of a and b = "<< c;
return 0;
}
```

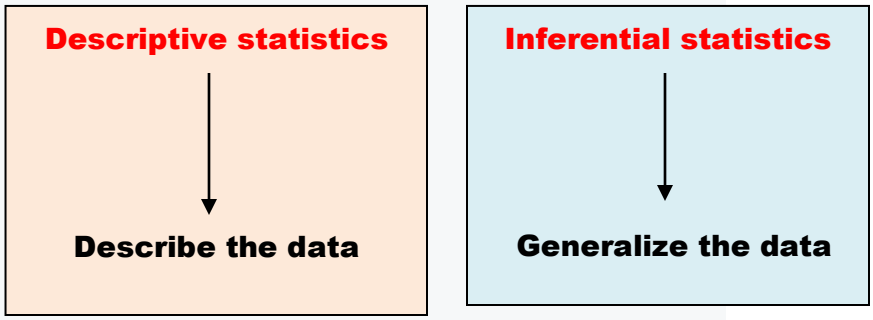
Question 17

Question:

Write a program to print the given number is even or odd.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int a;
cout<<"Enter any number:";
cin>>a;
if(a%2 = = 0)
{
cout<<"The number is even";
}
else
{
cout<<"The number is odd";
}
return 0;
}
```



Question 18

Question:

Write a program to print the characters from A to Z.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
char a = 'A';
while (a<='Z')
{
cout<<" \n"<< a++;
}
return 0;
}
```

Bounce rate:

The percentage of visitors to a website who leave without browsing around

---

Question 19

Question:

Write a program to find the incremented and decremented values of two numbers.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
cout<<"The incremented value of a = "<< c << endl;
cout<<"The incremented value of b = "<< d << endl;
cout<<"The decremented value of a = "<< e << endl;
cout<<"The decremented value of b = "<< f << endl;
return 0;
}
```

- 1 Brontobyte = 10^{27} bytes
- 1 Yottabyte = 10^{24} bytes
- 1 Exabyte = 10^{18} bytes
- 1 Zettabyte = 10^{21} bytes
- 1 Terabyte = 10^{12} bytes
- 1 Petabyte = 10^{15} bytes
- 1 Gigabyte =10^{9} bytes
- 1 Megabyte =10^{6} bytes

Question 20

Question:

Write a program to calculate the simple interest.

Solution:

```
#include<iostream>
```

```
using namespace std;
int main()
{
int P,T, R, SI;
cout<<"Enter principal amount:";
cin>>P;
cout<<"Enter time:";
cin>>T;
cout<<"Enter rate of interest:";
cin>>R;
SI = P*T*R/100;
cout<<"the simple interest = "<<SI;
return 0;
}
```

Correlation analysis: A method of statistical data analysis that determines whether there is a negative or positive relationship between variables

A correlation of + 0.9 indicates a strong positive relationship between 2 variables − whereas a correlation of −0.2 indicates a weak negative relationship

---

Question 21

Question:

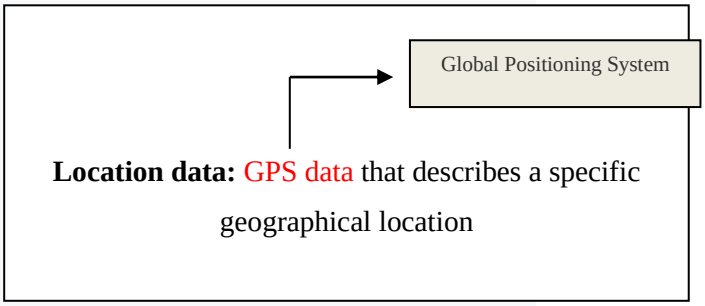
Write a program to Find the largest of three numbers.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int a, b, c;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
```

- **Failover:** Automatic workload transfer from a failed primary server to a backup server
- **Failback:** The process of returning back the work load to the primary server after the failure has been resolved

```
cin>>b;
cout<<"Enter any number:";
cin>>c;
if(a>b&&a>c)
{
cout<< a<<" is greater than "<< b<<" and "<<c;
}
else if (b>a&&b>c)
{
cout<< b<<" is greater than "<< a <<" and "<<c;
}
else
{
cout<< c<<" is greater than "<< b<<" and "<< a;
}
return 0;
}
```



Question 22

Question:

Write a program to print the factorial of the entered number.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int i, n, fact=1 ;
```

Multi-Dimensional Database: A database designed for data warehousing and online analytical processing applications that provide a multidimensional view of data

```
cout<<"Enter any number:";
cin>>n;
for(i=1; i<=n; i++)
fact = fact *i;
cout<<"\n Entered number is: "<< n;
cout<<"\n The factorial of the entered number "<< n <<" is: "<< fact;
return 0;
}
```

Object Database: A database system in which data is stored as objects

---

Question 23

Question:

Write a program to print the length of the entered string.

Solution:

```
#include<iostream>
#include<string.h>
using namespace std;
int main()
{
char ch[4];
cout<<"Enter any word: ";
cin>>ch;
cout<<"The length of the string = "<< strlen(ch);
return 0;
}
```

Optimization analysis

A method through which an enterprise organization improves the efficiency of a process by evaluating the optimization of the process under a variety of predefined constraints

---

Question 24

Question:

Write a program to print the ASCII value of the entered character.

Solution:

```
#include <iostream>
using namespace std;
int main()
{
 char c;
 cout << "Enter a character: ";
 cin >> c;
 cout << "ASCII Value of " << c << " is " << int(c);
 return 0;
}
```

Question 25

Question:

Write a program to check whether the entered character is a lower case letter or not.

Solution:

Data Analysis

Collect the Data → Analyze the Data → Visualize and share your findings

```
#include<iostream>
using namespace std;
int main()
{
char ch = 'a';
if(islower(ch))
cout<<"you have entered the lower case letter";
else
cout<<"you have entered the upper case letter";
return 0;
}
```

Decorators

↓

Functions which modify the functionality of other functions

Question 26

Question:

Write a program to check whether the entered character is a upper case letter or not.

Solution:

Data Infrastructure

A collection of servers and networks that enable

- data consumption
- data storage
- data sharing

```
#include<iostream>
using namespace std;
int main()
{
char ch = 'a';
if(isupper(ch))
cout<<"you have entered the upper case letter";
```

```
else
cout<<"you have entered the lower case letter";
return 0;
}
```

Question 27

Question:

Write a program to convert the lower case letter to upper case letter.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
char ch = 'a';
char b = toupper(ch);
cout<<" lower case letter "<<ch<<" is converted to upper case letter "<<b;
return 0;
}
```

Frame language: A technology used for knowledge representation in AI

Eager Learning	Lazy Learning
ML algorithm builds a model based on a training data before receiving a test data	**ML algorithm does not build a model immediately after receiving the training data rather waits until it receives the test data**

Question 28

Ebert test

Whether a computer-based synthesized voice can tell a joke with sufficient skill to make an audience laugh

Question:

Write a program to print the output:

Einstein [0] = E

Einstein [1] = I

Einstein [2] = N

Einstein [3] = S

Einstein [4] = T

Einstein [5] = E

Einstein [6] = I

Einstein [7] = N

Existential risk

AI progress could someday result in **human extinction** or some other irrecoverable global disaster

---

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int i;
char name [8] = {'E' , 'I', 'N', 'S', 'T', 'E', 'I', 'N'};
for(i=0; i<8; i++)
cout<<"Einstein ["<< i <<" ] = "<< name[i] << endl;
return 0;
}
```

Machine perception: The ability of a **machine to simulate** – the way that human beings perceive the world around them

---

Question 29

Error-driven learning

↓

The difference between the value predicted by ML model and the actual value is the prediction error and it is this error that drives learning

Question:

Write a program to print the output:

Name of the book = B

Price of the book = 135.00

Number of pages = 300

Edition of the book = 8

using structures.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
struct book {
char name;
float price;
int pages;
int edition;
};
struct book b1= {'B', 135.00, 300, 8};
cout<<"Name of the book = "<< b1.name<< endl;
cout<<"Price of the book = "<< b1.price<<endl;
cout<<"Number of pages = "<< b1.pages<<endl;
cout<<"Edition of the book = "<< b1.edition<< endl;
return 0;
}
```

Intelligence Amplification: The effective use of AI methods in enhancing human intelligence

Machine vision

↓

The ability of a machine to identify and evaluate images

Question 30

Question:

Write a program to find square of a number using functions.

Solution:

```
#include<iostream>
using namespace std;
int square();
int main()
{
int  answer;
answer = square();
cout<<"Square of the given number = "<< answer;
return 0;
}
int square()
{
int x;
cout<<"Enter any integer:";
cin>>x;
return x*x;
}
```

- **Correlation** → how strongly two variables are related
- **Covariance** → how the two variables differ

Case-based reasoning: A method for solving a new problem based on the solutions of similar past problems

Blackboard system: AI application based on and similar to a gathering of specialists situated in a room with a large blackboard functioning collectively to solve a common problem

Question 31

Protheses: A mechanical device with computer assistance that substitutes or enhances some basic human functionality

Question:

Write a program To print "hello world" 10 times.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int i;
for (i =1; i<=10; i ++)
cout<<"\n hello world";
return 0;
}
```

Merge/purge: The technique of combining records from one or more data sources and deleting duplicate entries

---

Question 32

Question:

Write a program to print first 5 numbers using do while loop statement.

Solution:

```
#include<iostream>
```

```
using namespace std;
int main()
{
int i =1;
do
{
cout<<" \n i= "<< i++;
} while (i<=5);
return 0;
}
```

Merge/Purge Problem:

A common problem involves the merging of data from numerous sources with different representations in the most effective and efficient manner possible while improving the correctness of the outcome

---

Question 33

Question:

Write a program to print the output:

body [b] = b

body [o] = o

body [d] = d

body [y] = y

Missing data	Sparse data
A large number of data points are unknown	All of the data points are known − but the majority of them have zero value

---

Solution:

```
#include <iostream>
using namespace std;
int main()
{
char i;
char body [4] = {'b', 'o', 'd', 'y'};
```

Machine Discovery

Automated processes that make an effort to find patterns or features in data

```
for(i=0; i<4; i++)
cout<<"\n body ["<<body[i] <<" ] = "<< body[i] << endl;
return 0;
}
```

Question 34

Question:

What will be the output of the below program:

```
#include <iostream>
using namespace std;
int main()
{
cout<<"linux\n";
exit (0);
cout<<"php\n";
return 0;
}
```

- **Class:** A label assigned to a group of records sharing similar characteristics
- **Epoch:** The number of times a learning algorithm visits a given training dataset

---

Solution:

```
linux
```

Question 35

Abduction: A form of reasoning where assumptions are made to explain observations

Question:

Write a program to check whether a character is an alphabet or not.

Solution:

```
#include <iostream>
using namespace std;
int main()
{
int a =2;
if(isalpha(a))
{
   cout<<"The character a is an alphabet";
}
else
{
cout<<"The character a is not an alphabet";
}
return 0;
}
```

Neural Machine Translation

↓

The use of neural network models to translate words from one language to another

---

Open Mind Common Sense

↓

AI project based at the MIT Media Lab whose aim is to build a large **commonsense knowledge base** from the contributions of the general public over the web

→ A database holding all the general knowledge that most people possess

Question 36

Question:

Write a program to calculate the discounted price and the total price after discount

Given:

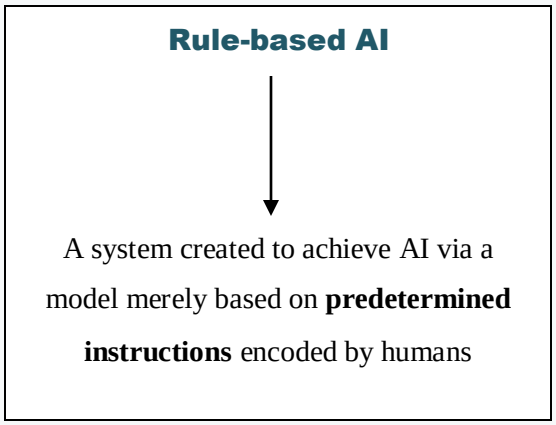
If purchase value is greater than 1000, 10% discount

If purchase value is greater than 5000, 20% discount

If purchase value is greater than 10000, 30% discount.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
double PV;
cout<<"Enter purchased value:";
cin>>PV;
if(PV>1000)
{
cout<<"Discount = "<< PV* 0.1 << endl;
cout<<"Total= "<< PV - PV* 0.1 << endl;
}
else if(PV>5000)
{
cout<<"Discount = "<< PV* 0.2 << endl;
cout<<"Total= "<< PV - PV* 0.2 << endl;
}
else
```



```
{
cout<<"Discount = "<< PV* 0.3 << endl;
cout<<"Total= "<< PV - PV* 0.3 << endl;
}
return 0;
}
```

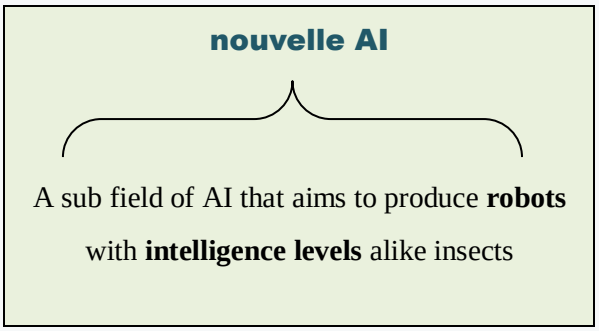
Question 37

Question:

Write a program to print the first ten natural numbers using while loop statement.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int i = 1;
while (i<=10)
{
cout<<"\n "<< i++;
}
return 0;
}
```



Data Stewardship:

The process of guaranteeing an Enterprise organization's information is accessible, usable, safe and credible

Question 38

Symbolic AI: The clear embedding of human knowledge and behavior rules into computer programs

Question:

What will be the output of the below program:

```
#include <iostream>
using namespace std;
int main()
{
int i;
for (i=1; i<=5; i++)
{
if (i==3)
{
continue;
}
cout<<"\n "<< i;
}
return 0;
}
```

Model checking: The method of checking whether the model meets a given specification

Self-management

↓

The process by which machines manage their own operations without human intercession

Solution:

Swarm intelligence: Biologically-inspired AI based on the behavioral models of a large or dense group of flying insects

```
1
2
4
5
```

Question

Machine vision

The ability of a machine to identify and evaluate images

Question:

Write a program to find the size of an array.

```
#include <iostream>
using namespace std;
int main() {
int num [] = {11, 22, 33, 44, 55, 66};
int n;
n = sizeof(num) / sizeof(num[0]);
cout<<"Size of the array is:"<<n;
return 0;
}
```

```
#include <iostream>
using namespace std;

int main() {
  int a = 25;   // Now a is 25
   a = 50;     // Now a is 50
  cout << a;
  return 0;
}

Output:
        50
```

Assigning a new value to an existing value

```
#include <iostream>
using namespace std;
int main() {
  // single-line comment before a line of code
  cout << "Albert"<<endl;
/* Multi-line
   Comment */
  cout << "Einstein"<<endl; // single-line comment at the end of a line of code
  return 0;
}
```

Recommendation system

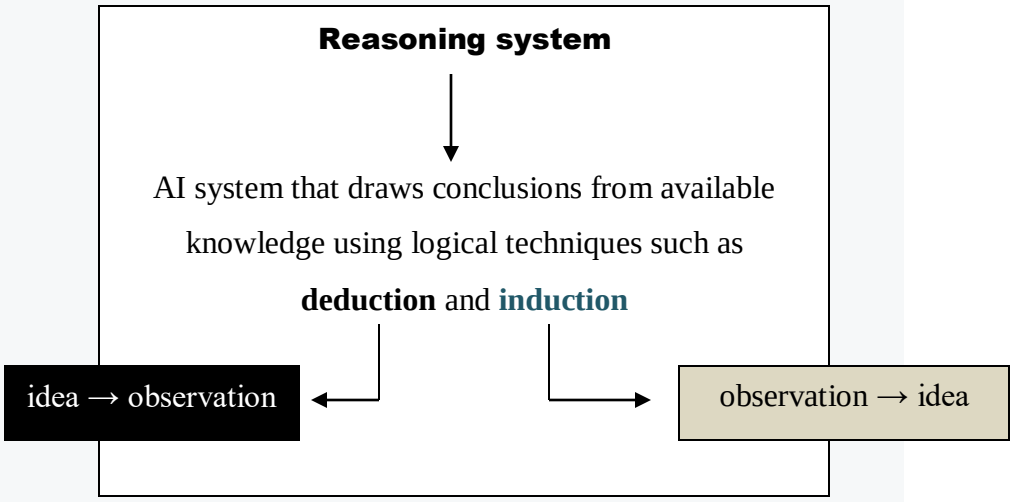
Information filtering system aimed at predicting what the users may like **(based on their previous activity or explicit feedback)** and recommend products or services that quite likely are interesting for them

Question

Question:

What would be the output of the following programs:

```
#include <iostream>
using namespace std;
int main()
{
int i;
for (i=1; i<=5; i++)
{
if (i==3)
{
break;
}
cout<<"\n "<< i;
}
return 0;
}
```



Solution:

```
1
2
```

```
#include<iostream>
using namespace std;
```

```
int main()
{
int i;
for(i=1;i<=5;i++)
{
if(i==3)
{
goto HAI;
}
cout<<"\n "<< i;
}
HAI : cout<<"\n Linux";
}
```

Qualification problem

How to deal with the things that keep us from accomplishing our intended result

---

Solution:

```
1
2
Linux
```

```
#include<iostream>
using namespace std;
int main()
{
int i = 54;
int y = i<<1;
cout<<"The value of y = "<< y;
return 0;
}
```

Solution:

```
The value of y = 108
```

```
#include<iostream>
using namespace std;
int main()
{
int i = 54;
int y = i>>1;
cout<<"The value of y = "<< y;
return 0;
}
```

Serialization: The process of converting data structures into a sequence of bytes which can be stored in a database or transmitted through the network

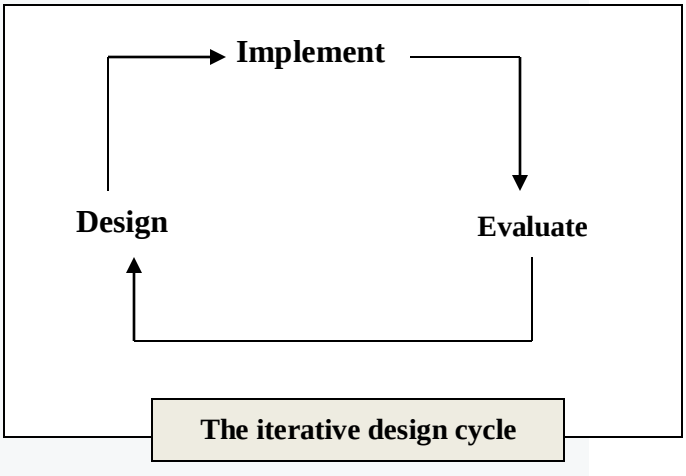
---

Solution:

Iteration → The repetition of a process

```
The value of y = 27
```

```
#include<iostream>
#include<cmath>
using namespace std;
int main()
{
int a, b;
a= - 2;
b= abs(a);
cout<<" Absolute value = "<< b<< endl;
return 0;
}
```



The iterative design cycle

Solution:

```
Absolute value = 2
```

```
#include <iostream>
using namespace std;
int main()
{
for( ; ; ) {
cout<<"This loop will run forever.\n";
}
return 0;
}
```

Heuristic Search technique: A realistic problem-solving strategy that reduce the number of searches for best solution by eliminates wrong choices

---

Solution:

```
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever. .........
```

Innovation Diffusion: An idea that attempts to describe how, why, and how quickly new ideas and technology spread

---

```
#include<iostream>
using namespace std;
int main()
{
cout<<"Hello World!";
```

Intelligent Enterprise Strategy: The management strategy that improves enterprise organizational effectiveness by incorporating advanced technology, best practices and new service models

```
return 0;
cout<<"Hello World!";
}
```

Solution:

```
Hello,world!
```

Question 41

Question:

Write a program to check whether the person is a senior citizen or not.

Solution:

```
#include<iostream>
using namespace std;
int main()
{
int age;
age=20;
if(age > = 60)
{
cout<<"Senior citizen";
}
if(age<60)
{
cout<<"Not a senior citizen";
```

Data Relevancy: The degree to which the content of data is helpful for the goal for which it is gathered

```
}
return 0;
}
```

Question 42

Question:

Write a program to compute inverse of tan x.

Pandas: An open-source software library written for the Python programming language – used for manipulation and analysis of data

---

Solution:

```
#include<iostream>
#include<math.h>
using namespace std;
int main()
{
int x = 20;
cout<<"Inverse of tan x = "<< atan(x);
return 0;
}
```

Timeliness of data: The information is available and accessible when required

```
#include <iostream>
using namespace std;
int main() {
int x = 2, y = 3;
double i;
i = (double)x/y;
cout << i << endl;
return 0;
}
// Output: 0.666667
```

Question 43

Question:

Write a program to find the highest value of 2 numbers

Solution:

```cpp
#include <iostream>
using namespace std;

int main() {
  cout << max(25, 20);
  return 0;
}
```

```cpp
#include <iostream>
using namespace std;
int main() {
string x, a("Alan"), b(" "), c("Turing");
x = a + b + c;
cout<<x;
return 0;
}

// Output: Alan Turing
```

Question 44

Personality Insights

Individual personality characteristics, needs, and philosophies can be predicted by analyzing email, social posts, etc.

Question:

Write a program to find the lowest value of 2 numbers

```cpp
#include <iostream>
using namespace std;

int main() {
  cout << min(25, 20);
  return 0;
}
```

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main() {
  cout << sqrt(4) << "\n";
  cout << round(5.8) << "\n";
  cout << log(1) << "\n";
  return 0;
}
```

Output:

```
2
6
0
```

Longitudinal Study:

An observational research design in which the same variables are examined repeatedly over a short or long period of time

Question 45

- `x += 5;` is equivalent to `x = x + 5;`
- `x *= y + 4;` is equivalent to `x = x * (y+4);`

Question:

Write a program to call a function multiple times

```
#include <iostream>
using namespace std;

void myFunc() {
  cout << "Albert\n";
}

int main() {
  myFunc();
  myFunc();
  myFunc();
  return 0;
}
```

```
#include <iostream>
using namespace std;
int main() {
cout.put('A');
return 0;
}

// Output: A
```

```
#include <iostream>
using namespace std;
int main() {
cout << "\nAlbert Einstein\t was a\n\t\t"
"German \"born\" theoretical physicist.\n";
return 0;
}
```

Output:

```
Albert Einstein was a
                German "born" theoretical physicist.
```

```cpp
#include <iostream>
using namespace std;
int main() {
double b = 15.569635;
cout.precision(4);
cout << " " << b;
return 0;
}

// Output: 15.57
```

```cpp
#include <iostream>
using namespace std;
int main() {
double b = 15.569635;
cout.precision(3);
cout << " " << b;
return 0;
}

// Output: 15.6
```

```cpp
#include <iostream>
#include <iomanip>
using namespace std;
int main() {
double b = 15.569635;
cout << setprecision(4) << b;
return 0;
}

// Output: 15.57
```

```cpp
#include <iostream>
#include <iomanip>
using namespace std;
int main() {
double b = 15.569635;
cout << setprecision(3) << b;
return 0;
}

// Output: 15.6
```

```cpp
#include <iostream>
#include <iomanip>
using namespace std;
int main() {
cout << setfill('*') << setw(4) << 15 <<endl;
cout << setfill('*') << setw(6) << 15 <<endl;
return 0;
}

// Output:
            **15
            ****15
```

```cpp
#include <iostream>
using namespace std;
int main() {
float x(7.0);
cout << x++ - 15.0/3.0;
return 0;
}
// Output: 2
```

```cpp
#include <iostream>
#include <iomanip>
using namespace std;
int main() {
string x("Albert");
cout << right << setfill('?') << setw(12) << x;
return 0;
}

// Output: ??????Albert
```

```cpp
#include <iostream>
using namespace std;
int main() {
string x("AL"), y("BERT");
x += y;
cout<<x;
return 0;
}

// Output: ALBERT
```

```cpp
#include <iostream>
#include <iomanip>
using namespace std;
int main() {
string x("Albert");
cout << left << setfill('?') << setw(12) << x;
return 0;
}

// Output: Albert??????
```

```cpp
#include <iostream>

using namespace std;

int main() {

int x(24), y(73);

cout << x++ << endl; // Output: 24

cout << x << endl; // Output: 25

cout << y-- << endl; // Output: 73

cout << --y << endl; // Output: 71

return 0;

}
```

```cpp
#include <iostream>

using namespace std;

int main() {

int x = -16;

cout << -x;

return 0;

}

// Output: 16
```

```cpp
#include <iostream>
using namespace std;
int main() {
int x = 0;
while( x < 5)
cout << ++x << endl;
return 0;
}

// Output:
                    1
                    2
                    3
                    4
                    5
```

```cpp
#include <iostream>
using namespace std;
int main() {
int x = 0;
while( x < 5)
cout << x++ << endl;
return 0;
}

// Output:
                    0
                    1
                    2
                    3
                    4
```

```cpp
#include <iostream>
using namespace std;
int main() {
int x = 3;
for (int i = -6; i < x; i++)
cout << i << endl;
return 0;
}

// Output:
               -6
               -5
               -4
               -3
               -2
               -1
                0
                1
                2
```

```cpp
#include <iostream>
using namespace std;
int x = 0;
namespace Special { int x = 50; }
int main() {
cout << x << endl; // Output: 0
cout << ++ ::x << endl; // Output: 1
cout << -- ::x << endl; // Output: 0
cout << Special::x * 4 << endl; // Output: 200
return 0;
}
```

```cpp
#include <iostream>
using namespace std;
int main() {
string x( 5, '+');
cout<< x;
return 0;
}
// Output: +++++
```

```cpp
#include <iostream>
#include <list>
using namespace std;
int main(){
   list<int> x = { 100, 200, 300, 400, 500 };
   cout<<"First element in the list is: "<< x.front() << endl;
   cout<<"Last element in the list is: "<< x.back() << endl;
   return 0;
}

// Output:
                First element in the list is: 100
                Last element in the list is: 500
```

```cpp
#include <iostream>
#include <list>
using namespace std;

int main() {
    list<int> x{ 1, 2, 3, 4, 5 };
    for (auto i = x.begin(); i != x.end(); ++i)
        cout << ' ' << *i;
    return 0;
}

// Output:  1 2 3 4 5
```

Image restoration: The method of getting a high quality image from a corrupted image

Computer Vision:

A branch of AI that deals with enabling machines to visualize, identify and derive meaningful information from images and videos in the same way that a human being does

Goodness of Fit

How well a set of actual values match those predicted by the ML model

Dummy variable:

A variable that takes values of 0 and 1, where 0 indicate the absence of something and 1 indicate presence of something

Amdahl's law:

$$\text{Speedup} = \frac{\text{Execution time without enhancement}}{\text{Execution time with enhancement}}$$

Tells us how much faster a task will be completed using the enhanced machine over the original machine

```cpp
#include <iostream>
#include <list>
using namespace std;

int main() {
    list<int> x{ 1, 2, 3, 4, 5 };
    for (auto i = x.begin(); i != x.end(); ++i)
        cout << ' ' << *i;  // 1 2 3 4 5

     x.remove(2);
    cout<<'\n';
for (auto i = x.begin(); i != x.end(); ++i)
        cout << ' ' << *i;  // 1 3 4 5
    return 0;
}

// Output:
                    1 2 3 4 5
                     1 3 4 5
```

Frequentist Statistics:

An statistical approach that tests whether an event occurs or not

Predicate Calculus

A formal logical framework for displaying and reasoning with logical statements

```cpp
#include <iostream>
using namespace std;

void message()
{
cout << "Albert Einstein is waiting." << endl;
}
int main(){
message();
return(0);
}

// Output: Albert Einstein is waiting.
```

```cpp
#include <iostream>
using namespace std;

int main()
{
    int x[] = { 20, 40, 60, 80 };

    for (int i : x)
        cout<< i <<" ";

}

// Output: 20 40 60 80
```

Question 46

Question:

Write a program to check if the vector container is empty or not.

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    vector<int> x{11, 12, 13, 14, 15};
    if (x.empty()) {
        cout << "True";
    }
    else {
        cout << "False";
    }
    return 0;
}
```

Output: False

```cpp
#include <iostream>
using namespace std;

int main()
{
    int x[] = { 20, 40, 60, 80 };

    for (auto i : x)
        cout<< i <<" ";

}

// Output: 20 40 60 80
```

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    vector<int> x{};
    if (x.empty()) {
        cout << "True";
    }
    else {
        cout << "False";
    }
    return 0;
}

// Output: True
```

```cpp
#include <iostream>
using namespace std;
#define concat(x, y) x ## y

int main() {
   int ab = 125;

   cout << concat(a, b);
   return 0;
}

// Output: 125
```

```cpp
#include <iostream>
using namespace std;
int main () {
   cout << __LINE__ << endl; // Line number: 4
   cout << __FILE__ << endl; // File name: main.cpp
   cout << __DATE__ << endl; // Date: Jul 20 2022
   cout << __TIME__ << endl; // Time: 16:58:55
   return 0;
}
```

Neuromorphic chip: A kind of computer chip that imitate closely in an uncomplicated way how neurons and neurotransmitters work in the human cerebrum

AI Adoption:

The ability of an **enterprise organization** to effectively adopt **AI** in its business cycle to tackle issues and accomplish objectives

Question 47

Question:

Write a program to demonstrate multiple cases with unique labels

---

Solution:

```
#include <iostream>
using namespace std;

int main() {
  char x='c';
  switch(x) {
    case 'a':
    case 'b':
       cout<<"Albert Einstein\n";
       break;

    case 'c':
    case 'd':
      cout<<"Max Born\n";
      break;

    default: cout<<"Wolfgang Pauli\n";
      break;
  }
  return 0;

}

// Output: Max Born
```

Soft Robotics: A branch of robotics that concentrates on advanced technologies that is more physically similar to the physical characteristics of living organisms

P-hacking: The inappropriate usage of information that yields misleading statistical results

Data Exhaust:

The unstructured data generated as a byproduct of the online activities of internet users. **For Example:** cookies, log files and temporary internet files are generated and deposited by web browsers and their plugin in the user's computer

```cpp
#include <bits/stdc++.h>
using namespace std;

int main() {
    vector<string> i{ "Albert",
                      "James",
                      "Alan"};

    for (const auto& b : i) {
        cout << b << '\n';
    }
    return 0;
}
```

```
// Output:

Albert
James
Alan
```

```cpp
extern "C" {
int printf(const char* format, ...);
}

int main() {
    printf("Albert Einstein");
    return 0;
}

// Output: Albert Einstein
```

```cpp
#include <iostream>

int main(int argc, char* argv[]) {
    std::cout << "Albert Einstein";
    return 0;
}

// Output: Albert Einstein
```

```cpp
#include <iostream>
using namespace std;

int main() {
  do{
        cout << "Hi....\n";
    } while(1);
  return 0;
}
```

Infinite Loop

```cpp
#include <iostream>
using namespace std;
int main() {
char ch;
cout << "Please enter a character: ";
cin >> ch;
cout <<"Integer equivalent of " << ch << " is: " << static_cast<int>(ch) << endl;
return 0;
}

// Output:
Please enter a character: A # entered character
Integer equivalent of A is: 65
```

```cpp
#include <iostream>
using namespace std;
int main() {
int i = 1;
while (i <= 6) {
cout << (i % 2 ? "????" : "#######") << endl;
++i;
}
return 0;
}
```

```
// Output:

       ????
       #######
       ????
       #######
       ????
       #######
```

```cpp
#include <iostream>
using namespace std;
int main() {
int a = 5, b = 15;
if (a < 12) {
if (b > 12)
cout << "??????" << endl;
}
else {
cout << "********" << endl;
cout << "#######" << endl;
}
return 0;
}
```

```
// Output:

              ??????
```

Question 48

- **Qualitative Data:** Data that can be described, categorized and expressed in terms of descriptive words rather than numerical values
- **Quantitative Data:** Data that can be measured, counted and easily expressed using numbers

Question:

Write a program to Access Elements of an Array Using Pointer

Solution:

```
#include <iostream>
using namespace std;
int main() {
int x[3];
cout << "Enter any three numbers: " << endl;
for(int i = 0; i < 3; ++i)
cin >> x[i];
cout << "You entered: ";
for(int i = 0; i < 3; ++i)
cout << endl << *(x + i);
return 0;
}
```

```
#include <iostream>
using namespace std;
int main() {
int x =6;
switch (x % 2) {
 case 0:
 cout << "x is even" << endl;
 break;
 case 1:
 cout << "x is odd" << endl;
 break;
}
 return 0;
}

// Output: x is even
```

```
#include <iostream>
using namespace std;
int main() {
int b = 28, a =25;
if (b == 28)
if (a == 25) {
cout << "+++++++\n"; }
else {
cout << "-------\n";
}
cout << "*******\n";
cout << "@#@#$$$$\n";
return 0;
}
```

```
// Output:

+++++++
*******
@#@#$$$$
```

```cpp
// C++ program print the sum of the odd numbers from 1 to 25

#include<iostream>
using namespace std;

int main() {
	int n, x, sum = 0;

	cout << "\n Please enter a number = ";
	cin >> x;

	cout <<"\n Odd numbers between 1 and " << x << " is: ";
	for(n = 1; n <= x; n++){
		if ( n % 2 != 0 ) {
			cout << n << " ";
			sum = sum + n;
		}
	}

	cout << "\n The sum of the odd numbers from 1 to " << x << " = " << sum;

	return 0;
}
```

Data generalization: The process of deriving conclusions from a small amount of data

```
// Output:

Please enter a number = 25 # entered number
Odd numbers between 1 and 25 is: 1 3 5 7 9 11 13 15 17 19 21 23 25
The sum of the odd numbers from 1 to 25 = 169
```

Data Profiling: The process of evaluating, exploring, analyzing, reviewing the data available from an existing information source and producing useful summaries of data to acquire understanding into the quality of data

```cpp
#include<iostream>
using namespace std;

int main() {
	int i, sum = 0;
	char ch[20];

	cout << "Please enter a string: ";
	cin.getline(ch, 20);

	for(i = 0; ch[i] != '\0'; i++) {
		cout << "\n ASCII value of "<< ch[i] << " : " << (int)ch[i];
		sum = sum + ch[i];
	}

	cout << "\n The sum of all ASCII values: "  << sum;

	return 0;
}
```

```
// Output:

Please enter a string: ALAN # entered string
 ASCII value of A : 65
 ASCII value of L : 76
 ASCII value of A : 65
 ASCII value of N : 78
 The sum of all ASCII values: 284
```

Traditional Machine Learning:

Input → Feature extraction → Classification → Output

Deep Learning:

Input → (Feature extraction + Classification) → Output

Python code:

```python
for a in range(5):
  x=0
  x=x+a
print(x) # Output: 4
```

Multithreading: A method of simultaneously handling multiple requests from the same user

Long Polling: A web application development strategy for pushing data as soon as possible from servers to clients

CORS

(Cross-Origin Resource Sharing):

An **HTTP-header based method** that allows regulated access to resources outside of a specific domain

Utterance:

Any human user's input into a Chabot

Everything as a Service:

The delivery of a wide variety of software applications, tools and technologies to the users as a service via the internet whenever they need them

Analogical reasoning:

2 things are compared and conclusions are made based on their similarities or dissimilarities

---

Java Exercises

Admissibility: An algorithm's ability to always discover the best solution

Question 1

Handwriting Recognition: A computer machine's capability to receive and translate handwritten letters or words into a format that it recognizes

Question:

Write a program to print Hello World!.

Solution:

```
public class MyClass {
public static void main(String [] args) {
System.out.println("Hello, World!");
}
}
```

Linguistic intelligence: The ability to effectively use and interpret spoken and written language

---

Question 2

Logical mathematical intelligence: The ability to think logically about problems and solve them using logical and mathematical methods

Question:

Write a program to find the area of a circle.

Solution:

```
public class MyClass {
public static void main (String [] args) {
int r, area;
r = 2;
area = 3.14 * r * r;
System.out.println("The area of the circle = " + area);
}
}
```

Episodic Learning: Learning by recalling sequences of witnessed or experienced events

---

Question 3

Question:

Write a program to find the sum of two numbers.

Solution:

Observational Learning: Learning by observing and mimicking others

```
public class MyClass {
public static void main(String [] args) {
```

```
int a, b, sum;
a=1;
b=2;
sum = a + b;
System.out.println("The sum of a and b = " + sum);
}
}
```

C++ Program:

```
#include <iostream>
using namespace std;
int main() {
string x("Alan "),y("Turing");
cout << x + y << endl;
// Output: Alan Turing
x += y;
cout <<x << endl;
// Output: Alan Turing
x += "!";
cout <<x;
// Output: Alan Turing!
return 0;
}
```

Question 4

Question:

Write a program to find the square of a number.

Solution:

```
public class MyClass {
public static void main(String [] args) {
int a, b;
a=2;
b = a * a;
System.out.println("The square of a = " + b);
}
}
```

The null hypothesis

A theoretical hypothesis based on **incomplete proof** – which requires further testing to prove whether it is true or false

Cognitive Computing: Simulating human thought processes in machines to make decisions and solve complex problems

Cognitive System: A system that can demonstrate the behavior of a human brain

Question 5

```
# Python code to print the minimum and maximum values present in a tuple
tuple=(4,5,6,7,8)
print(min(tuple))
print(max(tuple))
```

Output on the screen:

4

8

Question:

Write a program to find the greatest of two numbers.

Solution:

```
public class MyClass {
public static void main(String [] args) {
int a, b;
a=2;
b =3;
if(a>b)
{
System.out.println("a is greater than b");
}
else
{
System.out.println("b is greater than a");
}
}
}
```

```
#include <bits/stdc++.h>
using namespace std;

int main() {
  array<int, 5> x = { 11, 12, 13, 14, 15 };
  cout << "The element at index 3 is: " << x.at(3) << endl;
  // Output: The element at index 3 is: 14
  cout << "The element at index 3 is: " << x[3] << endl;
  // Output: The element at index 3 is: 14
  return 0;
}
```

x.at(i) is equivalent to x[i]

Question 6

Question:

Write a program to print the average of the elements in the array.

Solution:

```
public class MyClass {
public static void main(String[] args) {
int i, avg, sum = 0;
int [] num = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
avg = sum/5;
System.out.println("Sum of the Elements in the array = " + sum);
System.out.println("Average of the Elements in the array = " + avg);
}
}
```

Rolling Deployment Strategy: A software release approach in which an older version of a software application is progressively replaced with a newer one by completely replacing the infrastructure on which it runs

Question 7

Question:

Write a program such that a Switch (case) allows to make a decision from the number of choices, i.e., from the number of cases.

Solution:

```
public class MyClass {
public static void main(String[] args)throws Exception {
char ch;
System.out.print("Enter a character:");
ch = (char)System.in.read();
```

Process synchronization:

The technique in which an operating system manages processes that share the same resources and data

```
switch(ch)
{
case 'R':
System.out.print("Red");
break;
case 'W':
System.out.print("White");
break;
case 'Y':
System.out.print("Yellow");
break;
case 'G':
System.out.print("Green");
break;
default:
System.out.print("Error");
break;
}
}
}
```

Overlaying: The technique of replacing what is already stored in internal memory with a block of software code or other information

Reentrancy:

During the same time period, many users can use and share a single copy of a program

Context Switching:

A method of preserving the state of one process and loading the state of another process

---

Question 8

Question:

Write a program to read 10 numbers from the keyboard and find their sum and average.

Solution:

Time-sharing system: A system that allows several users to access a system's resources from multiple locations

```java
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, sum;
float X;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any ten Numbers: ");
N1 = scan.nextInt();
N2  =  scan.nextInt();
N3  =  scan.nextInt();
N4  =  scan.nextInt();
N5  =  scan.nextInt();
N6  =  scan.nextInt();
N7  =  scan.nextInt();
N8  =  scan.nextInt();
N9  =  scan.nextInt();
N10 = scan.nextInt();
sum = N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9 + N10;
X = sum /10;
System.out.println("The sum of 10 numbers = " + sum);
System.out.println("The average of 10 numbers = " + X);
}
}
```

```python
# Python code to check the memory usage of integer

import sys
a = 25
print(sys.getsizeof(a))
```

Question 9

Question:

Write a program to print the first 10 numbers starting from one together with their squares and cubes.

Solution:

```
public class MyClass {
public static void main(String[] args) throws Exception {
int i;
for( i=1; i<=10; i++)
System.out.println(" \n number = " + i + " its square = " + i*i + " its cube = " +i*i*i);
}
}
```

Question 10

Question:

Write a program:

If you enter a character M

Output must be: ch = M.

```
# Python code to reverse the string

a = "Albert"
print(a[: : -1])
```

Solution:

```
public class MyClass {
public static void main(String[] args) throws Exception {
char c;
System.out.print("Enter a character:");
c = (char)System.in.read();
System.out.println("ch= " + c);
}
}
```

```
a = 2
b = "Albert";
print(a * b)
```

Output on the screen:

AlbertAlbert

Question 11

Question:

Write a program to print the multiplication table of a number.

Solution:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int n, i;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
n = scan.nextInt();
for( i=1; i<=5; i++)
System.out.println (n + " * " + i + " = " + n * i);
}
}
```

Installation and checkout phase:

During this stage of the software development life cycle, a software application is integrated into its operational environment and tested there to ensure that it performs as expected

Question 12

Question:

Write a program to print the product of the first 10 digits.

Solution:

DML

(Data Manipulation Language)

↓

A computer language used to manipulate the data

```
public class MyClass {
public static void main(String [] args) {
int i, product = 1;
for( i=1; i<=10; i++)
product = product * i;
System.out.println("The product of the first 10 digits = " + product);
}
}
```

Question 13

Question:

Write a program to print whether the given number is positive or negative.

Solution:

DDL

(Data Definition Language)

↓

A computer language used to define data structures

```
public class MyClass {
public static void main(String [] args) {
int a;
a = -35;
if(a>0)
{
System.out.println("Number is positive");
}
else
{
System.out.println("Number is negative");
}
```

```
}
}
```

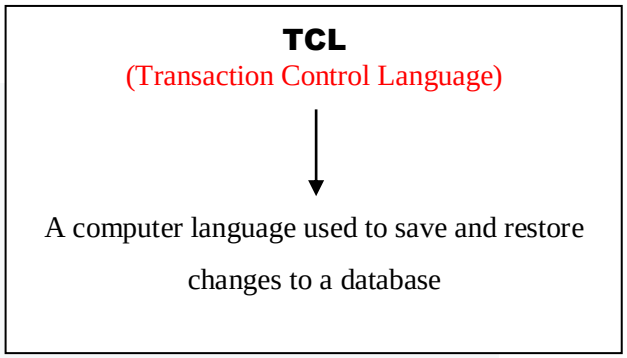
Question 14

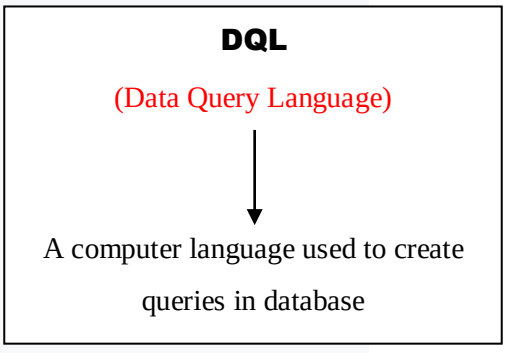
Question:

Write a program to check the equivalence of two numbers.

Solution:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int x, y;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
x = scan.nextInt();
System.out.println("Enter a number: ");
y = scan.nextInt();
if(x-y==0)
{
System.out.println("The two numbers are equivalent");
}
else
{
System.out.println("The two numbers are not equivalent");
}
}
}
```





Question 15

Question:

Write a program to print the remainder of two numbers.

Solution:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
a = scan.nextInt();
System.out.println("Enter a number: ");
b = scan.nextInt();
c = a%b;
System.out.println("The remainder of a and b = " + c);
}
}
```

RDBMS (Relational Database Management system)	DBMS (Database Management System)
Data stored is in table format	Data stored is in the file format
Deal with vast amount of data	Deal with small amount of data

Question 16

```
# Python code to print a 2*2 Numpy array with only zeroes
import numpy as np
a=np.zeros((2,2))
print(a)
```

Output on the screen:

```
[[0. 0.]
 [0. 0.]]
```

Question:

Write a program to print the given number is even or odd.

Solution:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int a;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
a = scan.nextInt();
if(a%2 == 0)
{
System.out.println("The number is even");
}
else
{
System.out.println("The number is odd");
}
}
}
```

Thrashing:

A situation in which there are a lot of **paging** operations

A storage technique that allows the operating system to fetch processes in the form of pages from secondary storage and store them in the main memory

Question 17

Bug leakage: The testing team is ignorant of the bug during testing, which is later found by the customer or end-user

Question:

Write a program to print the characters from A to Z.

Solution:

```
public class MyClass {
```

Bug Release:

With the understanding that a bug exists, a software application is delivered to the testing team

Mean Substitution:

The method of replacing missing values with mean of other available values

```
public static void main(String [] args) {
char a;
for( a='A'; a<='Z'; a++)
System.out.println("\n " + a);
}
}
```

```
#include <iostream>
using namespace std;
int main() {
string x("Albert");
x.erase(3);
cout<<x; // Output: Alb
return 0;
}
```

```
#include <iostream>
using namespace std;
int main() {
string x("Albert");
x.insert(3, "Mary");
cout<<x; // Output: AlbMaryert
return 0;
}
```

Question 18

Question:

Write a program to find the incremented and decremented values of two numbers.

Solution:

Dispersion: The spread of data

```
public class MyClass {
public static void main(String [] args) {
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
System.out.print("The incremented value of a = "+ c);
System.out.print("The incremented value of b = "+ d);
System.out.print("The decremented value of a = "+ e);
System.out.print("The decremented value of b = "+ f);
}
```

Translational invariance

The ability of a algorithm to successfully classify images even when the position of objects within the image changes

```
}
```

Question 19

```
#include <iostream>
using namespace std;
int main() {
string x("Albert Einstein");
x.erase(2,4);
cout<<x; // Output: Al Einstein
return 0;
}
```

Question:

Write a program to calculate the simple interest.

Solution:

```
public class MyClass {
public static void main(String [] args) {
int P,T, R, SI;
P = 1000;
T = 2;
R = 3;
SI = P*T*R/100;
System.out.println("The simple interest = " + SI);
}
}
```

Autoencoders

↓

Simple learning networks that target to transform inputs into outputs with the minimum possible error

---

Question 20

Naive Bayes theorem:

$$\text{Posterior probability} = \frac{\text{Likelihood} * \text{Prior probability}}{\text{evidence}}$$

Question:

Write a program to Find the largest of three numbers.

Solution:

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any number:");
a = scan.nextInt();
System.out.println("Enter any number:");
b = scan.nextInt();
System.out.println("Enter any number:");
c = scan.nextInt();
if(a>b&&a>c)
{
System.out.println("a is greater than b and c");
}
else if(b>a&&b>c)
{
System.out.println("b is greater than a and c");
}
else
{
System.out.println("c is greater than b and a");
}
}
}
```

Data Ecosystem: A platform that collect, analyze and store data

Data Hygiene: The process of ensuring that an enterprise organization has error-free, updated, consistent and accurate data

BYOD

(Bring Your Own Device)

A policy that allows employees in an enterprise organization to use their personal smartphones, laptops and tablets to access enterprise data from anywhere

Question 21

Crystal Methodology: An agile software development strategy that prioritizes individuals and their interactions rather than techniques and methodologies when working on a project

Question:

Write a program to print the factorial of the entered number.

Solution:

Product quality	Identify how well a product fits client requirements, fulfils its objective, and complies with industry benchmarks
Process quality	Identify inefficiencies in the production process that result in bad products
Usability	How simple the software application is to use or manage
Portability	The ability of a software application to run on different operating systems
User View	How successfully the software application meets the user's requirements
Value-based View	How many customers are ready to buy the software application

```
import java.util.Scanner;
public class MyClass {
public static void main(String []args){
int i, n, fact=1 ;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any number:");
n = scan.nextInt();
for(i = 1; i <= n; i++)
        {
            fact = fact * i;
        }
        System.out.println("Factorial of " + n + " is: " + fact);
    }
}
```

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
'A'	'l'	'b'	'e'	'r'	't'	' '	'i'	's'	' '	'a'	' '	't'	'e'	'c'	'h'	'i'	'e'

Question 22

Question:

Write a program to print the length of the entered string.

```
#include <iostream>
using namespace std;
int main() {
string x("Albert is a techie");
int first = x.find("tec");
cout<<first;
// Output: 12
return 0;
}
```

Solution:

```
import java.util.Scanner;
public class MyClass {
public static void main(String[] args) {
String a;
Scanner scan = new Scanner(System.in);
System.out.print("Enter Your Name : ");
      a = scan.nextLine();
System.out.println("The length of the String is: " + a.length());
}
}
```

Control program: A computer program that controls and manages the operation of a computer

Question 23

Question:

Write a program to print the output:

Einstein [0] = E

Einstein [1] = I

Einstein [2] = N

Einstein [3] = S

Einstein [4] = T

Einstein [5] = E

Einstein [6] = I

Einstein [7] = N

Commerce service provider: An organization that provides businesses with application tools to enable them sell their services and products to customers all over the world over the internet.

Compile time

The time it takes for a computer program to be built into an executable file

Solution:

```
public class MyClass {
public static void main(String[] args) throws Exception{
int i;
char [] num = {'E' , 'I', 'N', 'S', 'T', 'E', 'I', 'N'};
for(i=0; i<8; i++)
System.out.println("Einstein [" + i + " ] = " + num[i]);
}
}
```

Data communication: The transmission of data through the internet between a source and a recipient

---

Question 24

A brute force attack: A cryptographic attack that use a trial-and-error method to guess all possible combinations until the user passwords, login details, cryptographic keys and PINs are revealed

Question:

Write a program to find square of a number using method.

Solution:

```
import java.util.Scanner;
public class MyClass {
public static void main(String[] args) {
int x;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
x = scan.nextInt();
System.out.println("Square of the number = " + square (x));
}
public static int square (int x){
return x*x;
}
```

Concurrent server: A server that can handle multiple clients at once

```
}
```

Question 25

Question:

Write a program To print "hello world" 10 times.

Data check: An operational activity that verifies the quality, correctness, completeness and consistency of data

---

Solution:

```
public class MyClass {
public static void main(String [] args) {
int i;
for (i =1; i<=10; i ++)
System.out.println("\n hello world");
}
}
```

Connectivity: A device's capacity to be connected to other devices

---

Question 26

Question:

Write a program to print first 5 numbers using do while loop statement.

Solution:

```
public class MyClass {
public static void main(String [] args) {
int i =1;
do
{
System.out.println(" \n " + i++);
} while (i<=5);
}
}
```

Cryptographic algorithm: A collection of rules describing the mathematical steps needed to encrypt and decrypt information

---

Question 27

Question:

Write a program to print the output:

body [b] = b

body [o] = o

body [d] = d

body [y] = y

Cryptography

A branch of computer science concerned with converting information into formats that cannot be identified by unauthorized users

---

Solution:

```
public class MyClass {
public static void main(String[] args) throws Exception{
int i;
char [] body = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
System.out.println("body [" + body [i] + " ] = " + body [i]);
}
```

```
}
```

Question 28

Question:

Write a program to print the first ten natural numbers using while loop statement.

Solution:

Data Science Operationalization

↓

Moving data science models into production and then managing them

```
public class MyClass {
public static void main(String [] args) {
int i = 1;
while (i<=10)
{
System.out.println("\n " + i++);
}
}
}
```

- **Low bias machine learning algorithms include:** Decision Trees, k-nearest neighbors, Support Vector Machine
- **High bias machine learning algorithms include:** Linear Regression and Logistic Regression

---

Question 29

Question:

What will be the output of the below program:

Data Screening

↓

Checking data for errors and fixing the errors

```
public class MyClass {
public static void main(String []args) {
int i;
for (i=1; i<=5; i++) {
if (i==3) {
continue;
}
System.out.println("" + i);
}
}
}
```

Data Strategy

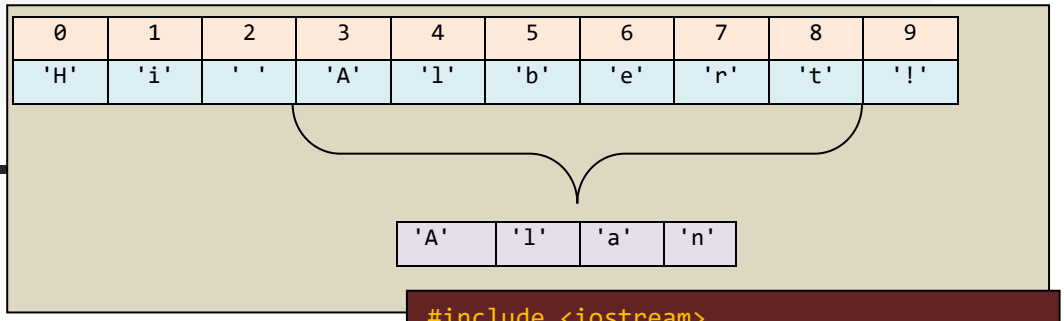
The method of defining how to collect, store, manage, share and use data in support of business and technology objectives

- **Interpolation** → Predicting values that are inside the range of data taken (Reliable)
- **Extrapolation** → Predicting values that are outside the range of data taken (Unreliable)

Solution:

```
1
2
4
5
```

Data verification	Data validation
Ensure that the data is accurate	Ensure that the data is correct



Question 30

Question:

Write a program to find the size of an array.

```
#include <iostream>
using namespace std;
int main() {
string x("Hi Albert!"),
y("Alan");
x.replace(3, 6, y);
cout<< x; // Output: Hi Alan!
return 0;
}
```

Solution:

```
public class MyClass {
public static void main(String[] args) {
   int num [] = {11, 22, 33, 44, 55, 66};
    System.out.println("Size of the array is: " + num.length);
}
}
```

0	1	2	3	4	5	6	7	8	9	10
'A'	'l'	'b'	'e'	'r'	't'	''	'J'	'o'	'h'	'n'

Question 31

Question:

The string y replaces 4 characters, "John", starting at position 7 in the string x. After this operation x contains the string "Albert Einstein"

What would be the output of the following programs:

```
#include <iostream>
using namespace std;
int main() {
string x("Albert John"),
y("Einstein");
int i = x.find("John");
if( i != string::npos )
x.replace(i, 4, y);
cout<<x;
// Output: Albert Einstein
return 0;
}
```

```
public class MyClass {
public static void main(String []args) {
int i;
for (i=1; i<=5; i++) {
if (i==3) {
break;
}
System.out.println("" + i);
}
}
}
```

Data aggregation	Data disaggregation
The process of gathering and combing data from various sources and presenting it as a whole for statistical analysis	The process of breaking down the aggregated data into smaller units of data to elucidate trends and patterns that are masked by aggregated data

---

Solution:

Data Aggregation: Collect the data → Process the data → Present the data

```
1
2
```

Denial-of-Service: A malicious attempt in which the perpetrator bring a system or network to a halt, making it unavailable to its intended users

```
public class MyClass {
public static void main(String [] args) {
int x = 2;
System.out.println(" Square of a number = " + Math.pow((x), 2));
}
}
```

```
#include <iostream>
using namespace std;
int main() {
string x = "Alberg";
x[x.length() -1] = 't';
cout<<x;
// Output: Albert
return 0;
}
```

Solution:

```
Square of a number = 4.0
```

```
public class MyClass {
public static void main(String [] args) {
int i = 54;
int y = i<<1;
System.out.println("The value of y = " + y);
}
}
```

Timeboxing: A time management approach for limiting the amount of time spent on a task

---

Solution:

```
The value of y = 108
```

Application security: The process of preventing harmful attacks on software applications

```
public class MyClass {
public static void main(String [] args) {
int i = 54;
```

```
int y = i>>1;
System.out.println("The value of y = " + y);
}
}
```

Network security: the technique for protecting a computer network from unauthorized access and malicious attacks

---

Solution:

```
The value of y = 27
```

```
import java.util.Scanner;
public class MyClass {
public static void main(String [] args) {
String m;
Scanner in = new Scanner(System.in);
System.out.print("Enter the name: ");
m = in.nextLine();
System.out.println("The name you entered = " + m);
}
}
```

Clickjacking: A malicious technique of scamming website visitors to click on user interface components that perform operations on another website. **For example:** A website that offers an outstanding prize may entice a user. When a user clicks a button to accept a prize, their click is instead used to make an e-commerce purchase

---

Solution:

Identity management: The procedure by which an enterprise organisation ensures that a person's access to technology resources is authorized

```
Enter the name:
Dennis
The name you entered = Dennis
```

```
public class MyClass {
```

```
public static void main(String[] args) {
for( ; ; )
{
System.out.println("This loop will run forever.\n");
}
}
}
```

Vulnerability Assessment: A method for identifying, detecting, and analyzing security flaws in a computer system

---

Solution:

```
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever. .........
```

Progressive Rendering: A technique that is commonly used to speed up the rendering of content on a website

---

```
public class MyClass {
public static void main(String [] args) {
System.out.println("Hello, World!");
System.exit(0);
System.out.println("Hello, World!");
}
}
```

Operational security: A data security and risk management technique that encrypts data in the database and prevents sensitive data from getting into the hands of cybercriminals

---

Solution:

```
Hello,world!
```

Question 32

Question:

Write a program to check whether the person is a senior citizen or not.

Solution:

```
public class MyClass {
public static void main(String [] args) {
int age;
age=20;
if(age>= 60)
{
System.out.println("senior citizen");
}
else
{
System.out.println("not a senior citizen");
}
}
}
```

Program Slicing:

A section of a code in which certain statements

(which aren't required to understand a certain property of the code)

are eliminated

Question 33

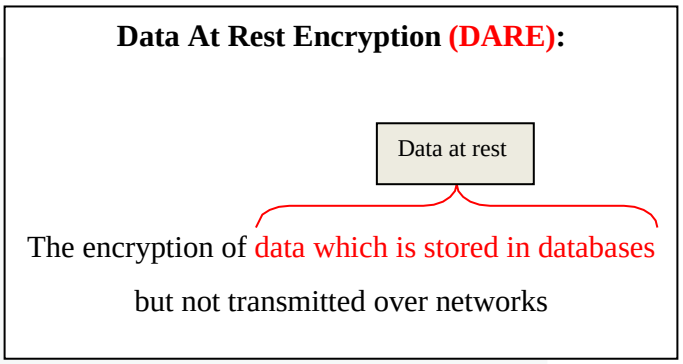
Question:

Write a program to Find ASCII Value of a character.

Encryption in Transit: Encrypting data while it travels between two services over the internet or a private network

Solution:

```
public class MyClass {

    public static void main(String[] args) {

        char ch = 'a';
        int ascii = ch;
        int castAscii = (int) ch;


        System.out.println("The ASCII value of " + ch + " is: " + ascii);
        System.out.println("The ASCII value of " + ch + " is: " + castAscii);
    }
    }
```




```
#include <iostream>
using namespace std;
int main() {
int a = 20, b = 2;
cout<< !(a < 6) && !(b >= 8);  // Output: 1
cout<< endl;
cout<< !((a < 6) || (b >= 8));  // Output: 1
return 0;
}
```

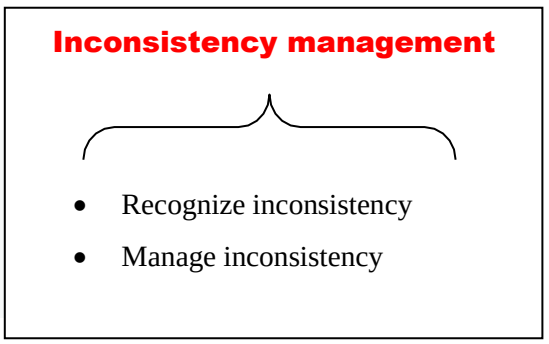
Question 34

Question:

Write a program to Swap Two Numbers.

Solution:

```
public class MyClass {

    public static void main(String[] args) {
```



```
        float first = 12.0f, second = 24.5f;

        System.out.println("Before swap:");
        System.out.println("First number = " + first);
        System.out.println("Second number = " + second);

        first = first - second;
        second = first + second;
        first = second - first;

        System.out.println("After swap:");
        System.out.println("First number = " + first);
        System.out.println("Second number = " + second);
    }
}
```

Shoulder surfing: An unlawful action in which one individual monitors the screen and keyboard of another person's computer or mobile device in order to gather sensitive information

```
#include <iostream>
using namespace std;
int main() {
int x = 5, y = 5, z = 7;
cout<< !(x == y) || !(z != 7); // Output: 0
cout<< endl;
cout<< !((x == y) && (z != 7)); // Output: 1
return 0;
}
```

```
#include <iostream>
using namespace std;
int main() {
int a = 6, b = 2, c = 3, d = 11;
cout<< !((a <= 9) && (b > 5)); // Output: 1
cout << endl;
cout<< !((a <= 9) || (b > 5)); // Output: 0
cout << endl;
cout<< !((c > 5) || (d <= 7)); // Output: 1
cout << endl;
cout<< !((c > 5) && (d <= 7)); // Output: 1
return 0;
}
```

```
public class MyClass {
public static void main(String[] args) {
    String a = "52";
    int b = 20;
    String c = a + b;
    System.out.println(c);
  }
}

// Output: 5220
```

```
public class MyClass {
  public static void main(String[] args) {
  int a = 20, b =19;

// returns true because the value of a is equal to 20
    System.out.println(a == 20); // Output: true

// returns true, because 20 is greater than 19
    System.out.println(a > b); // Output: true
  }
}
```

Question 35

Question:

Write a program to remove All Whitespaces from a String.

Solution:

```
public class MyClass {
 public static void main(String[] args) {
        String x = "A    lbert Ei n  ste    in";
        System.out.println(x);
        x = x.replaceAll("\\s", "");
        System.out.println(x);
    }
}
```

```
public class MyClass {
public static void main(String[] args) {
 int x =6;
 System.out.printf("%d + %d%d  + %d%d%d\n", x, x, x, x, x, x);
 }
}


// Output: 6 + 66  + 666
```

```
// Java program to display the Current Date time

public class MyClass {
 public static void main(String[] args) {
  System.out.format("%tc%n", System.currentTimeMillis());
 }
} // Output: Sun Jul 24 08:59:05 GMT 2022
```

```java
import java.util.*;

public class MyClass {
     public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter a number: ");
         int x = in.nextInt();
             if (x % 2 != 0) {
                 System.out.println(1);
                } else {
                 System.out.println(0);
               }
     }
}
```

Accept a number and check the number is odd or not. Prints 1 if the number is odd or 0 if the number is even

Question 36

Question:

Write a program to print the odd numbers from 1 to 49.

Solution:

```java
public class MyClass {
 public static void main(String[] args) {
 for (int x = 1; x <= 49; x++) {
 if (x % 2 != 0) {
 System.out.println(x);
 }
}
}
}
```

Question 37

Question:

Write a program to print the even numbers from 1 to 50.

Solution:

```
public class MyClass {
 public static void main(String[] args) {
 for (int x = 1; x <= 50; x++) {
 if (x % 2 == 0) {
 System.out.println(x);
 }
}
}
}
```

```
public class MyClass {
 public static void my2(int[]i) {
 System.out.println("Albert");
 }
public static void main(String[] args) {
 my2(new int[]{10,20,30});
 }
 }

// Output:
                    Albert
```

```
// Java program to print the next to last word of a sentence

import java.util.*;
public class MyClass {
     public static void main(String[] args) {
     String x = "Albert Einstein was a German-born theoretical physicist.";
     String[] i = x.split("[ ]+");
     System.out.println(i[i.length - 2]);
}
}

// Output: theoretical
```

Question 38

Question:

Write a program to print numbers between 1 to 50 which are divisible by 2, 4 and by both

Solution:

```
public class MyClass {
   public static void main(String args[]) {
   System.out.println("\n Numbers divisible by 2: ");
             for (int x=1; x<50; x++) {
                    if (x%2==0)
                    System.out.print(x + " ");
             }

       System.out.println("\n");

             System.out.println("\n Numbers divisible by 4: ");
             for (int x=1; x<50; x++) {
                    if (x%4==0) System.out.print(x + " ");
             }

       System.out.println("\n");

             System.out.println("\n Numbers divisible by 2 and 4: ");
             for (int x=1; x<50; x++) {
                    if (x%2==0 && x%4==0) System.out.print(x + " ");
             }
       System.out.println("\n");
  }
}
```

Question 39

Question:

Write a program to Count Number of Digits in an Integer.

Solution:

```
public class MyClass {

    public static void main(String[] args) {

        int i = 0, x = 5496;

        while(x != 0) {
            x /= 10;
            ++i;
        }

        System.out.println("Number of digits: " + i);
    }
}
```

```
public class MyClass {
 public static void my2(int... i) {
 System.out.println("Albert");
 }
public static void main(String[] args) {
 my2();
 my2(1);
 my2(1,2,3);
 }
 }
```

```
// Output:

        Albert
        Albert
        Albert
```

```
// Java program to print the first half of a string of even length
import java.lang.*;

public class MyClass {
public static void main(String[] args) {
    String x = "Einstein";
    System.out.println(x.substring(0, x.length()/2));
  }
 }
```

```
// Output:

        Eins
```

Question 40

Question:

Write a program to compare two strings.

```
public class MyClass {
   static {
 System.out.println("Albert Einstein");
 }
 public static void main(String[] args) {
 System.out.println("Alan Turing");
 }
}

// Output:
              Albert Einstein
                Alan Turing
```

Solution:

```
public class MyClass {

    public static void main(String[] args) {

        String x = "Albert";
        String y = "Albert";

        if(x == y)
            System.out.println("Equal");
        else
            System.out.println("Not Equal");
    }
}
```

```
public class MyClass {
public static void main(String[] args) {
int x=0;
for(;x<3;x++) {
System.out.println("Alan");
}
}
}
```

```
// Output:

        Alan
        Alan
        Alan
```

```
// Java program to Get Current Working Directory

public class MyClass {

  public static void main(String[] args) {

        System.out.println(System.getProperty("user.dir"));

    }

}
```

```
public class MyClass {
  public static void my2(int[]... i) {
  for(int[] z:i) {
     System.out.println(z[1]);
 }
 }
 public static void main(String[] args) {
 int[] x = {1,2,3,4};
 int[] y = {5,6,7};
 my2(x, y);
 }
}
```

```
// Output:

2
6
```

```
public class MyClass {
public static void main(String[] args) {
 String[] x={"Alan", "Albert", "Mary"};
 args=x;
 for(String i : args) {
 System.out.println(i);
 }
 }
}
```

```
// Output:

Alan
Albert
Mary
```

```
public class MyClass {

public static void main(String[] args) {

if (true) {

System.out.println("Albert Einstein");

}

System.out.println("Alan Turing");

}

}
```

```
// Output:

Albert Einstein
Alan Turing
```

```java
public class MyClass {
public static void main(String[] args) {
if (false) {
System.out.println("Albert Einstein");
}
System.out.println("Alan Turing");
}
}
```

```
// Output:

Alan Turing
```

```java
public class MyClass {
public static void main(String[] args) {
int x=2;
switch (x) {
case 1:System.out.println("1");
break;
case 1:System.out.println("2");
break;
case 3:System.out.println("3");
break;
default :System.out.println("0");
break;
}
}
}
```

```
// Output:

ERROR

In a switch statement,
duplicate case labels are not
permitted
```

```java
public class MyClass {
public static void main(String[] args) {
int x=40+50+60;
switch (x) {
case 10+20+30:System.out.println("1");
break;
case 40+50+60:System.out.println("2");
break;
case 70+80+90:System.out.println("3");
break;
default :System.out.println("0");
break;
}
}
}
```

```
// Output:

2
```

```
public class MyClass {
public static void main(String[] args) {
int x=0;
for(System.out.println("Albert");x<3;x++) {
System.out.println("Einstein");
}
}
}
```

```
// Output:

Albert
Einstein
Einstein
Einstein
```

```
// Infinite loop

public class MyClass {
public static void main(String[] args) {
for(int x=0;;x++) {
System.out.println("Alan");
}
 }
}
```

```
public class MyClass {
public static void main(String[] args) {
for (int x=0;true;x++) {
System.out.println("Alan");
}
}
}
```

```
public class MyClass {
public static void main(String[] args) {
for(int x=5; x>0; x--){
System.out.println(x);
}
}
}
```

```
// Output:

5
4
3
2
1
```

```
// Infinite loop

public class MyClass {
public static void main(String[] args) {
for(int x=1; x>=1; x++) {
System.out.println("Alan");
}
}
}
```

```
public class MyClass {
public static void main(String[] args) {
for(int x=0;x<2;System.out.println("Alan"),System.out.println("Albert")) {
System.out.println("John");
x++;
}
}
}
```

```
// Output:

John
Alan
Albert
John
Alan
Albert
```

```
public class MyClass {
public static void main(String[] args) {
int x=0;
do {
System.out.println("Alan");
}
while (false);
System.out.println("Albert");
}
}
```

```
// Output:

Alan
Albert
```

```
public class MyClass {

static int i=50;

public static void main(String[] args) {

System.out.println(MyClass.i);

}

}


// Output: 50
```

```
public class MyClass {
static int i=50;
public static void main(String[] args) {
MyClass x = new MyClass();
System.out.println(x.i);
}
}

// Output: 50
```

```java
public class MyClass {
public static void main(String[] args) {
int[] x={1,2,3,4};
for(int y:x) {
System.out.println(y);
}
}
}
```

```
// Output:

1
2
3
4
```

```java
public class MyClass {
public static void main(String[] args) {
StringBuffer x=new StringBuffer("Albert");
System.out.println(x.deleteCharAt(1));
// Output: Abert
}
}
```

```java
public class MyClass {
public static void main(String[] args) {
int[] x=new int[3];
x[0]=5;
x[1]=10;
x[2]=15;
// prints the length of the array
System.out.println(x.length); // Output: 3

for (int a=0;a<x.length;a++ ) {
System.out.println(x[a]);
}
// Output:
//          5
//         10
//         15
}
}
```

```java
import static java.lang.System.*;
public class Main {
public static void main(String[] args) {
out.println("Alan Turing");
}
}
// Output: Alan Turing
```

```java
public class MyClass {
public static void main(String[] args) {
String x[]={"Alan", "Mathison", "Turing"};
for(String i : x) {
System.out.println(i);
}
}
}
```

```
// Output:

Alan
Mathison
Turing
```

```
public class MyClass {
public static void main(String[] args) {
char[] x={'A','J','o','h','n','l','b'};
String y=new String(x,1,4);
System.out.println(y);
}
}

// Output: John
```

```
public class MyClass {
public static void main(String[] args) {
StringBuffer x=new StringBuffer("Albert");
System.out.println(x.delete(1,4));
// Output: Art
}
}
```

```
public class MyClass {
public static void main(String[] args) {
byte[] x={97,98,99,100,101,102};
String y=new String(x,2,4);
System.out.println(y);
}
}

// Output: cdef
```

Character	ASCII code
a	97
b	98
c	99
d	100
e	101
f	102

```
public class MyClass {
public static void main(String[] args) {
String x="albert";
String y="ALBERT";
System.out.println(x.equalsIgnoreCase(y)); // Output: true
System.out.println(y.endsWith("T")); // Output: true
System.out.println(x.startsWith("a")); // Output: true
}
}
```

```
public class MyClass {

public static void main(String[] args) {

StringBuffer x=new StringBuffer("Albert");

System.out.println(x.reverse()); // Output: treblA

}

}
```

```
public class MyClass {
public static void main(String[] args) {
StringBuffer x =new StringBuffer("Five");
String i=" Papers of ";
int y=1905;
x.append(i);
x.append(y);
System.out.println(x);
}
}
```

```
// Output:

    Five Papers of 1905
```

```
public class MyClass {
public static void main(String[] args) {
StringBuffer x=new StringBuffer("Einstein");
x.insert(0,"Albert ");
System.out.println(x);
}
}
```

```
// Output:

     Albert Einstein
```

```
public class MyClass {
public static void main(String[] args) {
StringBuffer x=new StringBuffer("Albert Einstein");
x.replace(0,6,"Evelyn");
System.out.println(x);
}
}
```

```
// Output:

      Evelyn Einstein
```

```
public class MyClass {
public static void main(String[] args) {
System.out.println("Albert Einstein");
System.exit(0);
System.out.println(500/0);
}
}
```

```
// Output:

      Albert Einstein
```

```
import java.util.*;
public class MyClass {
public static void main(String[] args) {
Stack x=new Stack();
x.push("Einsteinian");
x.push(1905);
x.push("Papers");
System.out.println(x);
}
}
```

```
// Output:

   [Einsteinian, 1905, Papers]
```

```
import java.util.*;
public class MyClass {
public static void main(String[] args) {
HashSet x=new HashSet();
x.add("Albert");
x.add("Papers");
x.add(1905);
x.add(null);
System.out.println(x);
}
}
```

```
// Output:

   [null, 1905, Papers, Albert]
```

```
import java.util.*;
public class MyClass {
public static void main(String[] args) {
HashSet x=new HashSet();
x.add("Albert");
x.add("Albert");
x.add("Papers");
x.add(1905);
x.add(1905);
x.add(null);
System.out.println(x);
}
}
```

```
// Output:

   [null, 1905, Papers, Albert]
```

```
import java.util.*;
public class MyClass {
public static void main(String[] args) {
TreeSet x=new TreeSet();
x.add(35);
x.add(15);
x.add(25);
x.add(10);
x.add(30);

System.out.println(x); // Output: [10, 15, 25, 30, 35]

SortedSet a=x.headSet(35);
System.out.println(a); // Output: [10, 15, 25, 30]

SortedSet b=x.tailSet(25);
System.out.println(b); // Output: [25, 30, 35]

SortedSet c=x.subSet(15,35);
System.out.println(c); // Output: [15, 25, 30]

}
}
```

```
import java.util.*;

public class MyClass {

public static void main(String[] args) {

HashMap x=new HashMap();

x.put("Albert", 1978);

x.put("John", 1965);

x.put("James", 1945);

System.out.println(x);

}

}
```

```
// Output:

  {James=1945, John=1965, Albert=1978}
```

```
import java.util.*;

public class MyClass {

public static void main(String[] args) {

Hashtable x=new Hashtable();

x.put("Albert", 1978);

x.put("John", 1965);

x.put("James", 1945);

System.out.println(x);

// Output: {James=1945, John=1965, Albert=1978}

System.out.println(x.contains(1945));

// Output: true

System.out.println(x.containsValue(1988));

// Output: false

}

}
```

```
// Java program to display all the multiples of 5 within the range 15 to 25

public class MyClass {
public static void main(String[] args) {
for(int x=15;x<=25;x++) {
      if(x%5==0)
      System.out.println(x);
}
}
}
```

```
import java.lang.*;
public class MyClass {
public static void main(String[] args) {
    String x = "John";
    String y = "Turing";
System.out.println(x.substring(1) + y.substring(1));
}
}
```

Java program to remove the first character of each string and concatenate the 2 strings

Question 41

Question:

Write a program to print the sum of even numbers from 1 to 10 using for loop

```
public class MyClass {
public static void main(String args[]) {
char ch;
ch = 'A';
System.out.println(ch); // Output: A
ch++;
System.out.println(ch); // Output: B
}
}
```

Solution:

```
public class MyClass {
public static void main(String[] args) {
   int sum=0;
   for(int x=1;x<=10;x++) {
        if(x%2 == 0){
          sum = sum + x;
        }
      }
  System.out.println("The sum of even numbers from 1 to 10 is: "+sum);
    }
}

// Output: The sum of even numbers from 1 to 10 is: 30
```

```
public class MyClass {
public static void main(String[] args) {
   int sum=0;
   for(int x=1;x<=10;x++) {
        if(x%2 != 0){
          sum = sum + x;
        }
      }
 System.out.println("The sum of odd numbers from 1 to 10 is: "+sum);
    }
}

// Output: The sum of odd numbers from 1 to 10 is: 25
```

Question 42

```
public class MyClass {
public static void main(String[] args) {
    System.out.println("20 > 19 is " + (20 > 19));
    // Output: 20 > 19 is true
 }
}
```

Question:

Write a program to remove HTML Tags from a string

Solution:

```
public class MyClass {
    public static void main(String[] args) {
        String x= "<p>Stephen William Hawking</p>";
        x = x.replaceAll("\\<.*?>", "");
        System.out.println(x);
    }
}

// Output: Stephen William Hawking
```

```
public class MyClass {
public static void main(String args[]) {
for(int i=1; i<5; i=i+1)
System.out.println(i);
}
}
```

```
// Output:

1
2
3
4
```

```
import java.util.*;

public class MyClass {
       public static void main(String[] args) {

           char ch[] = {'A', 'l', 'b', 'e', 'r', 't'};
              String x = String.valueOf(ch);

              System.out.println(x);
       }
}

// Output: Albert
```

Question 43

Question:

Write a program to check whether a number is a Buzz number or not

---

Solution:

```java
// A buzz number is any number that either ends with 7 or is divisible by 7.

import java.util.Scanner;
 public class MyClass {
       public static void main(String[] args) {
           Scanner in=new Scanner(System.in);
           System.out.println("Enter a number:");
              int x = in.nextInt();
              if(x%7==0 || x%10==7) {
                  System.out.println(x + " is a Buzz Number");
              } else {
                  System.out.println(x + " is not a Buzz Number");
              }
       }
}
```

```
// Output:

Enter a number:
127 # entered number
127 is a Buzz Number
```

```java
public class MyClass {
public static void main(String[] args) {
int a = 20, b = 60;
if(a < b) System.out.println("a<b");
// Output: a<b
a = a * 3;
if(a == b) System.out.println("a=b");
// Output: a=b
a = a * 3;
if(a > b) System.out.println("a>b");
// Output: a>b
}
}
```

```java
public class MyClass {
public static void main(String args[]) {
int y = 20;
for(int i = 0; i<5; i++) {
System.out.println(y);
y = y - 2;
}
}
}
```

```
// Output:

        20
        18
        16
        14
        12
```

Question 44

Question:

Write a program to check if 20 appears as either the first or last element of an array of numbers

---

Solution:

```
import java.lang.*;
 public class MyClass {
 public static void main(String[] args) {
    int[] x = {20, 40, 0, 60, 70, 80, 20};
    System.out.println((x[0] == 20 || x[x.length-1] == 20));
    // Output: true
 }
}
```

```
public class MyClass {
public static void main(String[] args) {
int a = 10;
if(a == 10) {
int b = 20;
a = b * 3;
}
System.out.println(a);
}
}

// Output: 60
```

```
public class MyClass {
public static void main(String[] args) {
int a = 20;
if(a == 10) {
int b = 20;
a = b * 3;
}
System.out.println(a);
}
}

// Output: 20
```

Question 45

Question:

Write a program to check if the first and the last element of an array of numbers are same

---

Solution:

```
import java.lang.*;
 public class MyClass {
 public static void main(String[] args) {
    int[] x = {50, 10, 0, 20, 30, 40, 50};
 System.out.println (x.length >= 2 && x[0] ==  x[x.length-1]);
      // Output: true
 }
}
```

```
public class MyClass {
public static void main(String[] args) {
for(int i=1; i<3; i++) {
int a = 1905;
System.out.println(a);
a = 1915;
System.out.println(a);
 }
}
}
```

```
// Output:
       1905
       1915
       1905
       1915
```

```
public class MyClass {
public static void main(String[] args) {
byte i = 25;
i = (byte)(i * 3);
System.out.println(i);
 }
}
// Output: 75
```

```
public class MyClass {
public static void main(String[] args) {
int x[] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
System.out.println("February has " + x[1] + " days.");
 }
}
// Output: February has 28 days.
```

```
public class MyClass {

public static void main(String[] args) {

for(int i=1; i<6; i++)

switch(i) {

case 1:

case 2:

System.out.println("i>3");

break;

case 3:

case 4:

System.out.println("i<5");

break;

default:

System.out.println("i=5 or i>5");

}

}

}
```

```
// Output:

      i>3
      i>3
      i<5
      i<5
   i=5 or i>5
```

```
public class MyClass {
public static void main(String[] args) {
int i = 5;
while(i > 0) {
System.out.println(i);
i--;
}
}
}

// Output:
                        5
                        4
                        3
                        2
                        1
```

```
public class MyClass {
public static void main(String[] args) {
int x = 10, y = 20;
while(++x < --y) ;
System.out.println(x);
}
}

// Output: 15
```

```
public class MyClass {
public static void main(String[] args) {
int x = 5;
do {
System.out.println(x);
} while(--x > 0);
}
}
```

```
// Output:

5
4
3
2
1
```

Question 46

Question:

Write a program to add all the digits of a given positive number until the result has a single digit

---

Solution:

```
public class MyClass {
 public static void main(String[] args) {
  int x = 1264;
    if (x>0) {
           System.out.print((x == 0 ? 0 : (x % 9 == 0 ? 9 : x % 9)));
      }
          System.out.println("\n");
 }
}

// Output: 4

// 1 + 2 + 6 + 4 = 13

// 1 + 3 = 4
```

```
public class MyClass {
public static void main(String args[]) {
int x;
x = 'C';
System.out.write(x);
System.out.write('\n');
}
}

// Output: C
```

```
import java.io.*;
public class MyClass {
public static void main(String args[]) {
PrintWriter x = new PrintWriter(System.out, true);
x.println("Albert Einstein");
}
}

// Output: Albert Einstein
```

```
public class MyClass {

public static void main(String args[]) {

Double x = new Double(1/0.);

Double y = new Double(0/0.);

System.out.println(x.isInfinite()); // Output: true

System.out.println(x.isNaN()); // Output: false

System.out.println(y.isInfinite()); // Output: false

System.out.println(y.isNaN()); // Output: true

}

}
```

```java
import java.util.*;
public class MyClass {
public static void main(String args[]) {
Formatter x = new Formatter();
x.format("%s spends between $%d and $%f annually for books" , "John", 100, 198.6);
System.out.println(x);
}
}

// Output: John spends between $100 and $198.600000 annually for books
```

```java
import java.util.*;
public class MyClass {
public static void main(String args[]) {
Formatter x = new Formatter();
x.format("|%f|%n|%10f|%n|%010f|",54.98765, 54.98765, 54.98765);
System.out.println(x);
}
}

// Output:
                    |54.987650|
                    | 54.987650|
                    |054.987650|
```

```java
public class MyClass {
static void x(String i) {
System.out.println(i);
}
public static void main(String args[]) {
x("Albert Einstein");
}
}

// Output: Albert Einstein
```

```java
import java.util.*;

public class MyClass {
public static void main(String args[]) {
for(int a=1; a <= 4; a++) {
Formatter x = new Formatter();
x.format("%3d %3d %3d", a, a+a, a+a+a);
System.out.println(x);
}
}
}
```

```
// Output:

  1   2   3
  2   4   6
  3   6   9
  4   8  12
```

```
import java.io.*;
public class MyClass {
public static void main(String args[]) {
Console con;
con = System.console();
con.printf("Albert Einstein");
}
}

// Output: Albert Einstein
```

```
import java.util.regex.*;

public class MyClass {
public static void main(String args[]) {
Pattern x = Pattern.compile("[ ,.!]");
String y[] = x.split("Albert, 1905! Papers.");
for(int a=0; a < y.length; a++)
System.out.println(y[a]);
}
}
```

```
// Output:

Albert

1905

Papers
```

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class MyClass {
  public static void main(String[] args) {
    Pattern pattern = Pattern.compile("turing", Pattern.CASE_INSENSITIVE);
    Matcher matcher = pattern.matcher("Alan Turing was a brilliant British mathematician.");
    if(matcher.find()) {
      System.out.println("Text found");
    } else {
      System.out.println("Text not found");
    }
  }
}
```

```
// Output:

Text found
```

```java
// Java program to print the Unicode of the first character in a string
// The Unicode value of "A" is 65

public class MyClass {
  public static void main(String[] args) {
    String x = "Albert";
    System.out.println(x.codePointAt(0));
  }
}
```

```java
public class MyClass {
  public static void main(String[] args) {
    String x = "Alan";
    // The Unicode value of "A" is 65
    System.out.println(x.codePointBefore(1)); // Output: 65
    // The Unicode value of "l" is 108
    System.out.println(x.codePointBefore(2)); // Output: 108
    // The Unicode value of "a" is 97
    System.out.println(x.codePointBefore(3)); // Output: 97
    // The Unicode value of "n" is 110
    System.out.println(x.codePointBefore(4)); // Output: 110
  }
}
```

```java
public class MyClass {
public static void main(String[] args) {
String x = "Albert Einstein";
System.out.println(x.contains("bert"));
  }
}

// Output: true
```

```java
public class MyClass {
  public static void main(String[] args) {
    String x = "Albert";
    String y = "";
    System.out.println(x.isEmpty());
    // Output: false
    System.out.println(y.isEmpty());
    // Output: true
  }
}
```

```java
// Java program to print the hash code of a string

public class MyClass {
  public static void main(String[] args) {
    String x = "Albert";
    System.out.println(x.hashCode());
  }
}

// Output: 1963655312
```

```java
public class MyClass {
  public static void main(String[] args) {
    char[] x = {'A', 'l', 'a', 'n'};
    String y = "";
    System.out.println(y.copyValueOf(x, 0, 4));
  }
}

// Output: Alan
```

```java
public class MyClass {
  private String x = "John";
  private String y = "Alan";
  private String z = "john_alan@gmail.com";
  private int i = 49;

  public static void main(String[] args) {
    MyClass my2 = new MyClass();
    System.out.println("Name: " + my2.x + " " + my2.y);
    System.out.println("Email: " + my2.z);
    System.out.println("Age: " + my2.i);
  }
}
```

```
// Output:

Name: John Alan
Email: john_alan@gmail.com
Age: 49
```

```java
public class MyClass {
  static void my2() {
    System.out.println("Albert Einstein");
  }

  public static void main(String[] args) {
    my2();
  }
}

// Output: Albert Einstein
```

```
public class MyClass {
  int i;

  public MyClass(int i) {
    this.i = i;
  }

  public static void main(String[] args)  {
    MyClass my2 = new MyClass(1905);
    System.out.println(my2.i);
  }
}

// Output: 1905
```

```
public class MyClass {
public static void main(String[] args) {

{ int i = 50;
      System.out.println(i); }

  }
}

// Output: 50
```

```
public class MyClass {
  static int my2(int i) {
    return 5 * i;
  }

public static void main(String[] args) {
    System.out.println(my2(12));
  }
}

// Output: 60
```

```
final class MyClass {

  public static void main(String[] args) {

    System.out.println("Albert");

  }

}

// Output: Albert
```

```
public class MyClass {

  public static void main(String[] args) {

    String x = "Albert", y = "albert";

    System.out.println(x.contentEquals("Albert")); // Output: true

    System.out.println(y.contentEquals("Albert")); // Output: false

  }

}
```

```
public class MyClass {
  public static void main(String[] args) {
        int x = 3>2 ? 10 : 5;
        System.out.println(x); // Output: 10
        int y = 1>2 ? 10 : 5;
        System.out.println(y); // Output: 5


  }
}
```

```
// Java program to print the total number of lines from a string

public class MyClass {
  public static void main(String[] args) {
        String x = "1.Alan\n2.John\n3.Mary\n4.Albert";
         // Output:
        //          1.Alan
        //          2.John
        //          3.Mary
        //          4.Albert
        System.out.println(x);
        int i = x.split("[\n]").length;
        System.out.println("Total number of Lines: "+ i);
        // Output: Total number of Lines: 4
  }
}
```

```
import java.util.*;

public class MyClass {
public static void main(String args[]) {
        // Create an empty Stack
        Stack<String> x = new Stack<String>();

        // Use push() to add elements into the Stack
        x.push("Alan Turing");
        x.push("was");
        x.push("a");
        x.push("brilliant");
        x.push("British mathematician");

        // Display the Stack
        System.out.println(x);
       // Output: [Alan Turing, was, a, brilliant, British mathematician]

        // Fetch the element at the head of the Stack
        System.out.println(x.peek());
       // Output: British mathematician

    }
}
```

```
// Java program to delete the file
import java.io.File;
public class MyClass {
  public static void main(String[] args) {
    // File to be deleted
    File file = new File("C:/Users/Manju/1.txt");
    // Deletes the file
    boolean x = file.delete();
    if(x) {
      System.out.println("1.txt is successfully deleted.");
    }
    else {
      System.out.println("1.txt doesn't exist.");
    }
  }
}
```

```
import java.util.*;

public class MyClass {
public static void main(String args[]) {
        // Create an empty Stack
        Stack<String> x = new Stack<String>();

        // Use push() to add elements into the Stack
        x.push("Alan Turing");
        x.push("was");
        x.push("a");
        x.push("brilliant");
        x.push("British mathematician");


        // Check for the emptiness of stack
        // Is the stack empty ?
        System.out.println(x.empty());
       // Output: false

        // Pop out all the elements
        x.pop();
        x.pop();
        x.pop();
        x.pop();
        x.pop();

        // Check for the emptiness of stack
        // Is the stack empty ?
        System.out.println(x.empty());
        // Output: true

    }
}
```

```
import java.util.*;
public class MyClass {
public static void main(String[] args) {
 System.out.println("Time to wake-up: ");
 System.out.println(new Date());
 }
}

// Output:
        Time to wake-up:
        Tue Jul 26 11:00:20 GMT 2022
```

```
public class MyClass {

public static void main(String[] args) {

for(int x = 1; x <= 20; x += 2) {

      System.out.println(x);

             }

       }

}
```

```
// Output:

        1
        3
        5
        7
        9
       11
       13
       15
       17
       19
```

```
public class MyClass {
public static void main(String[] args) {
String x = "ALBERT".toLowerCase( );
System.out.println(x); // Output: albert
String y = x.toUpperCase( );
System.out.println(y); // Output: ALBERT
}
}
```

```
public class MyClass {
public static void main(String[] args) {
String x = "Books";
String y = "Pencils";
System.out.println(x.concat(" and ").concat(y));
}
}

// Output: Books and Pencils
```

```
import java.util.Scanner;
public class MyClass {
public static void main(String[] args) {
Scanner in = new Scanner(System.in);
System.out.println("Which is the best place in the World to visit?");
int x = 0;
do {
System.out.println("1. Machu Picchu, Peru");
System.out.println("2. The Grand Canyon");
System.out.println("3. Colosseum, Rome");
System.out.println("Please select one: ");
x = in.nextInt();
} while(x > 3);
 System.out.println("You selected: " + x);
}
}
```

The user is given a menu of options by the program. Since there are only 3 possibilities, it then verifies that the user's entered number is not larger than 3. The user gets shown the menu again until they choose a number from 1 to 3 if the number is more than 3, which causes the condition to evaluate to true.

Two-factor authentication (2FA):

Users authenticate their identification using two-step verification steps in this security mechanism

Perimeter-based Cybersecurity:

The idea of securing data and IT resources from hackers by putting security measures in place at the network's perimeter

Network Sniffing: A method for analyzing packets of data sent via a network

Intrusion detection system (IDS):

A system that checks network traffic for malicious activity and sends out alerts when it is detected

Intrusion Prevention Systems (IPS):

A system that checks network traffic for malicious activity and takes appropriate security measures to prevent it

Python code:

```
print(*range(1,4))

# Output: 1 2 3 4
```

Dark data: Any data that typically is not used to obtain business insights or for decision making

Autonomic Computing

The ability of a computing system to operate independently of user input

Information Theory: The mathematical study of determining how much information there is in a message

Product personalization

The customization of products and services according to the needs or desires of the customers

Moore's Law

Roughly every 2 years, the number of transistors on silicon microchips will double

Domain Name System hijacking:

A type of cybercrime in which cybercriminals take advantage of flaws in the Domain Name System to route website visitors to malicious websites and steal data from computers

Network analytics: The collection and analysis of **network data** to improve the performance, reliability and security of the network

```
// calling a method multiple times

public class MyClass {

  static void my2() {

    System.out.println("Albert Einstein");

  }

public static void main(String[] args) {

    my2();

    my2();

    my2();

  }

}
```

```
// Output:

          Albert Einstein

          Albert Einstein

          Albert Einstein
```

Polymorphic virus:

A type of malware that is hazardous, destructive, or disruptive and that can change itself, making antimalware solutions difficult to detect it

```
public class MyClass {

  static void my2(int x, String y) {

  System.out.println(x + " " + y);

  }

public static void main(String[] args) {

    my2(1905, "Papers");

    my2(1915, "Papers");

  }

}
```

```
// Output:

       1905 Papers

       1915 Papers
```

Usability Testing: A software testing technique used to evaluate the software's usability

Data Ingestion: The process of collecting raw data from various sources and bringing it into a data processing system where it can be stored, examined and accessed

Project data analytics: The collection and analysis of **past** and **current project data** to make worthwhile decisions on project delivery

```
public class MyClass {

  static int my2(int a) {

    return a + 2 * a;

  }


public static void main(String[] args) {

  System.out.println(my2(4));

  }

}
```

```
// Output:

          12
```

Security Testing: A sort of software testing that identifies security flaws, cyber-attacks and potential risks in a software application and guards against hacker attacks

Active reconnaissance

↓

In this type of cyber attack: an attacker engages with the target device or a system to acquire information about vulnerabilities

Cross-browser testing:

Comparing and evaluating how a web-based application or website behaves in various browser environments such as Google Chrome, Mozilla Firefox, Opera, Chromium, Safari etc.

End to End testing: The entire process of testing a software application from beginning to end

A **necessary condition** →

One that must exist for an event to take place

A sufficient condition

↓

One that will cause the event to occur

Data Mining Query Language

(DMQL):

A SQL-based query language developed for data mining activities

- **Fan-in:** The largest number of digital inputs a single logic gate can receive
- **Fan-out:** The largest number of digital inputs that a single logic gate's output can sustain

Implementation Design Description:

A detailed explanation of all the data required to build a working system from one or perhaps more functional designs

```
public class MyClass {

// Create a checkMarks() method with an integer parameter called marks

  static void checkMarks(int marks) {

// If marks is less than 35, print "You are Failed"

    if (marks < 35) {

      System.out.println("You are Failed...");

// If marks is greater than or equal to 35, print "You are Passed"

    } else {

      System.out.println("You are Passed...");

    }

  }


  public static void main(String[] args) {

    checkMarks(35); // Call the checkMarks method and pass along an marks of 35

  }

}
```

```
// Output:

        You are Passed...
```

Soft state:

Even without input, the status of the system may vary with time

API testing: The method of testing that verifies that the APIs used by the software application perform as intended

A critical bug:

A bug that has a significant impact on the application's main functionality

Latent defect: A type of flaw or bug that has been present in a software application for so long but has only now been detected

Database administrator: A person who is in charge of performing all activities related to controlling, security, maintenance, performance and recovery of database

Data store: A location where information is held

```
enum names {
  Albert,
  James,
  Mary
}
public class MyClass {
  public static void main(String[] args) {
    for (names i : names.values()) {
      System.out.println(i);
    }
  }
}
```

```
// Output:
Albert
James
Mary
```

Data encrypting key: A key that can be used to encrypt, decrypt and authenticate data

Data streaming: The continuous transmission of data at a high speed

Human-in-the-loop: The process of inserting humans into ML process to create effective ML algorithms

Ground Truth:

Checking the results of the ML algorithm for accuracy against the accurate data gathered from the real world

Traditionally,

ML algorithms rely on **large amounts of data** in order to reach logical conclusions

Data Efficient Learning →

ML algorithms rely on **smaller data sets** in order to reach logical conclusions more quickly

```
public class MyClass {

enum names {

  Albert,

  James,

  Mary

}

 public static void main(String[] args) {

    for (names i : names.values()) {

      System.out.println(i);

    }

  }

}
```

```
// Output:

Albert

James

Mary
```

Deep computing

(High-speed computing + Complicated analytical programs)

↓

Allows business decision makers to evaluate, detect patterns in and act on gathered data in order to generate significant solutions to extremely complex and tough problems

Delta: The differences between 2 programs

Down: Not working

Direct access storage device: A fixed or removable storage device on which a computer system stores information

Disable: Make it nonfunctional

Disk operating system: A computer operating system that runs from a hard disk drive

```
enum names {
  Albert,
  James,
  Mary
}

public class MyClass {
  public static void main(String[] args) {
    names i = names.Mary;
    System.out.println(i);
  }
}
```

```
// Output:
Mary
```

Display: A visual presentation of information

Display station: A device with a display screen and a keyboard connected that enables a user to send and receive data from the system

Docking station: A hardware component that allows a mobile device to function as a desktop

Down-level: Software with limited functionality

Dump: A location where all available data about a problem is stored in piles

Downtime: The period during which a system is unavailable to users due to maintenance, upgrades, resets and other technical issues

Artificial Neuron

A digital construct that imitate closely the behavior of a biological neuron in the human brain

Pruning:

The process of removing unnecessary and inaccessible code so as to make the code more readable, easily maintainable and to eliminate unwanted solutions

Dashboard: A visual representation of all of the company's business data that allows stakeholders to evaluate where the company stands in real time and make important decisions

Apache Spark: An open-source data processing platform for big data analysis and modeling

Complex Event Processing: A method for combining, processing and evaluating enormous amounts of data concerning things that happen in real time to get real-time insights

Unified Data Analytics

The process of combining data science and data engineering to make AI more accessible to businesses and allow the businesses to speed their AI activities

Data Vault: A data modeling methodology for implementing a highly business-focused data warehouse successfully and efficiently

Demand Forecasting: The process of analyzing historical data and other information to make future forecasts about customer demand for a product or service

Model Risk Management: The supervision of the negative outcomes of decisions made using inaccurate or misused models

Early stopping: A feature that allows the training to be stopped automatically when a given metric stops progressing

Computational Linguistics: The discipline of linguistics concerned with the analysis and synthesis of language and speech using computers

```
enum names {
  Albert,
  Alan,
  Hilbert
}


public class MyClass {
  public static void main(String[] args) {
    names i = names.Alan;
    switch(i) {
      case Albert:
        System.out.println("Albert Einstein");
        break;
      case Alan:
        System.out.println("Alan Turing");
        break;
      case Hilbert:
        System.out.println("David Hilbert");
        break;
    }
  }
}
```

Data Consumer: A person who consumes data professionals' data insights and analytics to make data-driven decisions

Clickstream analytics

The tracking and analysis of visits to websites

```
// Output:
        Alan Turing
```

Continuous Application: An end-to-end application that reacts in real time to data

Data flow testing: The data flow is examined in relation to the variables used in the software program

Testbed: A piece of software equipment that is used for testing of a software application in a precise, transparent, and consistent manner

Test report: A report that contains a concise overview of the testing goals, methods, procedures and findings

Test deliverables: The reports or documents generated during the testing process

```
public class MyClass {

public static void main(String[] args) {

  System.out.println(-6 + 9 * 7); // Output: 57

  System.out.println((77+8) % 10); // Output: 5

  System.out.println(55 + -13*6 / 5); // Output: 40

  System.out.println(65 + 75 / 6 * 3 - 9 % 2); // Output: 100

  System.out.println((35.6 * 4.6 - 4.7 * 2.8) / (56.1 - 6.7));

  // Output: 3.048582995951417

}

}
```

Metric: A measurement unit used to evaluate and track performance over a period of time

Test matrix: A software testing equipment that records the overall quality, resources, concentrated effort, strategy, and time spent on each phase of software testing

Browser automation:

A method of automatically evaluating the functioning of a web application in a browser, in which a software program opens the web browser, traverses to the application, and performs actions, much like a normal user would

A use case →

Describes how a machine will execute a particular task if particular conditions are met

Data Engineer: A data professional in charge of getting the appropriate data into the hands of data scientists and analysts

Data Pipeline: A method of automating data flow across an enterprise organization by extracting, transforming and loading data to a destination for storage and analysis

Random Forest

A supervised learning approach that combines the outputs of several decision trees into a single model and is commonly employed in regression and classification issues

Blockchain: A data storage technology that makes it extremely difficult or impossible to edit, hack or fake the data − thereby making the data secure and unchangeable

Mixed Reality: A blend of physical and virtual worlds

Guided analytics

A developer-created analytics solution tailored to a specific business requirement

Summary Statistics: A branch of applied mathematics that provide a summary and explanation of your sample data

```
public class MyClass {

public static void main(String[] args) {

        System.out.println(" +\"\"\"\"\"+ ");

        System.out.println("+********+ ");

        System.out.println("[| @ @ |]");

        System.out.println(" (  <>   ) ");

        System.out.println(" { '='  } ");

        System.out.println("+########+ ");

 }

}
```

```
+"""""+

+********+

[| @ @ |]

(  <>   )

{ '='  }

+########+
```

Mobile Testing: The technique for evaluating mobile applications for functionality, accessibility, efficiency, reliability, security, consistency, and other factors

Market Mix Modeling:

A technique for determining how various promotional strategies influence a product's business Key performance metrics

Solution requirements: Specific features that a product must have in order to meet business requirements

Watson

A question-answering supercomputer system developed by IBM's DeepQA project and named after IBM founder Thomas J Watson

Cluster Analysis: The statistical method of finding similar groups of objects in order to form clusters

Endurance testing: A kind of performance testing used to evaluate the system's performance that is always in usage

Localization testing

Guarantees that the software application is suitable for use in that particular region

Benchmark Testing: The practice of evaluating an application's performance against an industry standard set by another enterprise organization

Baseline Testing: A kind of non-functional testing in which a series of tests are performed in order to gather software application's performance metrics

Gorilla Testing:

A software testing technique in which a software module is repeatedly evaluated using random inputs to confirm that it is working properly and that it has no flaws

Monkey testing:

A type of software testing technique that evaluates the behavior of the software application and validates whether it crashes or not based on certain random inputs without any predefined test cases

Pesticide paradox: Every bug-prevention or bug-finding method leaves a residue of more sophisticated bugs against which the methods are ineffective

Python Exercises

Data Refinement: The method of standardizing, categorizing, tagging and streamlining data to empower necessary business decisions

Question 1

```
print("A 6'2\" tall object")
# Output: A 6'2" tall object
print('A 6\'2" tall object')
# Output: A 6'2" tall object
```

Question:

Write a program to Add Two Numbers.

Solution:

```
a = 1
b = 2
c= a+b
print(c)
```

```
a = int(input("enter a number: "))
b = int(input("enter a number: "))
c= a+b
print(c)
```

ELIZA Effect

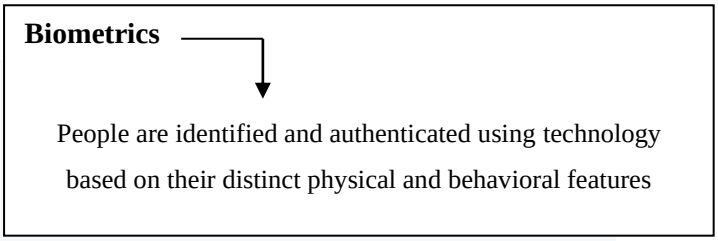
The tendency to unconsciously presume computer behaviors are comparable to human behaviors

Question 2

Question:

Write a program to find whether a given number (accept from the user) is even or odd, print out an appropriate message to the user.

Solution:




```
a = int(input("enter a number: "))
if a % 2 == 0:
    print("This is an even number.")
else:
    print("This is an odd number.")
```

Question 3

Question:

Write a program to check whether a number entered by the user is positive, negative or zero.

Solution:

```
a = int(input("Enter a number: "))
if a > 0:
```

```
    print("Positive number")
elif a == 0:
    print("Zero")
else:
    print("Negative number")
```

Comparative analysis: The process of comparing and contrasting 2 or more data sets in order to uncover and discover new insights about them

---

Question 4

Question:

Write a program to display the calendar of a given date.

Solution:

```
import calendar
yy = int(input("Enter year: "))
mm = int(input("Enter month: "))
print(calendar.month(yy, mm))
```

Data aggregation tools

Tools for combining data from numerous sources into a single location in order to prepare the combined data for data processing

---

Question 5

Question:

Write a program to ask the user to enter the string and print that string as output of the program.

Solution:

```
string = input("Enter string: ")
print("You entered:",string)
```

```
from string import Template

x = Template("$ab Turing")

print(x.substitute(ab="Alan"))

# Output: Alan Turing

print(x.substitute(ab="John"))

# Output: John Turing
```

Question 6

Question:

Write a program to Concatenate Two Strings.

Solution:

Data custodian: A person who is in charge of the administrative and operational aspects of a technical environment where data is stored

```
string1 = input("Enter first string to concatenate: ")
string2 = input("Enter second string to concatenate: ")
string3 = string1 + string2
print("String after concatenation = ",string3)
```

Question 7

Reporting Performance

The system's ability to generate reports quickly and effectively in response to queries from end users

Question:

Write a program to check if an item exists in the list.

Solution:

Key-value Database: A NoSQL database that stores data using a simple key-value method

```
list_of_items = ["ball", "book", "pencil"]
item = input("Type item to check: ")
if item in list_of_items:
print("Item exists in the list.")
else:
  print("Item does not exist in the list.")
```

Data feed: A way of getting updated information from data sources such as a Twitter feed or RSS

Question 8

Question:

Write a program to Join two or more lists.

Grid computing: A group of computing machines that collaborate to do a task that would be impossible for a single computing machine to complete

---

Solution:

```
list1 = ["This" , "is", "a", "sample", "program"]
list2 = [10, 2, 45, 3, 5, 7, 8, 10]
finalList = list1 + list2
print(finalList)
```

Deidentification: The process for preventing the disclosure of a person's personal information

---

Question 9

Question:

Write a program to Calculate Cube of a Number.

Graph Database

↓

A NoSQL database for storing, querying and modifying network graphs

---

Solution:

```
import math
a = int(input("Enter a number: "))
b=math.pow(a,3)
print (b)
```

Load balancing: The process of distributing incoming network traffic across numerous servers in an efficient manner

---

Question 10

Question:

Write a program to Calculate Square root of a Number.

Solution:

Fault-tolerant design

A system that is designed to keep running even if some of its components fail

```
import math
a = int(input("Enter a number: "))
b=math.sqrt(a)
print (b)
```

Question 11

Question:

Write a program that takes a list of numbers (for example, a = [5, 10, 15, 20, 25]) and makes a new list of only the first and last elements of the given list.

Solution:

```
a = [5, 10, 15, 20, 25]
print([a[0], a[4]])
```

Question 12

Question:

Take a list, say for example this one: a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89] and write a program that prints out all the elements of the list that are less than 5.

Solution:

Decision automation: The use of software to automate the decision-making process

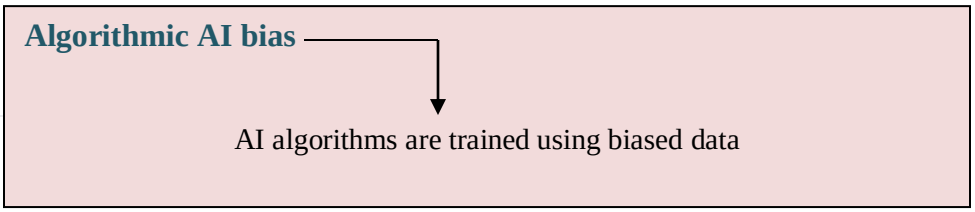
```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
for i in a:
    if i < 5:
        print(i)
```

Algorithmic learning theory

(Math about computer programs learning from experience)

A branch of statistical learning theory which aims to examine ML problems and algorithms

Question 13



Question:

Let's say I give you a list saved in a variable: a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]. Write one line of Python that takes this list 'a' and makes a new list that has only the even elements of this list in it.

Solution:

```
a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
b = [number for number in a if number % 2 == 0]
print(b)
```

Question 14

Question:

Ask the user for a string and print out whether this string is a palindrome or not (A palindrome is a string that reads the same forwards and backwards).

Solution:

```
a=input("Please enter a word: ")
c = a.casefold()
b = reversed(c)
```

```
if list(c) == list(b):
   print("It is palindrome")
else:
   print("It is not palindrome")
```

Question 15

Question:

Take two lists, say for example these two: a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89] b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes.

Solution:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
result = [i for i in set(a) if i in b]
print(result)
```

Question 16

Question:

Write a program to add a string to text file.

Haptics: The science of transmitting and understanding information via touching

Solution:

```
file = open("testfile.txt","w")
file.write("Hello World")
file.write("This is our new text file")
file.write("and this is another line.")
file.write("Why? Because we can.")
file.close()
```

Question 17

Question:

Write a program to read a file and display its contents on console.

Solution:

```
with open('testfile.txt') as f:
    line = f.readline()
    while line:
        print(line)
        line = f.readline()
```

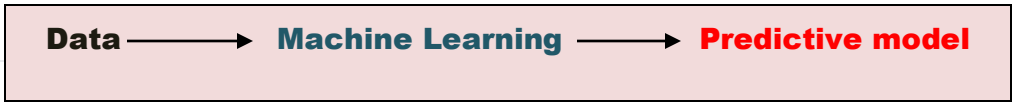
Abnormal end:

Abnormal termination of the execution of a program prior to completion

---

Data Curation: The organization and integration of data collected from various sources to address the needs and interests of a specific group of people

Question 18



Question:

Take two sets, say for example these two: a = {1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89} b = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13} and write a program that returns a set that contains only the elements that are common between the sets.

Solution:

```
a = {1, 1, 2, 2, 3, 5, 8, 13, 21, 34, 55, 89}
b = {1, 2, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13}
c = set(a) & set(b)
print(c)
```

Question 19

Question:

Write a program to split the characters of the given string into a list.

Solution:

```
s = "mystring"
l = list(s)
print (l)
```

Programmatic Advertising:

The automated process of buying and selling of online advertising

Question 20

Question:

Create a program that asks the user for a number and then prints out a list of all the divisors of that number.

Solution:

```
n=int(input("Enter an integer: "))
print("The divisors of the number are: ")
for i in range(1,n+1):
    if(n%i==0):
        print(i)
```

Mobile Device Testing: A testing technique for evaluating hardware, software and overall quality of a mobile device

Question 21

Question:

Write a program to Find the largest of three numbers.

Block bugs: Bugs that prevent other bugs from being fixed

Solution:

```
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
```

```
c = int(input("Enter third number: "))
if (a > b) and (a > c):
   largest = a
elif (b > a) and (b > c):
   largest = b
else:
   largest = c
print("The largest number is", largest)
```

Installation Testing: A Testing that is carried out to ensure that the software application has been installed properly with all of its inherent features and that the software application is functioning as intended

---

Question 22

Jailbreaking: A way to bypass all of Apple's restrictions and customize your iPhone whatever you like

Question:

Write a Program to Find Absolute value of a Number.

Solution:

Storage Testing: A kind of software testing that verifies if the software application saves necessary data in the appropriate folders and has enough space to avoid abrupt termination due to disc space constraints

```
num = int(input("Enter a number: "))
if num >= 0:
        print(num)
else:
        print(-num)
```

Sampling bias:

Imagine that there are **10 individuals** in a bakery and if you are asked to survey whether they like butter or jam. If you only surveyed the **3 individuals** and concluded that the majority of individuals like butter, you'd have exhibited a **sampling bias**

Question 23

Question:

Write a program to Find the length of a String.

Solution:

```
print("Enter 'y' for exit.")
string = input("Enter a string: ")
if string == 'y':
    exit()
else:
    print("Length of the string =", len(string))
```

Mobile usability testing

The goal is to make mobile devices as user-friendly as possible by evaluating them according to how they are utilized by end-users

---

Question 24

Question:

Write a program to Print Natural Numbers from 1 to N.

Solution:

```
N = int(input("Please Enter any Number: "))
for i in range(1, N+1):
    print (i)
```

Cloud-based mobile testing

Instead of using an on-premises environment, mobile applications are evaluated using an online cloud platform

Question 25

Question:

Write a program to calculate the sum and average of Natural Numbers from 1 to N.

Solution:

```
N = int(input("Please Enter any Number: "))
sum = 0
for i in range(1,N+1):
  sum = sum + i
print(sum)
average = sum / N
print(average)
```

Quantified Self: A trend toward using software applications and wearable smart devices to track one's every step throughout the day in order to obtain a better knowledge of one's behavior

Question 26

Question:

Write a program to Print a Statement Any Number of Times.

Solution:

```
n = int(input("Please Enter any Number: "))
for i in range(n):
    print("hello world")
```

Question 27

Question:

Write a program To Multiply Two Numbers Using Function.

Solution:

```
def my_function():
    a = int(input("enter a number: "))
    b=int(input("enter a number: "))
    c= a*b
    return c
d = my_function()
print (d)
```

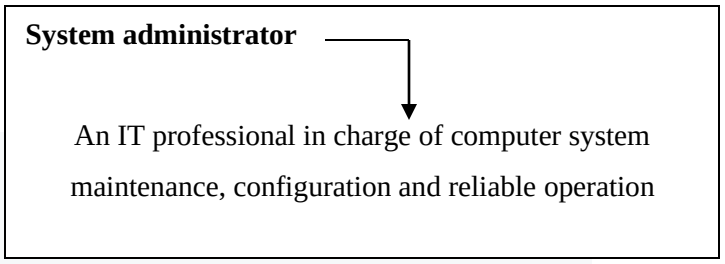
Question 28

Question:

Write a program To add an item to the end of the list.

Solution:

```
list1 = ["pen", "book", "ball"]
list1.append("bat")
print(list1)
```



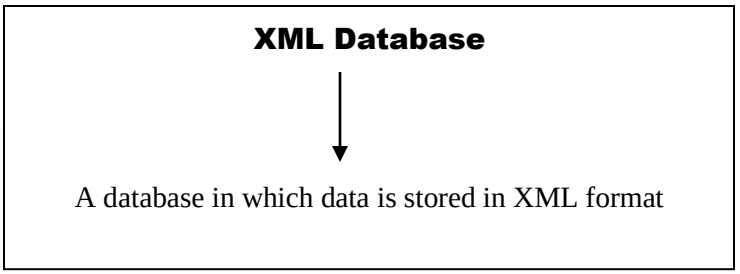
Question 29

Question:

Write a program To remove an item from the list.

Solution:

```
list1 = ["pen", "book", "ball"]
list1.remove("ball")
print(list1)
```



Question 30

Question:

Write a program To print the number of elements in an array.

Solution:

```
list1 = ["pen", "book", "ball"]
a = len(list1)
print(a)
```

Question 31

Question:

Write a program To calculate the variance and standard deviation of the elements of the list.

Solution:

```
import numpy as np
a= [2,6,8,12,18,24,28,32]
variance= np.var(a)
std = np.std(a)
print(variance)
print(std)
```

Performance testing: The process of determining how well a computing machine performs in terms of speed, responsiveness and stability under a specified workload

---

Question 32

Question:

Write a program to get the difference between the two lists.

Solution:

```
list1 = [4, 5, 6, 7]
list2 =  [4,  5]
print(list(set(list1) - set(list2)))
```

Question 33

Question:

Write a program to select an item randomly from a list.

Solution:


```
import random
list = ['Paper', 'Pencil', 'Book', 'Bag', 'Pen']
print(random.choice(list))
```

Question 34

Question:

Write a program that prints all the numbers from 0 to 6 except 2 and 6.

Solution:

```
for x in range(6):
    if (x == 2 or x==6):
        continue
    print(x)
```

Microservices: An architectural approach in which a single application is developed as a collection of small independent services that perform specific functions and communicate over well-defined APIs

---

Question 35

Question:

Write a program that takes input from the user and displays that input back in upper and lower cases.

Solution:

Method cascading

↓

Multiple methods can be called on the same object

```
a = input("What's your name? ")
print(a.upper())
print(a.lower())
```

Question 36

Synthetic monitoring:

A technique for evaluating application performance that imitates the steps users might take when interacting with a software application

Question:

Write a program to check whether a string starts with specified characters.

Solution:

```
string = "myw3schools.com"
print(string.startswith("w3s"))
```

Question 37

Question:

Write a program to create the multiplication table (from 1 to 10) of a number.

Solution:

Column-oriented database: A database in which data is stored by column rather than row

```
n = int(input("Enter a number: "))
for i in range(1,11):
   print(n,'x',i,'=',n*i)
```

Question 38

Question:

Write a program to check a triangle is equilateral, isosceles or scalene.

Solution:

Confabulation →

Making an intuition-based conclusion generally rely on data

```
print("Enter lengths of the triangle sides: ")
a = int(input("a: "))
b = int(input("b: "))
c = int(input("c: "))
if a == b == c:
    print("Equilateral triangle")
elif a==b or b==c or c==a:
    print("isosceles triangle")
else:
    print("Scalene triangle")
```

- **Hot data storage:** The storage of data that is accessed frequently.
- **Warm data storage:** The storage of data that is accessed less frequently
- **Cold data storage:** The storage of data that is rarely accessed

Prometheus

A freely available tool created to support tracking and reporting in cloud-native platforms, such as Kubernetes

Question 39

Question:

Write a program to sum of two given integers. However, if the sum is between 15 to 20 it will return 20.

Solution:

```
a = int(input("enter a number: "))
b = int(input("enter a number: "))
c= a+b
if c in range(15, 20):
        print (20)
else:
        print(c)
```

CRUD

(Create, Read, Update, Delete)

The four basic operations for manipulating the stored data

Incident Response

The procedure used by an enterprise organization to deal with a data leak or hack, as well as how the enterprise organization attempts to control the attack's or breach's effects

Question 40

Cross-channel analytics: The process of collecting, evaluating and presenting data from a wide range of marketing channels in one location in order to investigate how customers are impacted to purchase a product or service

Question:

Write a program to convert degree to radian.

Solution:

```
pi=22/7
degree = int(input("Input degrees: "))
radian  =  degree*(pi/180)
print(radian)
```

Database as a service

↓

A cloud-based database that is rented on a per-use basis

Question 41

Log Aggregation

Collecting and aggregating various event log files from numerous services, applications and other sites

Question:

Write a program to Generate a Random Number.

Solution:

Demographic data: Information on a human population's traits

```
import random
print(random.randint(0,9))
```

IIS Log Viewer

↓

An application that makes it easier to view log files from an IIS web server

Question 42

Distributed cache: A data cache that is distributed over numerous computer systems but functions as a single entity

Question:

Write a Program to find the semi-perimeter of triangle.

Solution:

```
a = int(input('Enter first side: '))
b = int(input('Enter second side: '))
c = int(input('Enter third side: '))
s = (a + b + c) / 2
print(s)
```

Event analytics: The process of collecting and analyzing an event that occurs when a visitor to your website performs an action

---

Question 43

Question:

Given a list of numbers, Iterate it and print only those numbers which are divisible of 2.

Solution:

```
List = [10, 20, 33, 46, 55]
for i in List:
    if (i % 2 == 0):
      print(i)
```

Flume: A framework for loading data into **Hadoop**

Question 44

Question:

Write a program to Multiply all numbers in the list.

Solution:

```
import numpy
list = [1, 2, 3]
result = numpy.prod(list)
print(result)
```

In-database analytics: Data analytics is integrated into the data warehouse operation

Question 45

Question:

Write a program to print ASCII Value of a character.

Solution:

```
a = 'j'
print("The ASCII value of '" + a + "' is", ord(a))
```

In-memory data grid (IMDG):

Data is stored in memory across numerous servers in order to speed up and scale services and applications

Question 46

Question:

Write a program to list files in a directory.

Solution:

```
# Import os module to read directory
import os

# Set the directory path
path = 'C:/Users/Manju/.spyder-py3/'

# Read the content of the file
files = os.listdir(path)

# Print the content of the directory
for file in files:
    print(file)
```

Question 47

Question:

Write a program to Read and Write File.

Solution:

```
#Assign the filename
filename = "languages.txt"
# Open file for writing
fileHandler = open(filename, "w")

# Add some text
fileHandler.write("Bash\n")
fileHandler.write("Python\n")
fileHandler.write("PHP\n")

# Close the file
fileHandler.close()

# Open file for reading
fileHandler = open(filename, "r")

# Read a file line by line
for line in fileHandler:
  print(line)

# Close the file
fileHandler.close()
```

```
# hex() - converts integer to hexadecimal
a=hex(14)
print(a)
```

Output on the screen:

0xe

```
# oct() - converts integer to octal
a=oct(14)
print(a)
```

Output on the screen:

0o16

```
a = [1, 2, 3, 4, 5]
b = a[::-1]
print(b)
```

Output on the screen:

[5, 4, 3, 2, 1]

Question 48

Question:

Write a program to add and search data in the dictionary.

Solution:

```
# Define a dictionary
customers = {'1':'Mehzabin Afroze','2':'Md. Ali',
'3':'Mosarof Ahmed','4':'Mila Hasan', '5':'Yaqub Ali'}

# Append a new data
customers['6'] = 'Mehboba Ferdous'

print("The customer names are:")
# Print the values of the dictionary
for customer in customers:
    print(customers[customer])

# Take customer ID as input to search
name = input("Enter customer ID:")

# Search the ID in the dictionary
for customer in customers:
    if customer == name:
        print(customers[customer])
        break
```

Test stubs

(A top-down testing strategy)

↓

Testing of upper-level code while lower-level code is still being created

Test drivers

(A bottom-up testing approach)

↓

Testing of lower-level code while upper-level code is still being created

---

Question 49

Multiplexing:

Over a single Transmission Control Protocol connection, numerous requests are delivered between browsers and servers at the same time

Question:

Write a program to add and search data in the set.

Solution:

```
# Define the number set
numbers = {23, 90, 56, 78, 12, 34, 67}

# Add a new data
numbers.add(50)
# Print the set values
print(numbers)

message = "Number is not found"

# Take a number value for search
search_number = int(input("Enter a number:"))
# Search the number in the set
for val in numbers:
    if val == search_number:
        message = "Number is found"
        break

print(message)
```

Low code development platform	No-code development platform
A software development platform that enables developers to create new business applications visually and with little coding	A software development platform that enables developers to create new business applications visually without writing a single line of code

Decision Science:

↓

A set of analytical and behavioral methodologies for making decisions based on data with the goal of solving business challenges

---

Question 50

Question:

Write a program to demonstrate throw and catch exception.

Solution:

Metadata repository →

↓

A software tool designed to store and share metadata

```
# Try block
try:
```

```
    # Take a number
    number = int(input("Enter a number: "))
    if number % 2 == 0:
        print("Number is even")
    else:
        print("Number is odd")

# Exception block
except (ValueError):
  # Print this message
  print("Enter a numeric value")
```

If the entered value is not a number

Statistical power

(The power of a hypothesis test)

The probability of correctly rejecting the null hypothesis when the null hypothesis is false

---

Question 51

Question:

Write a program to illustrate password authentication.

Weka: A set of ML algorithms and tools for data mining

Solution:

```
# import getpass module
import getpass

# Take password from the user
passwd = getpass.getpass('Password:')

# Check the password
if passwd == "python":
    print("You are authenticated")
else:
```

Data observability: An enterprise organization's ability to monitor, record and make suggestions about what's going on inside its data systems in order to keep them healthy and minimize downtime

```
    print("You are not authenticated")
```

Question 52

Question:

Write a program to calculate the average of numbers in a given list.

Solution:

```
n=int(input("Enter the number of elements to be inserted: "))
a=[]
for i in range(0,n):
    x=int(input("Enter element: "))
    a.append(x)
avg=sum(a)/n
print("Average of elements in the list: ",round(avg,2))
```

Data onboarding: The process of collecting and uploading offline data for marketing purposes into an online environment

---

Question 53

Question:

Write a program to exchange the values of two numbers without using a temporary variable.

Convergence: A point at which extra data training will no longer improve the **ML model**

Solution:

```
a=int(input("Enter value of first variable: "))
b=int(input("Enter value of second variable: "))
a=a+b
b=a-b
a=a-b
print("a is:",a," b is:",b)
```

Automated testing:

The process of using an automation tool to check a software application for bugs

Question 54

Manual testing:

The method of manually checking software application for bugs

Question:

Write a program to reverse a given number.

Solution:

```
n=int(input("Enter a multi-digit number: "))
rev= str(n)[::-1]
print("Reverse of the number:", rev)
```

Role-Based Access Control

A technique for limiting network access depending on the functions of specific individuals within an enterprise organisation

Question 55

Question:

Write a program to take in the marks of 5 subjects and display the grade.

Solution:

```
sub1=int(input("Enter marks of the first subject: "))
sub2=int(input("Enter marks of the second subject: "))
sub3=int(input("Enter marks of the third subject: "))
sub4=int(input("Enter marks of the fourth subject: "))
sub5=int(input("Enter marks of the fifth subject: "))
avg=(sub1+sub2+sub3+sub4+sub4)/5
if(avg>=90):
    print("Grade: A")
elif(avg>=80 and avg<90):
    print("Grade: B")
elif(avg>=70 and avg<80):
    print("Grade: C")
elif(avg>=60 and avg<70):
    print("Grade: D")
else:
    print("Grade: F")
```

- Absolute error = (Experimental value − Theoretical value)
- Relative error = $\frac{(\text{Experimental value} - \text{Theoretical value})}{\text{Theoretical value}}$
- Percent error = $\frac{(\text{Experimental value} - \text{Theoretical value})}{\text{Theoretical value}} \times 100$

Law of large numbers

↓

The **percentage difference** between the experimental and theoretical values goes to zero − if we repeat the same experiment a large number of times

Question 56

General Availability

The time at which software product is made widely accessible for purchase by anyone who wants it

Question:

Write a program to print all numbers in a range divisible by a given number.

Solution:

```
lower=int(input("Enter lower range limit:"))
upper=int(input("Enter upper range limit:"))
n=int(input("Enter the number to be divided by:"))
for i in range(lower,upper+1):
    if(i%n==0):
        print(i)
```

Enable: Make it functional

Function key: A button on a computer keyboard that performs a specific set of actions when it is pressed

---

Question 57

Question:

Write a program to read two numbers and print their quotient and remainder.

Solution:

Flag: Selecting an information item for further processing by marking it

```
a=int(input("Enter the first number: "))
```

```
b=int(input("Enter the second number: "))
quotient=a//b
remainder=a%b
print("Quotient is:",quotient)
print("Remainder is:",remainder)
```

File server: A computer that stores and manages data files so that they can be accessed by other computers on the same network

---

Question 58

Question:

Write a program to accept three distinct digits and print all possible combinations from the digits.

Solution:

```
a=int(input("Enter first number:"))
b=int(input("Enter second number:"))
c=int(input("Enter third number:"))
d=[]
d.append(a)
d.append(b)
d.append(c)
for i in range(0,3):
    for  j   in   range(0,3):
        for k in range(0,3):
            if(i!=j&j!=k&k!=i):
                print(d[i],d[j],d[k])
```

Upstream

Data sent from local computer to the server

(Uploading)

Downstream

Data sent from server to the local computer

(Downloading)

Question 59

Question:

Escalate: Take a disputed matter to a higher level of management for settlement

Write a program to print odd numbers within a given range.

Solution:

```
lower=int(input("Enter the lower limit for the range:"))
upper=int(input("Enter the upper limit for the range:"))
for i in range(lower,upper+1):
    if(i%2!=0):
        print(i)
```

Error log: A file in a computer system where information about critical errors that are encountered by the software application is stored for later access

Fix: A solution to a software issue

Security Remediation:

The process of identifying vulnerabilities and taking the appropriate action to address them

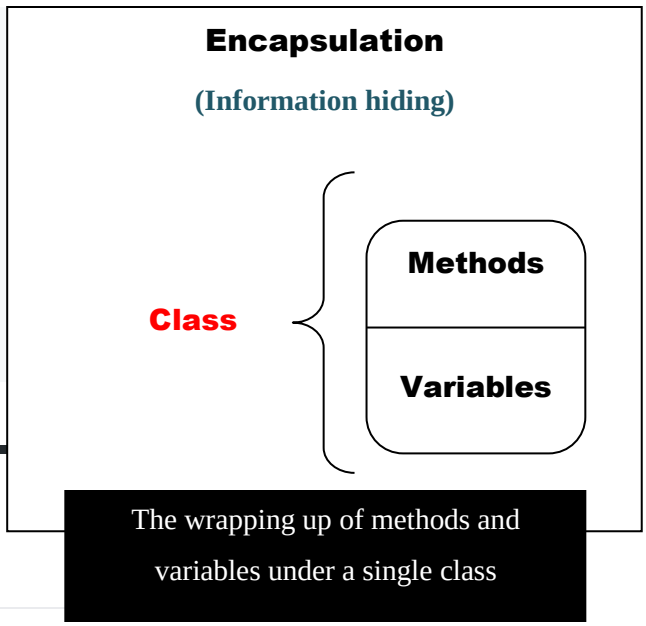
Question 60

Question:

Write a program to find the smallest divisor of an integer.

Solution:

```
n=int(input("Enter an integer:"))
a=[]
for i in range(2,n+1):
    if(n%i==0):
        a.append(i)
a.sort()
print("Smallest divisor is:",a[0])
```



Question 61

Question:

Write a program to count the number of digits in a number.

Solution:

```
n=int(input("Enter number:"))
count=0
while(n>0):
    count=count+1
    n=n//10
print("The number of digits in the number are:",count)
```

Error recovery procedures: Procedures for isolating and, if possible, recovering from system errors in order to greatly boost the availability of the system services

Virtualization	Emulation
Hardware can be accessed directly	We need a software connector to access hardware

Question 62

Exception: An abnormal condition that occurs during the execution of a software program that causes the normal flow of the program's instructions to be disrupted

Question:

Write a program to read a number n and print and compute the series "1+2+...+n=".

Solution:

```
n=int(input("Enter a number: "))
a=[]
for i in range(1,n+1):
    print(i,sep=" ",end=" ")
    if(i<n):
        print("+",sep=" ",end=" ")
    a.append(i)
print("=",sum(a))

print()
```

Facsimile machine: A machine which transmits and receives documents over the telephone system using fax machines

Fault: An unintentional condition in which a system fails to accomplish its intended function

Output:

```
Enter a number: 4 # entered number
1 + 2 + 3 + 4 = 10
```

Question 63

Graphical interchange format: A digital format for compressing and transferring graphical data via computer networks

Question:

Write a program to read a number n and print the natural numbers summation pattern.

Solution:

```
n=int(input("Enter a number: "))
for j in range(1,n+1):
    a=[]
    for i in range(1,j+1):
        print(i,sep=" ",end=" ")
        if(i<j):
            print("+",sep=" ",end=" ")
        a.append(i)
    print("=",sep(a))

print()
```

Fault Management

1. Detect the Fault
2. Isolate the Fault
3. Notify the administrators about the Fault
4. Identify the root cause of the Fault
5. Resolve the Fault

Output:

```
Enter a number: 4 # entered number
1 = 1
1 + 2 = 3
1 + 2 + 3 = 6
1 + 2 + 3 + 4 = 10
```

Question 64

Question:

Write a program to read a number n and print an identity matrix of the desired size.

Solution:

Forum: An online discussion site where people can share their thoughts and opinions on a specific topic

```
n=int(input("Enter a number: "))
for i in range(0,n):
```

```
    for j in range(0,n):
        if(i==j):
            print("1",sep=" ",end=" ")
        else:
            print("0",sep=" ",end=" ")
    print()
```

Question 65

Question:

Write a program to read a number n and print an inverted star pattern of the desired size.

Solution:

```
n=int(input("Enter number of rows: "))
for i in range (n,0,-1):
    print((n-i) * ' ' + i * '*')
```

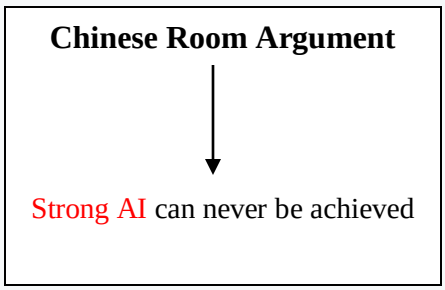
Question 66

Question:

Write a program to print prime numbers in a range using Sieve of Eratosthenes.

Solution:

```
n=int(input("Enter upper limit of range: "))
sieve=set(range(2,n+1))
while sieve:
    prime=min(sieve)
    print(prime,end="\t")
    sieve-=set(range(prime,n+1,prime))

print()
```



Question 67

Question:

Write a program to find the largest number in a list.

Solution:

```
a=[]
n=int(input("Enter number of elements:"))
for i in range(1,n+1):
    b=int(input("Enter element:"))
    a.append(b)
a.sort()
print("Largest element is:",a[n-1])
```

Operational effectiveness: The plan of action that a **company** puts in place in order to arrive at a better final product or service

---

Question 68

Question:

Write a program to find the second largest number in a list.

Solution:

```
a=[]
n=int(input("Enter number of elements:"))
for i in range(1,n+1):
    b=int(input("Enter element:"))
    a.append(b)
a.sort()
print("Second largest element is:",a[n-2])
```

Perplexity = $2^{-\text{cross entropy}}$

↓

A measure of how well a model is accomplishing its goal

Question 69

Appropriateness Recognizability: The extent to which users can identify whether a product is suitable for their needs

Question:

Write a program to put the even and odd elements in a list into two different lists.

Solution:

```
a=[]
n=int(input("Enter number of elements:"))
for i in range(1,n+1):
    b=int(input("Enter element:"))
    a.append(b)
even=[]
odd=[]
for j in a:
    if(j%2==0):
        even.append(j)
    else:
        odd.append(j)
print("The even list",even)
print("The odd list",odd)
```

Hyperautomation

↓

(Robotic Process Automation + ML + AI)

The organizations use to rapidly automate complex business and IT processes

---

Question 70

Question:

Write a program to sort the list according to the second element in the sublist.

Solution:

```
a=[['A',34],['B',21],['C',26]]
for i in range(0,len(a)):
    for j in range(0,len(a)-i-1):
        if(a[j][1]>a[j+1][1]):
            temp=a[j]
            a[j]=a[j+1]
            a[j+1]=temp

print(a)
```

SnowBall: A data transport service that allows us to move terabytes of data both inside and outside of the AWS cloud

Output:

```
[['B', 21], ['C', 26], ['A', 34]]
```

Question 71

Question:

Write a program to find the second largest number in a list using bubble sort.

The simplest sorting algorithm, which continuously swaps nearby elements if they are out of sequence

Solution:

```
a=[]
n=int(input("Enter number of elements:"))
for i in range(1,n+1):
    b=int(input("Enter element:"))
    a.append(b)
for i in range(0,len(a)):
    for j in range(0,len(a)-i-1):
```

Geo-Targeting: A method for advertisers to define geographical location where their advertising should appear

```
        if(a[j]>a[j+1]):
            temp=a[j]
            a[j]=a[j+1]
            a[j+1]=temp
print('Second largest number is:',a[n-2])
```

 # Output:

```
Enter number of elements:4 # entered number
Enter element:15 # entered number
Enter element:8 # entered number
Enter element:6 # entered number
Enter element:9 # entered number
Second largest number is: 9
```

Question 72

Question:

Write a program to sort a list according to the length of the elements.

Solution:

```
a=[]
n=int(input("Enter number of elements:"))
for i in range(1,n+1):
    b=input("Enter element:")
    a.append(b)
a.sort(key=len)
print(a)
```

 # Output:

```
Enter number of elements:4 # entered number
Enter element:10 # entered number
Enter element:8 # entered number
Enter element:6 # entered number
Enter element:11 # entered number
['8', '6', '10', '11']
```

Pair Testing

Software testing practice in which 2 individuals test the same feature at the same time on the same computer by continuously interchanging ideas and feedback

Question 73

Conversational AI

↓

(Natural Language Processing + ML)

Makes machines capable of understanding, processing and responding to human language

Question:

Write a program to create a list of Tuples with the first element as the number and the second element as the square of the number.

Solution:

Conversational UI: A user interface that enables computers to interact with humans using voice or text

```
l_range=int(input("Enter the lower range:"))
u_range=int(input("Enter the upper range:"))
a=[(x,x**2) for x in range(l_range,u_range+1)]
print(a)
```

```
# Output:
Enter the lower range:2 # entered number
Enter the upper range:6 # entered number
[(2, 4), (3, 9), (4, 16), (5, 25), (6, 36)]
```

Question 74

Call Attribution:

Linking details of certain marketing campaigns to phone calls

Question:

Write a program to create a list of all numbers in a range which are perfect squares and the sum of the digits of the number is less than 10.

Solution:

```
l=int(input("Enter lower range: "))
u=int(input("Enter upper range: "))
a=[]
a=[x for x in range(l,u+1) if (int(x**0.5))**2==x and sum(list(map(int,str(x))))<10]
print(a)
```

 # Output:

```
Enter lower range: 2 # entered number

Enter upper range: 6 # entered number

[4]
```

Question 75

Question:

Write a Program to check whether the given number is valid mobile number or not.

Solution:

```
import re
num=input("Enter a mobile number:")
x=re.fullmatch("[7-9]\d{9}",num)
if x!= None:
   print("mobile number is valid")
else:
   print("mobile number is not valid")
```

Conversion Attribution

A method of determining the channel through which a user entered a website or application and carried out the desired action

---

Connection String:

A string that describes a data source's details and how to connect to it

Question 76

Cognitive Automation:

The process of extracting information from unstructured data and deriving meaningful conclusion

Robotic Process Automation

Automating repetitive manual tasks like data entry

Question:

Write a program to generate random numbers from 1 to 20 and append them to the list.

Solution:

Marketing Qualified Lead

A lead that the marketing team believes has a higher chance of becoming a customer than the others

```
import random
a=[]
n=int(input("Enter number of elements:"))
for j in range(n):
    a.append(random.randint(1,20))
print('Randomized list is: ',a)
```

 # Output:

```
Enter number of elements:6 # entered number
Randomized list is:  [19, 14, 3, 13, 6, 7]
```

Question 77

Question:

Write a Program to check whether the given mail id is valid Gmail id or not.

Solution:

```
import re
x=input("Enter your Gmail address:")
i=re.fullmatch("\w[a-zA-Z0-9_.]*@gmail[.]com", x)
if i!=None:
 print("A valid email address");
else:
 print("A invalid email address")
```

 # Output:

```
Enter your Gmail address:manjunath5496@gmail.com # entered mail address
A valid email address
```

Question 78

```
x = -2
while x < 0:
  print(x)
  x = x + 1
print('Python!')

# Output:
                -2
                -1
              Python!
```

Question:

Write a program to Convert Bytes to a String.

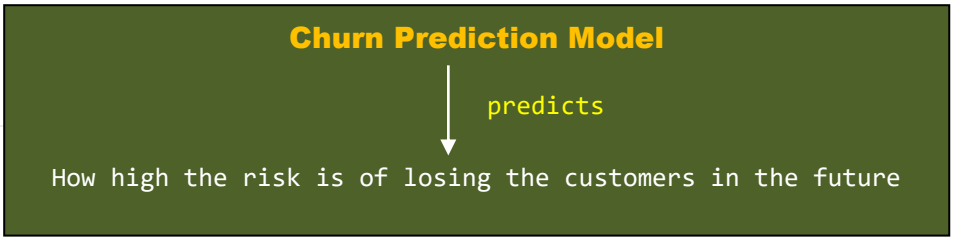
Solution:

```
print(b'Python \xE2\x9C\x85'.decode("utf-8"))

# Output: Python ✓
```

```
[manju@localhost ~]$ cat hello.py
print('Hello world!')
[manju@localhost ~]$ python hello.py
Hello world!
```

```
a = 6
b = 0
print(a >= 2 and b != 0 and (a/b) > 2)
# Output: False
```

Question 79



Question:

Write a program to remove the duplicate items from a list.

Solution:

```
a=[]
n= int(input("Enter the number of elements in list:"))
for x in range(0,n):
    element=int(input("Enter element" + str(x+1) + ":"))
    a.append(element)
b = set()
unique = []
for x in a:
    if x not in b:
        unique.append(x)
        b.add(x)
print("Non-duplicate items:")
print(unique)
```

Brain-computer interface: Technology that sends and receives signals between the human brain and a computer

```
# Output:

Enter the number of elements in list: 4 # entered number

Enter element [1]:5 # entered number

Enter element [2]:5 # entered number

Enter element [3]:4 # entered number

Enter element [4]:2 # entered number

Non-duplicate items:
[5, 4, 2]
```

Question 80

(Digital Transformation + Design strategy)

↓

Experience Transformation

Question:

Write a program to take input from the user until they type alan.

Solution:

```
while True:
   x = input('>> ')
   if x == 'alan':
      break
   print(x)
print('Done!')
```

Output:

```
>> albert (# entered)
albert

>> alan (# entered)
Done!
```

Cybernetics: The science of studying

- how information is communicated in the machine
- how information is communicated in the human brain

```
Comparing the way machine and human brain works and creating
machines that perform task or think like humans
```

Question 81

Question:

Write a program to only count digits that are either 0 or 2 in a positive integer

---

Solution:

```
x = 2896510
i = 0
while x > 0:
    a = x % 10
    if a == 0 or a == 2:
       i = i + 1
    x = x // 10
print(i)
```

Hard error: An error that corrupts system files and causes a failure of an operating system without providing any recovery options

Handheld device: Any portable computing device that can be carried and held in one's hand

Hook: A location in a software program where an instruction for performing a certain function is inserted

```
i = 0
while i < 3:
   print(f"index {i}")
   i = i+1

# Output:
                index 0
                index 1
                index 2
```

```
i = 0
while True:
   print(f"index {i}")
   if i > 2:
      break
   i= i + 1
```

Output:

```
index 0
index 1
index 2
index 3
```

Net Promoter Score

An evaluation of customer satisfaction and loyalty that is based on a 0 –10 scale that asks clients whether they would suggest your goods or services to others

Question 82

Question:

Write a program to print the multiples of 3 all on one line

Solution:

```
for x in range(1, 6):
  print(3 * x, end=" ")
print()

# Output: 3 6 9 12 15
```

```
for x in [22, 24, 27, 28, 30]:
    if x % 2 == 1: # If the number is odd
      break # ... immediately exit the for loop
    print(x)
print("Done!")

# Output:
              22
              24
              Done!
```

Hover help: Help information that pop-ups when a mouse pointer hovers over a specific area of a computer screen for a predetermined amount of time

```
for x in [23, 25, 28, 29, 30]:
    if x % 2 == 0: # If the number is even
      break # ... immediately exit the for loop
    print(x)
print("Done!")

# Output:
              23
              25
              Done!
```

```
print("Hi", "Alan Turing", sep="---")

# Output: Hi---Alan Turing
```

```
x = 0
while True:
    i = input("\n Enter a number: ")
    if i == "0" or i == "-1":
       break
    x = x + int(i)
print("The sum of the numbers you entered is: ", x)

# Output:
             Enter a number: 2  # entered number
             Enter a number: 3  # entered number
             Enter a number: 0  # entered number
             The sum of the numbers you entered is: 5
                          2 + 3 + 0 = 5
```

Hung terminal

If the previous work session is still visible on the display screen but the terminal does not respond to keyboard input, it is termed 'hung'

Immediate access storage: A storage device that houses all of the data and programs that the CPU is currently using

```
for x in [13, 16, 19, 14, 39, 40]:
    if x % 2 == 0: # If the number is even
       continue # Skip the number and continue
    print(x)
print("Done!")

# Output:
```

```
13
19
39
Done!
```

Question 83

Hand off: Get rid of an unwanted task

Question:

Write a program to check if there is an odd number in a given list of numbers

---

Solution:

```
n = [18, 26, 28, 4, 6]
i = 0
for n in n:
    if n % 2 == 1:
       i = i+1 # Count the odd numbers
if i > 0:
   print(True)
else:
   print(False)

# Output: False
```

- If data is being put onto the server, it's inbound transfer
- If data is being requested from the server, it's outbound transfer

Information appliance:

Any portable computing device with the ability to transfer information to and from another device

```
print(max(2 * 22, 5**2, 420 - 10, 5026**0))
# Output: 410

print(abs(13 - 100) + 20)
# Output: 107
```

```
x = "XYZ"
y= "ohn"

for i in x:
 print(i + y)
```

Output:

Xohn
Yohn
Zohn

```
x = "Albert"
print(len(x)) # Output: 6
print(x[len(x)-1]) # Output: t
print(x[len(x)-2]) # Output: r
print(x[len(x)-3]) # Output: e
print(x[len(x)-4]) # Output: b
print(x[len(x)-5]) # Output: l
print(x[len(x)-6]) # Output: a
```

```
x = "Alan"
print("O" + x[1:])
# Output: Olan

print("bert" in "albert") # Output: True
print("" in "albert") # Output: True
```

Question 84

```
x=len(["albert einstein", 6, ["1905", "1915", "Papers"], [4, 5, 8]])
print('The length of the list is: ', x )
# Output: The length of the list is:  4
```

Question:

Write a program to measure the time elapsed in executing a code.

Solution:

```
import time

x = time.time() # start time
print("Hello") # code
y = time.time() # end time
print(y - x) # prints the execution time
```

Task force: A team of experts and support staff brought together to solve a specific problem or achieve a specific objective

Transcoding: The process of conversion of a video or an audio file from one format to another

```
print("%.3f" % 2.58967) # Output: 2.590
print("|{0:>10}||{0:<10}|".format('A','L'))
# Output: |         A||A         |
print("|{0:*>10}||{0:#<10}|".format('A','L'))
# Output: |*********A||A##########|
```

Question 85

```
x=["albert einstein", 6, ["1905", "1915", "Papers"], [4, 5, 8]]
print(x[2])
# Output: ['1905', '1915', 'Papers']
```

Question:

Write a program to print all the multiples of 4 between 0 and 20.

Solution:

```
for i in range(21):
  if i % 4 == 0:
     print(i)
```

Output:

```
0
4
8
12
16
20
```

Value add: A enhancement in functionality or performance of a software application

Wallet: Software application that allows a user to make accepted payments to verified merchants over internet – as well as manage payment card accounts and purchases

```
x = {}
for i in "Alan":
  print(x.get(i, 4))
```

Output:

```
4
4
4
4
```

```
x = {}
for i in "Ala":
  print(x.get(i, 4))
```

Output:

```
4
4
4
```

```
from math import  sqrt
print(sqrt(4))
# Output: 2.0

print ("Albert", 5 + 3 / 2)
# Output: Albert 6.5

print ("What is 2 + 16?", 2 + 16)
# Output: What is 2 + 16? 18

print("Is it greater?", 15 >13)
# Output: Is it greater? True
```

```
x = False
y = "Is -5 greater than 3? %r"
print(y % x)
# Output: Is -5 greater than 3? False
```

Volatile storage: A computer memory that requires power to preserve stored data

Stored data is lost when power is switched off

```
x = 'Albert'
y = 'Einstein'
print(x+y) # Output: AlbertEinstein
print(x, y) # Output: Albert Einstein
```

Question 86

Wild duck: An individual who is creative and technical and does unusual things

```
x = "%r %r %r"
print(x % (11, 12, 13)) # Output: 11 12 13
print(x % ("A", "B", "C")) # Output: 'A' 'B' 'C'
```

Question:

Write a program to compute simple interest given all the required values.

Solution:

```
x=float(input("Enter the principal amount:"))
y=int(input("Enter the time in years:"))
z=float(input("Enter the rate of interest:"))
simple_interest=(x*y*z)/100
print("The simple interest is:", simple_interest)
```

Well-behaved application program: An application program that runs without causing network disruption

---

Nice to have: It's a good idea, but it's not required

Question 87

```
print(list(range(3, 0, -1)))
# Output: [3, 2, 1]
```

Question:

Write a program to check whether a given year is a leap year or not.

Solution:

```
x=int(input("Enter a year:"))
if(x%4==0 and x%100!=0 or x%400==0):
    print("The year is a leap year!")
else:
    print("The year isn't a leap year!")
```

Workaround: Making something operate despite the problem without completely solving it

---

Question 88

Roll out: Introduce a new product or service

```
print(2,000,000)
# Output: 2 0 0
```

Question:

Write a program to print out all the positions in a list where it finds negative numbers.

```
a = [21, 23, -27, 24, -29, -25, 24]
for x in range(len(a)):
 if a[x] < 0:
     print("Found a negative number at index: ", x)

# Output:     Found a negative number at index:  2
              Found a negative number at index:  4
              Found a negative number at index:  5
```

Predictive Analytics

(What will happen?)

Collect data → Clean data → Identify patterns → Make predictions

```
# python program to print out the reciprocal of each number in the list
a = [2.0, 4.0, 6.0]
for i in a:
  print(1 / i)
```

Output:

```
0.5
0.25
0.16666666666666666
```

Question 89

Mobile analytics: A technique especially created to track and comprehend how users engage with a mobile application in order to better understand their behavior and the functionality of the application

Question:

Write a program to generate all the divisors of an integer.

Solution:

```
x=int(input("Enter a number: "))
print("All divisors of", x, "are:")
for i in range(1,x+1):
    if(x%i==0):
      print(i)
```

Output:

```
Enter a number: 4
All divisors of 4 are:
1
2
4
```

Question 90

```
print("Main(\\return)")
# Output: Main(\return)
```

Question:

Write a program to print the table of a given number.

Solution:

```
print("Alan \" Turing")
# Output: Alan " Turing
```

```
a=int(input("Enter a number:"))
for i in range(1,11):
    print(a,"x",i,"=",a*i)
```

Protocol analyzer: A hardware or software tool that allows recording of network traffic and analyzing it in order to detect network traffic issues or suspected fraudulent activity

---

Question 91

Omnichannel Retailing:

The process through which many platforms are combined and used to impact a customer's decision to buy

Question:

Write a program to check if a number is an Armstrong number.

Solution:

Network troubleshooting: A technique for detecting, diagnosing, and resolving problems in a computer network and improving network performance

```
n=int(input("Enter a number: "))
a=list(map(int,str(n)))
b=list(map(lambda x:x**3,a))
if(sum(b)==n):
```

```
    print("The number is an Armstrong number. ")
else:
    print("The number isn't an Armstrong number. ")
```

Output:

```
Enter a number: 153 # 1³ + 5³ + 3³ = 153
The number is an Armstrong number.

Enter a number: 370 # 3³ + 7³ + 0³ = 370
The number is an Armstrong number.
```

Question 92

Question:

Write a program to print the real and imaginary parts of a complex number.

Solution:

```
a = (5+10j)
print(a.real)
# Output: 5.0
print(a.imag)
# Output: 10.0
```

Cloudstorming: The gathering of multiple cloud computing environments

```
print(int(2e2))
# Output: 200
print(int(2.0e-8))
# Output: 0
print(float(2.0e-8))
# Output: 2e-08
```

```
a = [21, 22, 23, 24]
a[len(a):] = [25, 26, 27]
print(a)
# Output: [21, 22, 23, 24, 25, 26, 27]
```

```
a = [21, 22, 23, 24]
a[:0] = [-11, 0]
print(a)
# Output: [-11, 0, 21, 22, 23, 24]

a = [21, 22, 23, 24]
a[1:-1] = []
print(a)
# Output: [21, 24]
a.insert(0, "1905 Papers")
print(a)
# Output: ['1905 Papers', 21, 24]
```

Question 93

Self-service business intelligence

A methodology of data analytics that makes it possible for business consumers to connect and use business data even if they lack experience with business intelligence, data mining or statistical analysis

Question:

Write a program to check if a number is a Perfect number.

Solution:

- **Veracity** → The quality of data
- **Variability** → The inconsistency of the data
- **Variety** → The type and nature of the data

```
x = int(input("Enter a number: "))
sum = 0
for i in range(1, x):
    if(x % i == 0):
        sum = sum + i
if (sum == x):
    print("The number is a Perfect number!")
else:
    print("The number is not a Perfect number!")
```

```
a = [21, 22, 23, 24]
a.insert(-2, "1905 Papers")
print(a)
# Output: [21, 22, '1905 Papers', 23, 24]
```

Question 94

```
i = 'A B C D'
print(i.split(' ', 1))
# Output: ['A', 'B C D']
print(i.split(' ', 2))
# Output: ['A', 'B', 'C D']
print(i.split(' ', 3))
# Output: ['A', 'B', 'C', 'D']
```

```
p = "~a ^ (b % c)"
i = p.maketrans("~^()", "!&[]")
print(p.translate(i))
# Output:!a & [b % c]
```

Question:

Write a program to find the LCM of two numbers.

Solution:

```
a=int(input("Enter the first number:"))
b=int(input("Enter the second number:"))
if(a>b):
    min=a
else:
    min=b
while(1):
    if(min%a==0 and min%b==0):
        print("LCM is:",min)
        break
    min=min+1
```

```
print('albert'.capitalize())
print('Einstein1905'.isalnum())
print('1905'.isdigit())
print('1905'.isnumeric())
```

Output on the screen:

```
Albert
True
True
True
```

Question 95

```
a = "www.snapdeal.com"
print(a.strip("w"))
# Output: .snapdeal.com
print(a.strip("com"))
# Output: www.snapdeal.
print(a.strip(".wmoc"))
# Output: snapdeal
```

Question:

Write a program to find the GCD of two numbers.

Solution:

```
import math
a=int(input("Enter the first number:"))
b=int(input("Enter the second number:"))
print("The GCD of the two numbers is", math.gcd(a,b))
```

```
print("alan".replace("a", "+*&"))
# Output: +*&l+*&n
```

Question 96

```
a = [2, 4, 6, 8]
b = []
for i in a:
    b.append(i * i)
print(b)

# Output: [4, 16, 36, 64]
```

```
a = [2, 4, 6, 8]
b = [i * i for i in a]
print(b)

# Output: [4, 16, 36, 64]
```

Question:

Write a program to keep count of the list:

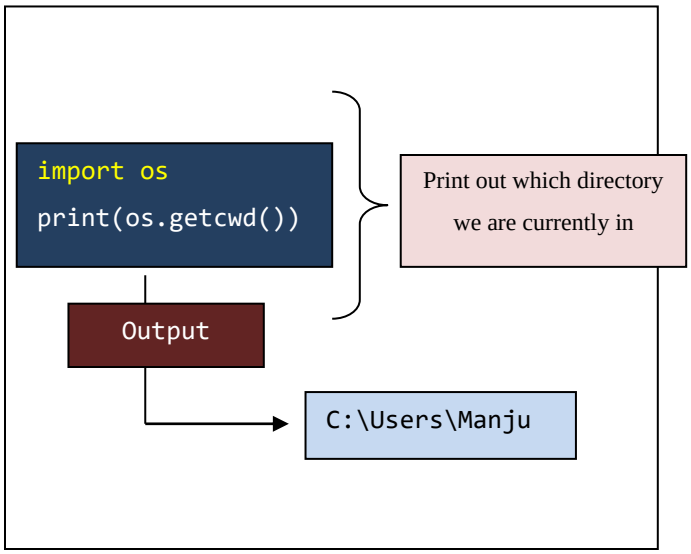
["Alan", "Thomas", "Rider", "Albert", "James"] starting from 11.

Solution:

```
x = ["Alan", "Thomas", "Rider", "Albert", "James"]
# set the start index to 11
for a,b in enumerate(x,11):
      print (a,b)
```

Output:

```
11 Alan
12 Thomas
13 Rider
14 Albert
15 James
```




```
x = ['Einsteinian', 1915, "Papers"]
print("The %s: %s" % ("Albert", x))

# Output:
 The Albert: ['Einsteinian', 1915, 'Papers']

b = {11: 'Eleven', 12: 'Twelve'}
a = {0: 'zero', 11: 'eleven'}
a.update(b)
print(a)

# Output:
   {0: 'zero', 11: 'Eleven', 12: 'Twelve'}
```

Question 97

Big Data Visualization: The **graphical representation** of **big data** to give the individual the capability to understand trends, define plans and make better decisions

Question:

Write a program to check if two numbers are amicable numbers.

Solution:

```
x=int(input('Enter a number: '))
y=int(input('Enter a number: '))
sum1=0
sum2=0
for i in range(1,x):
    if x%i==0:
        sum1+=i
for j in range(1,y):
    if y%j==0:
        sum2+=j
if(sum1==y and sum2==x):
   print( x, 'and', y, 'are Amicable numbers')
else:
   print(x, 'and', y, 'are not Amicable numbers')

# Output:
Enter a number: 220 # entered number
Enter a number: 284 # entered number
220 and 284 are Amicable numbers
```

Divisors of 220 = 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110

1+2+4+5+10+11+20+22+44+55+110=284

Divisors of 284 = 1, 2, 7, 71, 142

1+2+7+71+142=220

2 numbers are said to be "Amicable numbers" if each is equal to the sum of the divisors of the other

Streaming analytics:

The processing and analysis of data from **sources**

that continuously sends data

Voice Assistant:

The **digital assistant** that uses

(Natural Language Processing + Voice Recognition + Voice Synthesis)

to perform a variety of actions after hearing a **wake word** or **command** from the end user

Question 98

```
a = [0]
b = [a, 1]
print(a is b[0])
# Output: True
```

```
a = [0]
b = [a, 1]
print(a == b[0])
# Output: True
```

Question:

Write a program to find the area of a triangle given all three sides.

```
import math
a=int(input("Enter first side: "))
b=int(input("Enter second side: "))
c=int(input("Enter third side: "))
s=(a+b+c)/2
area=math.sqrt(s*(s-a)*(s-b)*(s-c))
print("Area of the triangle is: ",round(area,2))
```

Theoretical probability:

$$P(\text{event}) = \frac{\text{Number of favorable outcomes}}{\text{Total Number of possible outcomes}}$$

For example:

When a coin is tossed:

- $P(\text{head}) = \frac{1}{2}$
- $P(\text{tail}) = \frac{1}{2}$

Experimental probability:

$$P(\text{event}) = \frac{\text{Number of times event occurs}}{\text{Total number of trails}}$$

For example:

When a coin is tossed 7 times: A head is recorded 5 times and a tail 2 times

- $P(\text{head}) = \frac{5}{7}$
- $P(\text{tail}) = \frac{2}{7}$

```
# prints a list of names of all the files present in the current directory

import os
print(os.listdir(os.curdir))
```

Question 99

```
print(r"Alan" == "Alan")
# Output: True
print(r"\Turing" == "\\Turing")
# Output: True
print(r"\Turing" == "\Turing")
# Output: True
```

Question:

Write a program to find the gravitational force acting between two masses.

Solution:

```
M=float(input("Enter the first mass: "))
m=float(input("Enter the second mass: "))
r=float(input("Enter the distance between the centers of the masses: "))
G=6.673*(10**-11)
F=(G*M*m)/(r**2)
print("Hence, the gravitational force is: ",round(F,2),"N")
```

Question 100

A foreground service → Performs some operation that the user is aware of

A background service → Performs some operation that the user is unaware of

Question:

Write a program to replace a, t with 2, 6 respectively in a given string.

```
x = 'alanalbertjohn'
y = x.maketrans('ab', '26')
print(x)
# Output: alanalbertjohn
print(x.translate(y))
# Output: 2l2n2l6ertjohn
```

```
import re
print(re.compile(r"\\alan"))
# Output: re.compile('\\\\alan')
```

```
def myfunc(x= "Albert" ):
      print (x)

myfunc()

# Output: Albert
```

```
x= 0
if x % 2 == 0 :
  if x == 0 :
     print ("x is even but zero" )
  else :
     print ("x is even" )
else :
  print ("x is odd" )

# Output: x is even but zero
```

Question 101

Question:

Suppose we have a text with three numbers in it. Write a program to filter these out and print them separately.

Solution:

```
import re
x = '''
Prince Albert died on 14 December 1861
at the age of 42
'''
print (re.findall( r'\d{1,4}' , x))
```

Output:

```
['14', '1861', '42']
```

Cohort analysis: A kind of behavioral analytics where users are grouped based on shared characteristics to help us better monitor and analyze their activity

```
import re
x = '''
Prince Albert died on 14 December 1861
at the age of 42
'''
print (re.sub( r'\d{1,4}', "76" , x))
```

```
# Output:

Prince Albert died on 76 December
76
at the age of 76
```

Question 102

Yield management

A marketing approach that focuses on reaching the right customer at the right moment and at the right price

Question:

Write a program to create a 2×4×3 matrix which is filled with eights

---

Solution:

```
import numpy as np
print (np.full(( 2 , 4 , 3 ), 8 ))
```

```
# Output:

[[[8 8 8]
  [8 8 8]
  [8 8 8]
  [8 8 8]]

 [[8 8 8]
  [8 8 8]
  [8 8 8]
  [8 8 8]]]
```

```
import numpy as np
print (np.linspace( 0 , 50 , 5))
# Output: [ 0.  12.5 25.  37.5 50. ]
print(np.arange( 1 , 10 , 3 ))
# Output: [1 4 7]
```

First In, First Out: A data structure processing method in which the first element is handled first and the newest element is handled last

```
import numpy as np

a = 2
print(np.exp(a))
# Output: 7.38905609893065
print(np.log(a))
# Output: 0.6931471805599453
```

Question 103

Question:

Write a program to print:

```
* * * *
* * * *
* * * *
* * * *
```

Solution:

```
x= 4
for i in range(1,x+1):
  print("* "*x)
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(j, end=" ")
  print()

# Output:
            1 2 3 4
            1 2 3 4
            1 2 3 4
            1 2 3 4
```

Question:

Write a program to print:

```
1 1 1 1
2 2 2 2
3 3 3 3
4 4 4 4
```

Solution:

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(i, end=" ")
  print()
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(chr(64+i),end=" ")
  print()

# Output:
            A A A A
            B B B B
            C C C C
            D D D D
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(chr(64+j),end=" ")
  print()

# Output:
            A B C D
            A B C D
            A B C D
            A B C D
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(x+1-i, end=" ")
  print()

# Output:
            4 4 4 4
            3 3 3 3
            2 2 2 2
            1 1 1 1
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(x+1-j, end=" ")
  print()

# Output:
            4 3 2 1
            4 3 2 1
            4 3 2 1
            4 3 2 1
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(chr(65+x-i),end=" ")
  print()

# Output:
            D D D D
            C C C C
            B B B B
            A A A A
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+1):
      print(chr(65+x-j),end=" ")
  print()
```

```
# Output:

      D C B A
      D C B A
      D C B A
      D C B A
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+2-i):
      print("*",end=" ")
  print()

# Output:
            * * * *
            * * *
            * *
            *
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+2-i):
      print(i, end=" ")
  print()

# Output:
            1 1 1 1
            2 2 2
            3 3
            4
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+2-i):
      print(j, end=" ")
  print()

# Output:
            1 2 3 4
            1 2 3
            1 2
            1
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+2-i):
      print(chr(64+i),end=" ")
  print()

# Output:
            A A A A
            B B B
            C C
            D
```

```
x= 4
for i in range(1,x+1):
  for j in range(1,x+2-i):
      print(chr(64+j),end=" ")
  print()
```

```
# Output:

      A B C D
      A B C
      A B
      A
```

```
x= 4
for i in range(1,x+1):
   print(" "*(x-i),"*"*i, end=" ")
   print()

# Output:
               *
              **
             ***
            ****
```

```
x= 4
for i in range(1,x+1):
  print(" "*(x-i),end="")
  for j in range(1,i+1):
     print("*",end=" ")
  print()

# Output:
               *
              * *
             * * *
            * * * *
```

```
x= 4
for i in range(1,x+1):
   print(" "*(x-i),(str(i)+" ")*i)
   print()

# Output:
               1

              2 2

             3 3 3

            4 4 4 4
```

```
x= 4
for i in range(1,x+1):
    print(" "*(x-i),end="")
    for j in range(1,2*i):
        print(j, end=" ")
    print()

# Output:
               1
             1 2 3
           1 2 3 4 5
         1 2 3 4 5 6 7
```

```
x= 4
for i in range(1,x+1):
    print(" "*(x-i),end="")
    for j in range(2*i-1,0,-1):
        print(j, end=" ")
    print()
```

```
# Output:
            1
          3 2 1
        5 4 3 2 1
      7 6 5 4 3 2 1
```

Question 104

```
print(str(6) == '6')
# Output: True
```

Question:

Write a program to find the sum of first N Natural Numbers.

Solution:

```
x=int(input("Enter a number: "))
sum = 0
while(x > 0):
    sum = sum + x
    x=x-1
print("The sum of first N natural numbers is: ",sum)
```

Question 105

```
print([i.lower() for i in ['ALAN', 'TURING']])
# Output: ['alan', 'turing']
```

Question:

Write a program to find the sum of series: 1 + 1/2 + 1/3 + ... + 1/N.

Solution:

```
x=int(input("Enter the number of terms: "))
sum=0
for i in range(1,x+1):
    sum = sum +(1/i)
print("The sum of series is: ", round(sum,2))
```

Question 106

```
print([(a, b) for a in range(2) for b in range(2)])
# Output:
[(0, 0), (0, 1), (1, 0), (1, 1)]
```

Question:

Write a program to find whether a number is a power of two.

Solution:

```
def myfunc(x):

# returns True if x is a power of two.

   if x <= 0:
        return False
   else:
        return x & (x - 1) == 0

x = int(input('Enter a number: '))

if myfunc(x):

   print('{} is a power of two.'.format(x))
else:
   print('{} is not a power of two.'.format(x))
```

Question 107

Question:

Write a program to replace all occurrences of 'a' with '$' in a string.

Solution:

```
x=input("Enter a string: ")
x=x.replace('a','$')
x=x.replace('A','$')
print(x)
```

```
# Output:
Enter a string: Alan # entered string
$l$n
```

Question 108

Question:

2 strings are said to be **anagrams** if they have the same characters but are arranged differently. **For example:** "keep", "peek"

Write a program to detect if two strings are anagrams

Solution:

```
x=input("Enter the first string: \n")
y=input("Enter the second string: \n")
if(sorted(x)==sorted(y)):
   print("The 2 strings are anagrams.")
else:
   print("The 2 strings aren't anagrams.")
```

```
# Output:

Enter the first string:
keep # entered string

Enter the second string:
peek # entered string

The 2 strings are anagrams.
```

```
salary = [("Mary", 24000),
 ("Ram", 12000),
 ("Alan", 10000),
 ("Sony", 84000)]

# prints the names of the employees whose salaries are above 20000
i = [a for a, b in salary if b>20000]
print(i)

# Output: ['Mary', 'Sony']
```

Question 109

Question:

Write a program to modify a string where the first character and the last character have been exchanged.

Solution:

```
def myfunc(x):
    return x[-1:] + x[1:-1] + x[:1]
x=input("Enter a string: ")
print("Modified string: ")
print(myfunc(x))
```

```
# Output:
Enter a string: james # entered string
Modified string:
samej
```

Question 110

Question:

Write a program to count the number of vowels in a string.

Solution:

```
x=input("Enter a string:")
v=0
for i in x:
    if(i=='a' or i=='e' or i=='i' or i=='o' or i=='u' or i=='A' or i=='E' or i=='I' or i=='O' or i=='U'):
        v=v+1
print("Number of vowels: ")
print(v)


# Output:
Enter a string: albert # entered string
Number of vowels:
2
```

Question 111

Question:

Write a program to take a string and replace every blank space with a Plus symbol.

Solution:

```
x=input("Enter a string: ")
x=x.replace(' ','+')
print("Modified string: ")
print(x)
```

```
# Output:

Enter a string: alan turing # entered string
Modified string:
alan+turing
```

```
print([line.strip() for line in open("1.txt")])
# Output: ['John', 'Mary', 'James', 'Albert']
```

```
# contents of 1.txt
John
Mary
James
Albert
```

Question 112

Question:

Write a program to calculate the length of a string without using library function.

Solution:

```
x=input("Enter a string: ")
b=0
for i in x:
    b=b+1
print("Length of the string: ")
print(b)
```

```
# Output:
Enter a string: Albert # entered string
Length of the string:
6
```

```
x = 'Alan Mathison Turing'

print(x[6:3:-1]) # Output: aM

print(x[4:8:2]) # Output: a
```

```
x = ['Albert Einstein was a theoretical physicist.',
 'Albert Einstein grew up in Munich.',
 'One of the greatest physicists of all time.']

a = map(lambda s: (True, s) if 'Albert' in s else (False, s), x)

print(list(a))

# Output:

[(True, 'Albert Einstein was a theoretical physicist.'), (True, 'Albert Einstein
grew up in Munich.'), (False, 'One of the greatest physicists of all time.')]
```

```
x = ['Albert Einstein was a theoretical physicist.',
 'Albert Einstein grew up in Munich.',
 'One of the greatest physicists of all time.']

a = map(lambda s: (True, s) if 'time' in s else (False, s), x)

print(list(a))

# Output:

[(False, 'Albert Einstein was a theoretical physicist.'), (False, 'Albert
Einstein grew up in Munich.'), (True, 'One of the greatest physicists of all
time.')]
```

Question 113

Question:

Write a program to calculate the number of words and characters present in a string.

Solution:

```
x=input("Enter a string: ")
character=0
word=1
for i in x:
   character=character+1
   if(i==' '):
      word=word+1
print("Number of words in the string:")
print(word)
print("Number of characters in the string:")
print(character)
```

```
# Output:
Enter a string: Alan Turing # entered string
Number of words in the string:
2
Number of characters in the string:
11
```

```
y = [[2.9, 2.8, 2.8, 2.4, 2.5, 2.7],
 [3.5, 3.4, 3.4, 3.3, 3.2, 3.1],
 [4.4, 4.9, 4.9, 4.1, 4.0, 4.0]]
x = [i[::2] for i in y]
print(x)

# Output:
[[2.9, 2.8, 2.5], [3.5, 3.4, 3.2], [4.4, 4.9, 4.0]]
```

```
x = '''
Python is a high-level, interpreted,
general-purpose programming language.
Its design philosophy emphasizes code
readability with the use of significant indentation
'''

find = lambda i, b: i[i.find(b)-10:i.find(b)+10] if b in i else -1

print(find(x, 'the'))

# Output: lity with the use of


x = '''
Python is a high-level, interpreted,
general-purpose programming language.
Its design philosophy emphasizes code
readability with the use of significant indentation
'''

find = lambda i, b: i[i.find(b)-17:i.find(b)+17] if b in i else -1

print(find(x, 'the'))

# Output: readability with the use of signif
```

```
names = ['Alan', 'Albert', 'John', 'Mary',
 'Joseph', 'David', 'Mendel', 'Watson']
names[1::2] = names[::2]
print(names)

# Output: ['Alan', 'Alan', 'John', 'John', 'Joseph', 'Joseph', 'Mendel', 'Mendel']
```

Question 114

Question:

Write a program to count number of lowercase characters in a string.

Solution:

```
x=input("Enter a string:")
c=0
for i in x:
  if(i.islower()):
    c=c+1
print("The number of lowercase characters: ")
print(c)
```

```
Enter a string: Alan
The number of lowercase characters:
3
```

Question 115

Question:

Write a program to count the number of lowercase letters and uppercase letters in a string.

Solution:

```
x=input("Enter a string: ")
c=0
d=0
for i in x:
      if(i.islower()):
        c=c+1
      elif(i.isupper()):
         d=d+1
print("The number of lowercase characters: ")
print(c)
print("The number of uppercase characters: ")
print(d)
```

Question 116

```
import numpy as np
x = np.array([31, 32, 33, 34], dtype=np.int16)
print(x) # Output: [31 32 33 34]
print(x.dtype) # Output: int16
```

Question:

Write a program to check if a substring is present in a given string.

Solution:

```
x=input("Enter a string: ")
y=input("Enter the substring: ")
if(x.find(y)==-1):
   print("A substring is not present in a string.")
else:
   print("A substring is present in a string.")
```

```
import numpy as np

x = np.array([11, 12, 13, 14])
print(x.ndim)
# Output: 1 (one-dimensional array)

y = np.array([[12, 11, 12], [13, 12, 13], [14, 13, 14]])
print(y.ndim)
# Output: 2 (two-dimensional array)

z = np.array([[[11, 12, 13], [12, 13, 14], [13, 14, 15]],
 [[11, 12, 14], [12, 13, 15], [13, 14, 16]]])
print(z.ndim)
# Output: 3 (three-dimensional array)
```

```
import numpy as np
x = np.array([2] * 4)
print(x)
# Output: [2 2 2 2]
x[1::] = [21] * 3
print(x)
# Output: [ 2 21 21 21]
x[1:3:2] = 8
print(x)
# Output: [ 2  8 21 21]
```

```
import numpy as np

print(np.abs(np.array([21, -21, 22, -52])))

# Output: [21 21 22 52]
```

```
print([(x,y) for x in range(4) for y in range(x+1,3)])

# Output: [(0, 1), (0, 2), (1, 2)]
```

Question 117

Question:

Write a program to concatenate two dictionaries into one dictionary.

```
x={'A':1,'B':2}
y={'C':3}
x.update(y)
print(x)
# Output: {'A': 1, 'B': 2, 'C': 3}
```

Question 118

Question:

Write a program to check if a given key exists in a dictionary or not.

```
x={'A':1,'B':2,'C':3}
key=input("Enter a key to check: ")
if key in x.keys():
   print("Key is present and value of the key is:")
   print(x[key])
else:
   print("Key isn't present!")
```

```
# Output:
Enter a key to check: A
Key is present and value of the key is:
1
```

Question 119

Question:

Write a program to find the sum of values in the dictionary

```
x={'A':100,'B':540,'C':239}
print("The sum of values in the dictionary: ")
print(sum(x.values()))
# Output:
The sum of values in the dictionary:
879
```

Question 120

```
x = 3
fact = lambda x: x * fact(x-1) if x > 1 else 1
print(fact(x))
# Output: 6 (!3=3×2×1=6)
```

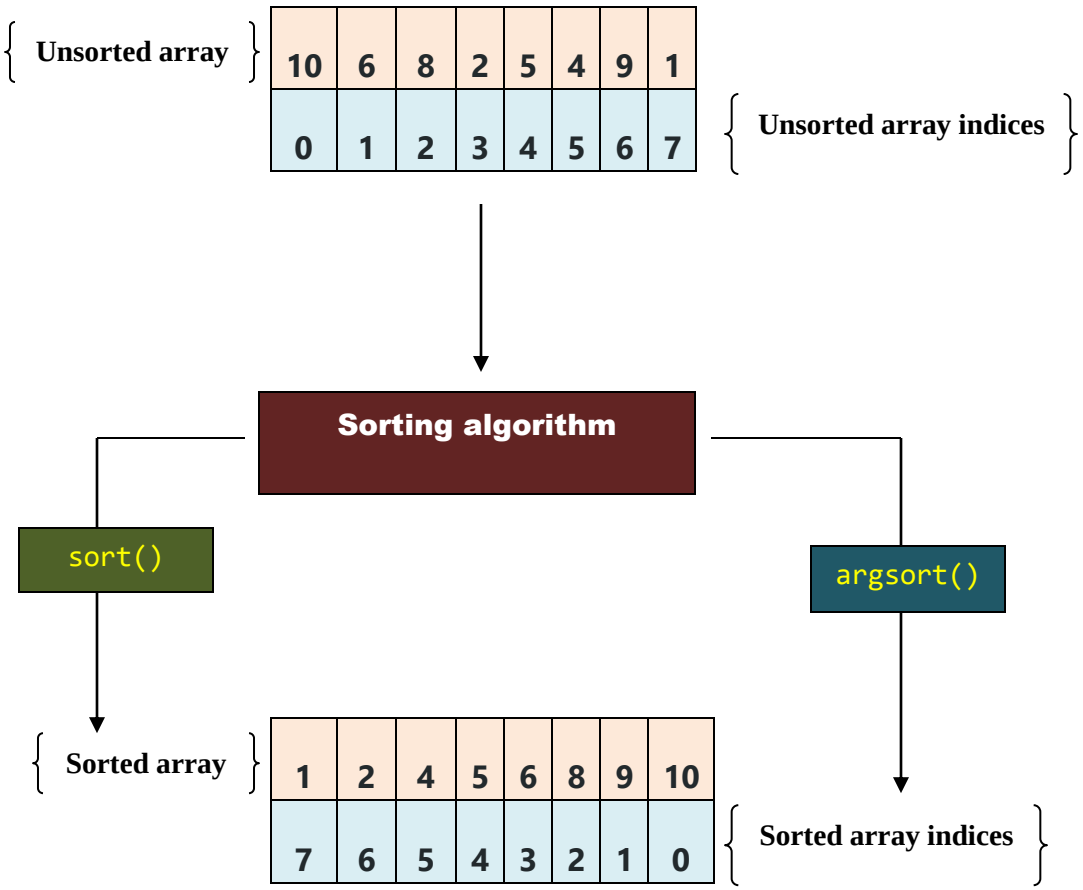
Question:

Write a program to find the product of values in the dictionary

---

```
x={'A':10,'B':10,'C':10}
a=1
for i in x:
   a=a*x[i]
print("The product of values in the dictionary: ")
print(a)
```

```
# Output:
The product of values in the dictionary:
1000
```




```
import numpy as np
x = np.array([10, 6, 8, 2, 5, 4, 9, 1])

print(np.sort(x))
# Output: [ 1  2  4  5  6  8  9 10]

print(np.argsort(x))
# Output: [7 3 5 4 1 2 6 0]
```

```
import re
x = 'Albert Einstein profoundly changed physics and ideas about space and time'

print(re.findall('p.*?c.*?s', x))

# Output: ['profoundly changed phys']
```

```
import re
x = 'Albert Einstein profoundly changed physics and ideas about space and time'
print(re.findall('a.d', x))
# Output: ['and', 'and']

import re
x = 'Albert Einstein profoundly changed physics and ideas about space and time'
print(re.findall('A.*n', x))
# Output:
['Albert Einstein profoundly changed physics and ideas about space an']
```

```
import re
print(re.findall('a{2,4}b', 'ab'))
# Output: []
print(re.findall('a{2,4}b', 'aaab'))
# Output: ['aaab']
print(re.findall('a{2,4}b', 'aaaaab'))
# Output: ['aaaab']
print(re.findall('a{2,4}b', 'aaaaaab'))
# Output: ['aaaab']
```

```
from sympy import sqrt
print(sqrt(16))
# Output: 4
```

```
import re
x = 'Albert profoundly changed physics and ideas about space and time'
print(re.sub("Albert(?!')", 'Einstein', x))
# Output:
Einstein profoundly changed physics and ideas about space and time
```

```
is_anagram = lambda a, b: sorted(a) == sorted(b)

print(is_anagram("peek", "keep"))

# Output: True
```

```
# check whether a given number x is prime or not

def myfunc(x):
 for i in range(2,x):
   if x % i == 0:
         return False
 return True


print(myfunc(13))

# Output: True

print(myfunc(178))

# Output: False
```

```
import sympy


print(sympy.isprime(13))
# Output: True


print(sympy.isprime(178))
# Output: False
```

```
# prints all prime numbers <= a
a = 10
print([x for x in range(2,a+1) if myfunc(x)])

 # Output: [2, 3, 5, 7]
```

```
from sympy import Float

print(Float(7.54))

# Output: 7.54000000000000

print(Float('1.44E5'))

# Output: 144000.000000000

print(Float(1.55555,2))

# Output: 1.6
```

```
from sympy import Integer

print(Integer(5.4))

# Output: 5

print(Integer(3/8))

# Output: 0
```

```
import sympy
# prints the SymPy version
print(sympy.__version__)

from sympy import FiniteSet
x=[1,3,5,2,8]
y=[12,10,8,7,1]
print(FiniteSet(*x))
# Output: {1, 2, 3, 5, 8}
print(FiniteSet(*y))
# Output: {1, 7, 8, 10, 12}
```

```
x = set('abc')
print(x)
# Output: {'b', 'a', 'c'}
x = {'a', 'b', 'c'}
print(x)
# Output: {'b', 'a', 'c'}
```

```
from sympy import FiniteSet, Complement
x=[1,3,5,2,8]
y=[12,10,8,7,1]
a=FiniteSet(*x)
print(a)
# Output: {1, 2, 3, 5, 8}
b=FiniteSet(*y)
# Output: {1, 7, 8, 10, 12}
print(b)
print(Complement(a,b), Complement(b,a))
# Output: {2, 3, 5} {7, 10, 12}
```

```
from sympy import FiniteSet
x="alan"
y="jonny"
print(FiniteSet(*x))
# Output: {a, l, n}
print(FiniteSet(*y))
# Output: {j, n, o, y}
```

```
from sympy import FiniteSet, SymmetricDifference
x=[13,11,15,17]
y=[19,17,12,11]
a=FiniteSet(*x) # {11, 13, 15, 17}
b=FiniteSet(*y) # {11, 12, 17, 19}
print(SymmetricDifference(a,b))
# Output: {12, 13, 15, 19}
```

```
for i in ['Alan', 'Albert', 'James', 'John']:
    print('Mary ' + i)

# Output:
            Mary Alan
            Mary Albert
            Mary James
            Mary John
```

```
x = 'Alan'
y = 'Albert'
z = 'John'
print(x, y, z)
# Output: Alan Albert John
```

```
x = [[11, 12, 13], [14, 15, 16], [17, 18, 19]]
print([x[s][s] for s in [0, 1, 2]])
# Output: [11, 15, 19]
```

```
# Repeat characters in a string
print([i * 3 for i in 'mary'])
# Output: ['mmm', 'aaa', 'rrr', 'yyy']
```

```
x={'a': 1, 'c': 3, 'b': 2}
x['e'] = 4
print(x)
# Output: {'a': 1, 'c': 3, 'b': 2, 'e': 4}
x['a'] = 4
print(x)
# Output: {'a': 4, 'c': 3, 'b': 2, 'e': 4}

if not 'd' in x:
 print('Missing!')
# Output: Missing!

print(x.get('a')) # Output: 4


print(x['d'] if 'd' in x else 'Missing!')
# Output: Missing!
```

```
print(11 > 12, 11 < 12)
# Output: False True
```

```
x = 1.0 / 3.0
print('%e' % x)
# Output: 3.333333e-01
print('%4.2f' % x)
# Output: 0.33
print('{0:4.2f}'.format(x))
# Output: 0.33
```

```
import decimal
decimal.getcontext().prec = 3
print(decimal.Decimal('2.00') / decimal.Decimal('6.00'))
# Output: 0.333

decimal.getcontext().prec = 2
print(decimal.Decimal('2.00') / decimal.Decimal('6.00'))
# Output: 0.33
```

```
print('%.1f' % 9.496, '{0:.2f}'.format(9.496))
# Output: 9.5 9.50

print((2.5).as_integer_ratio())
# Output: (5, 2)
# 5/2 = 2.5

x = {'A', 'B', 'C', 'D'}
print({'A', 'C'} < x)
# Output: True

print('apple\'s', "apple\"s")
# Output: apple's apple"s

print(str('alan'), repr('alan'), int("16"))
# Output: alan 'alan' 16

x = 'David'
print(x[:5] + 'Hilbert' + x[-1])
# Output: DavidHilbertd
```

```
text = """
Albert %(name)s was born on
March 14, %(year)s
"""
x = {'name': 'Einstein', 'year': 1879}
print(text % x)

# Output:
Albert Einstein was born on
March 14, 1879
```

```
i = 5.987654321
print('%e | %f | %g' % (i, i, i))
# Output: 5.987654e+00 | 5.987654 | 5.98765
```

Question 121

Question:

Write a program to count the frequency of words appearing in a string using a dictionary.

```
x=input("Enter a string: ")
i=[]
i=x.split()
print(dict(zip(i, [i.count(s) for s in i])))

# Output:
Enter a string: alan alan is # entered string
{'alan': 2, 'is': 1}
```

Question 122

```
for ((x, y), z) in [([21, 22], 23), ['AB', 26]]: print(x, y, z)

# Output:
            21 22 23
            A B 26
```

Question:

Write a program to check common letters in the two input strings.

```
x=input("Enter the first string: ")
y=input("Enter the second string: ")
a=list(set(x)&set(y))
print("The common letters are: ")
for i in a:
    print(i)
```

```
# Output:
Enter the first string: alan' # entered string
Enter the second string: albert # entered string
The common letters are:
a
l
```

Question 123

Question:

Write a program to display which letters are in the first string but not in the second string.

```
x=input("Enter the first string: ")
y=input("Enter the second string: ")
a=list(set(x)-set(y))
print("The letters in the first string but not in the second string are: ")
for i in a:
    print(i)
```

```
print(list(range(-3, 3)))
# Output: [-3, -2, -1, 0, 1, 2]
print(str([41, 42]) + "44")
# Output: [41, 42]44
print([41, 42] + list("44"))
# Output: [41, 42, '4', '4']
```

Question 124

Question:

Write a program to display which letters is present in both the strings

---

```
x=input("Enter the first string: ")
y=input("Enter the second string: ")
a=list(set(x)|set(y))
print("The letters present in both the strings are: ")
for i in a:
    print(i)
```

```
M = {}
M[(12, 13, 14)] = 78
M[(17, 18, 19)] = 64
X = 12; Y = 13; Z = 14
print(M[(X, Y, Z)])
# Output: 78
print(M)
# Output: {(12, 13, 14): 78, (17, 18, 19): 64}

if (12,13,16) in M:
 print(M[(12,13,16)])
else:
 print('Missing!')

# Output: Missing!
```

```
x = {'name': 'John',
     'courses': ['python', 'java'],
      'email': 'john156@yahoo.in',
      'address': {'state': 'Canada', 'zip':54600}}
print(x['name']) # Output: John
print(x['courses']) # Output: ['python', 'java']
print(x['courses'][1]) # Output: java
```

```
print(dict.fromkeys(['x', 'y'], 1))
# Output: {'x': 1, 'y': 1}

print({i: i * 3 for i in 'ALAN'})
# Output: {'A': 'AAA', 'L': 'LLL', 'N': 'NNN'}

print({i.lower(): i + '!' for i in ['ALAN', 'JOHN', 'MARY']})
# Output: {'alan': 'ALAN!', 'john': 'JOHN!', 'mary': 'MARY!'}

print({i:11 for i in ['x', 'y', 'z']})
# Output: {'x': 11, 'y': 11, 'z': 11}

print(dict(x=21, y=22, z=23))
# Output: {'x': 21, 'y': 22, 'z': 23}
```

```
x = [11, ('i', 13)]
y = [11, ('i', 12)]
print(x < y, x == y, x > y)
# Output: False False True

x = {'p':11, 'q':12}
y = {'p':11, 'q':13}
print(x == y)
# Output: False

x= [11, 12, 13]
y = ['A', x, 'D']
print(y)
# Output: ['A', [11, 12, 13], 'D']
```

```
x = '1988'
y= 'max'
print(x.isdigit(), y.isdigit())
# Output: True False
```

```
while True:
 x = input('Enter a string: ')
 if x == 'END': break
print(x.lower())

# Output:
Enter a string: Alan # entered string
Enter a string: END  # entered string
end
```

```
x = 'ALAN'
a, b, c, d = x
print(a, d) # Output: A N

a, b, c = x[0], x[1], x[2:]
print(a, b, c) # Output: A L AN

a, b, c = list(x[:2]) + [x[2:]]
print(a, b, c) # Output: A L AN
```

```
while True:
      x = input('Enter a number: ')
      if x == 'end': break
      print(int(x) ** 2)
print('Have a Good Day!')

# Output:
Enter a number: 5 # entered number
25
Enter a number: end # entered string
Have a Good Day!
```

```
x = 'ALAN'
a, b = x[:2]
c = x[2:]
print(a,b,c)
# Output: A L AN
(a, b), c = x[:2], x[2:]
print((a, b), c)
# Output: ('A', 'L') AN

((a, b), c) = ('AL', 'AN')
print(((a, b), c))
# Output: (('A', 'L'), 'AN')
```

```
while True:
 x = input('Enter a number: ')
 if x == 'end': break
 elif not x.isdigit():
   print('Wrong Entry ' * 2)
 else:
   print(int(x) ** 2)
print('Have a Good Day!')
```

```
# Output:

Enter a number: 5
25
Enter a number: max
Wrong Entry Wrong Entry
Enter a number: end
Have a Good Day!
```

```
while True:
 x = input('Enter a number: ')
 if x == 'end': break
 try:
   i = int(x)
 except:
   print('Wrong Entry ' * 2)
 else:
   print(int(x) ** 2)
print('Have a Good Day!')
```

```
# Output:

Enter a number: 5
25
Enter a number: max
Wrong Entry Wrong Entry
Enter a number: end
Have a Good Day!
```

```
while True:
 x = input('Enter a number: ')
 if x == 'end': break
 elif not x.isdigit():
      print('Wrong Entry ' * 2)
 else:
       i = int(x)
       if i < 25:
           print('Enter a number > 25')
       else:
           print(i ** 2)
print('Have a Good Day!')
```

```
# Output:

Enter a number: 5
Enter a number > 25

Enter a number: 36
1296

Enter a number: max
Wrong Entry Wrong Entry

Enter a number: end
Have a Good Day!
```

Question 125

Question:

Write a program to implement birthday dictionary.

```
if __name__ == '__main__':

    birthdays = {
        'Albert Einstein': '03/14/1879',
        'Benjamin Franklin': '01/17/1706',
        'Ada Lovelace': '12/10/1815',
        'Donald Trump': '06/14/1946',
        'Rowan Atkinson': '01/6/1955'}

    print('Welcome to the birthday dictionary. We know the birthdays of:')
    for name in birthdays:
        print(name)

    print('Who\'s birthday do you want to look up?')
    name = input()
     if name in birthdays:
        print('{}\'s birthday is {}.'.format(name, birthdays[name]))
    else:
        print('Sorry, we don\'t have {}\'s birthday.'.format(name))
```

```
# Output:
            11 [12, 13, 14]
            12 [13, 14]
            13 [14]
            14 []
# Solution:
            x = [11, 12, 13, 14]
            while x:
                i, x = x[0], x[1:]
                print(i, x)
```

Or

```
x = [11, 12, 13, 14]
while x:
   i, *x = x
   print(i, x)
```

```
x = [11, 12, 13, 14]
x += [15, 16]
print(x)
# Output: [11, 12, 13, 14, 15, 16]

a = 'Einsteinian'
b = 1905
c = ['Papers']
print(a, b, c)
# Output: Einsteinian 1905 ['Papers']

print('%s: %-.4f, %05d' % ('x', 6.98768, 12))
# Output: x: 6.9877, 00012
```

```
x = 'alan'
if 'al' in 'albert':
 print(x * 2)
 x += 'einstein'
 if x.endswith('einstein'):
    x *= 2
    print(x)

# Output:
      alanalan
      alaneinsteinalaneinstein
```

```
import sys

sys.stderr.write(('Alan ' * 2) + '\n')

# Output: Alan Alan

print('Albert ' * 2, file=sys.stderr)

# Output: Albert Albert
```

```
x=0; y=5
while x < y:
  print(x, end=' ')
  x += 1

# Output: 0 1 2 3 4
```

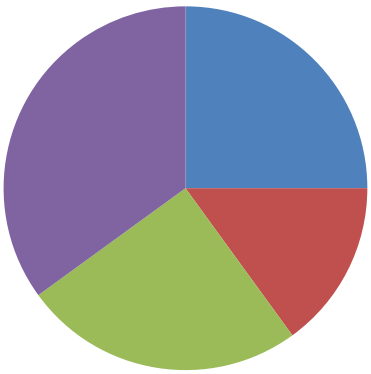
```
i = 'apple'
while i:
 print(i, end=' ')
 i = i[1:]

# Output: apple pple ple le e
```

```
while True:
   x = input('Enter a string: ')
   if x == 'einstein': break
   y = input('Enter a number: ')
   print(x, '=>', int(y) ** 2)

# Output:
Enter a string: alan # entered string
Enter a number: 5 # entered number
alan => 25
Enter a string: einstein # entered string
```

```
print(list(range(0, len('albert'), 2)))

# Output: [0, 2, 4]

for x in range(0, len('albert'), 2): print('albert'[x], end=' ')

# Output: a b r
```

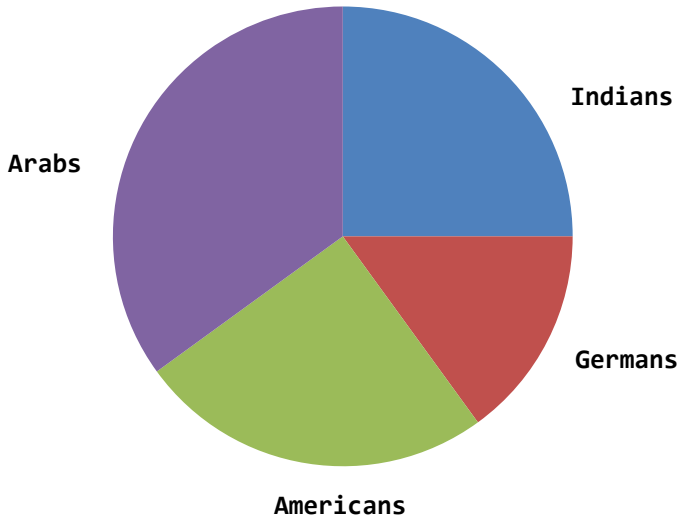
```
import matplotlib.pyplot as plt

import numpy as np

plt.pie(np.array([25, 15, 25, 35]))

plt.show()
```

```
# Output:
```



```
import matplotlib.pyplot as plt

import numpy as np

plt.pie(np.array([25, 15, 25, 35]), labels = ["Indians", "Germans", "Americans", "Arabs"])

plt.show()
```

```
# Output:
```




```
x = [1, 2, 3, 4]
for s in range(len(x)):
    x[s] += 10
    print(x)
```

```
[11, 2, 3, 4]
[11, 12, 3, 4]
[11, 12, 13, 4]
[11, 12, 13, 14]
```

```
x = [1,2,3,4]
print([i + 10 for i in x])

# Output: [11, 12, 13, 14]

a = [15,16,17,18]
b = [25,26,27,28]
for (x, y) in zip(a, b):
   print(x, y, '...', x+y)

# Output:
            15 25 ... 40
            16 26 ... 42
            17 27 ... 44
            18 28 ... 46
```

```
x, y, z = (1,2,3), (4,5,6), (7,8,9)
print(z)
# Output: (7, 8, 9)
print(list(zip(x, y, z)))
# Output: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]
```

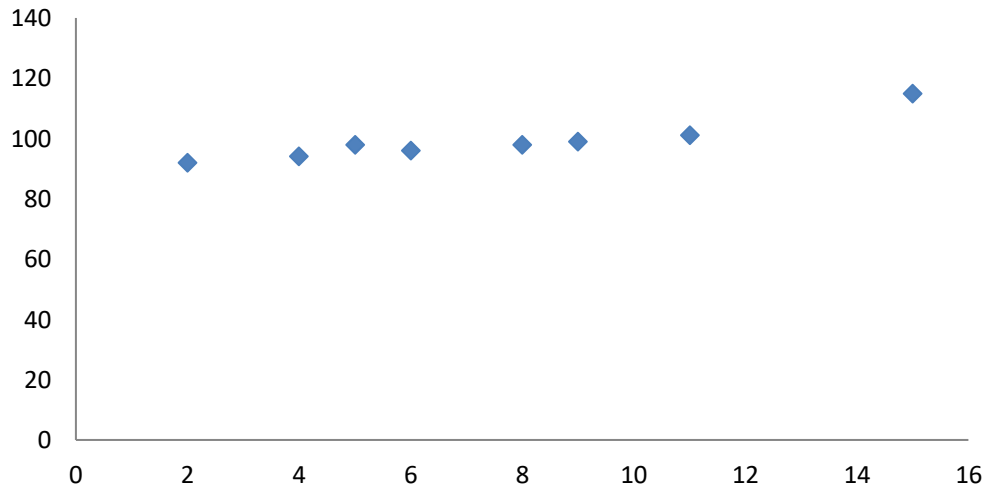
```
John
Mary
James
Albert
```

Contents of 1.txt

```
print([line.upper() for line in open('1.txt')])
# Output: ['JOHN\n', 'MARY\n', 'JAMES\n', 'ALBERT']

print([line.rstrip().upper() for line in open('1.txt')])
# Output: ['JOHN', 'MARY', 'JAMES', 'ALBERT']

print([line.split() for line in open('1.txt')])
# Output: [['John'], ['Mary'], ['James'], ['Albert']]

print([line.replace('n', '!') for line in open('1.txt')])
# Output: ['Joh!\n', 'Mary\n', 'James\n', 'Albert']

print([('Alb' in line, line[0]) for line in open('1.txt')])
# Output: [(False, 'J'), (False, 'M'), (False, 'J'), (True, 'A')]
```

```
import matplotlib.pyplot as plt
import numpy as np

a = np.array([2,4,5,9,11,6,8,15])
b = np.array([92,94,95,99,101,96,98,115])
plt.scatter(a, b)
plt.show()
```

```
# Output:
```




```
import pandas as pd

x= pd.DataFrame({
   'names': ['Alan', 'John', 'James', 'Albert'],
   'ID':['ab15','ab17','ab19','ab20']},
   index=[1,2,3,4])

y = pd.DataFrame({
   'names': ['Mary', 'Jessie', 'David', 'Sony'],
   'ID':['mx40','mx44','mx41','mx49']},
   index=[5,6,7,8])

print(pd.concat([x,y]))
```

```
      names  ID
1    Alan  ab15
2    John  ab17
3   James  ab19
4   Albert ab20
5    Mary  mx40
6   Jessie mx44
7   David  mx41
8    Sony  mx49
```

Question 126

```
print(list(filter((lambda i: i < 0), range(-4, 4))))
# Output: [-4, -3, -2, -1]
print(list(filter((lambda i: i > 0), range(-4, 4))))
# Output:  [1, 2, 3]
```

Question:

Write a program to remove punctuations from a string.

```
# define punctuation
i = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''

# take a input from the user
x = input("Enter a string: ")

# remove punctuation from the string
y = ""

for s in x:
   if s not in i:
       y = y + s

# display the unpunctuated string
print(y)
```

```
# Output:
Enter a string: Alan: Turing! # entered string
Alan Turing
```

```
y, z = 11, 2
def myfunc():
 global x
 x = y * z
 print(x)

myfunc()
# Output: 22
```

```
print(list(range(-4, 4)))
# Output: [-4, -3, -2, -1, 0, 1, 2, 3]

ab = []
for i in range(-4, 4):
    if i > 0:
       ab.append(i)
print(ab)

# Output: [1, 2, 3]
```

Question 127

Question:

Write a program to get IP address of your computer.

```
import socket
x = socket.gethostname()
y = socket.gethostbyname(x)
print("My Computer Name is: " + x)
print("My Computer IP Address is: " + y)

# Output:
My Computer Name is: DESKTOP-68GI94H
My Computer IP Address is: 192.168.6.1
```

```
import random
minimum = 1
maximum = 6

x = "yes"

while x == "yes" or x == "y":
    print ("Rolling the dice...")
    print ("The value is....")
    print (random.randint(minimum, maximum))

    x = input("Roll the dice again?")
```

Python program to illustrate Dice Roll Simulator

```
import re

x = 'Saint John was one of the 12 Apostles of Jesus.'

y = b'Saint John was one of the 12 Apostles of Jesus.'

print(re.match('(.*) was (.*) of (.*)', x).groups())

# Output: ('Saint John', 'one of the 12 Apostles', 'Jesus.')

print(re.match(b'(.*) was (.*) of (.*)', y).groups())

# Output: (b'Saint John', b'one of the 12 Apostles', b'Jesus.')
```

```
print('{0:,d}'.format(5000000))

# Output: 5,000,000

print('{0:12,.2f}'.format(5000000))

# Output: 5,000,000.00

print('{0:%} {1:,.2%}'.format(2.52, 5496))

# Output: 252.000000% 549,600.00%

print('{0}/{1}/{2}'.format('Users', 'Manju', '1.txt'))

# Output: Users/Manju/1.txt

print('{}/{}/{}'.format('Users', 'Manju', '1.txt'))

# Output: Users/Manju/1.txt

print('%s/%s/%s' % ('Users', 'Manju', '1.txt'))

# Output: Users/Manju/1.txt

print('{x} {i:s}'.format(x=1905, i="Papers"))
# Output: 1905 Papers


print('{i:.0E} => [{x:<6s}]'.format(**dict(i=10, x='albert')))

# Output: 1E+01 => [albert]
```

```
from datetime import date, timedelta

print(date(2022, 11, 16) - date(2022, 10, 21))

# Output: 26 days, 0:00:00

print(date(2022, 10, 21) + timedelta(26))

# Output: 2022-11-16

print(date(2022, 11, 16) - timedelta(26))

# Output: 2022-10-21
```

```
from collections import OrderedDict

x = OrderedDict()

x['alan'] = 1

x['Alex'] = 2

x['john'] = 3

x['mary'] = 4

for i in x:

 print(i, x[i])

# Output:

       alan 1

       Alex 2

       john 3

       mary 4


import json

print(json.dumps(x))

# Output: {"alan": 1, "Alex": 2, "john": 3, "mary": 4}
```

```
salary = {

 'Alan': 24000,

 'Albert': 5000,

 'Mary': 2000,

}

print(min(zip(salary.values(), salary.keys())))

# Output: (2000, 'Mary')

print(max(zip(salary.values(), salary.keys())))

# Output: (24000, 'Alan')
```

```
x = {
 'a' : 11,
 'b' : 12,
 'c' : 13
}

y = {
 'd' : 10,
 'a' : 11,
 'b' : 12
}

# prints the keys in common
print(x.keys() & y.keys())
# Output: {'a', 'b'}
# prints the keys in x that are not in y
print(x.keys() - y.keys())
# Output: {'c'}
# prints the (key, value) pairs in common
print(x.items() & y.items())
# Output: {('b', 12), ('a', 11)}
```

```
x = 'data.csv'
print(x[-4:] == '.csv')
# Output: True

x = 'https://www.wikipedia.org'
print(x[:5] == 'http:' or x[:6] == 'https:' or x[:4] == 'ftp:')
# Output: True
```

```
x = {'a': 1, 'c': 5 }
y = {'b': 2, 'd': 4 }

from collections import ChainMap
z = ChainMap(x,y)
print(z['a']) # Output: 1
print(z['b']) # Output: 2
print(z['c']) # Output: 5
print(z['d']) # Output: 4
```

```
x= 'Albert; Einstein, was, a, theoretical, physicist'

import re

print(re.split(r'[;,\s]\s*', x))

# Output: ['Albert', 'Einstein', 'was', 'a', 'theoretical', 'physicist']
```

```
import re

x = 'https://www.wikipedia.org'

print(re.match('http:|https:|ftp:', x))

# Output: <re.Match object; span=(0, 6), match='https:'>
```

```
from fnmatch import fnmatch

print(fnmatch('1.csv', '*.csv'))

# Output: True

print(fnmatch('data.txt', '?ata.txt'))

# Output: True

print(fnmatch('file1.txt', 'file[0-9]*'))

# Output: True

x = ['file1.txt', 'file.pdf', 'file2.txt', 'file.doc']

print([i for i in x if fnmatch(i, 'fil*.txt')])

# Output: ['file1.txt', 'file2.txt']
```

```
x = 'Albert Einstein was a German-born theoretical physicist'

print(x=='Albert')

# Output: False

print(x.startswith('Albert'))

# Output: True

print(x.endswith('st'))

# Output: True

# prints the location of the first occurrence of 'or'

print(x.find('or'))

# Output: 30
```

```
x = '12/29/1992'
y = 'Dec 29, 1992'

import re
if re.match(r'\d+/\d+/\d+', x):
   print('Albert')
else:
   print('Alan')

if re.match(r'\d+/\d+/\d+', y):
   print('Einstein')
else:
   print('Turing')
```

Output: Albert

Output: Turing

```
import re
x = 'Albert Einstein (3/14/1879 – 4/18/1955).'
print(re.compile(r'\d+/\d+/\d+').findall(x))
# Output: ['3/14/1879', '4/18/1955']
```

```
import re

x = 'Albert Einstein (3/14/1879 – 4/18/1955).'

print(re.sub(r'(\d+)/(\d+)/(\d+)', r'\3-\1-\2', x))

# Output: Albert Einstein (1879-3-14 – 1955-4-18).
```

```
print('Acia\u00f1o') # Output: Aciaño
print('Acya\u0303o') # Output: Acyão
```

```
import re

x = 'ALBERT EINSTEIN, albert einstein, Albert Einstein'

print(re.findall('einstein', x, flags=re.IGNORECASE))

# Output: ['EINSTEIN', 'einstein', 'Einstein']

print(re.sub('einstein', 'alan', x, flags=re.IGNORECASE))

# Output: ALBERT alan, albert alan, Albert alan
```

```
x = 'Albert'

y = 'Einstein'

print(x + ' ' + y)

# Output: Albert Einstein

print('{} {}'.format(x,y))

# Output: Albert Einstein

print(','.join(str(i) for i in ['James', 19, 15]))

# Output: James,19,15
```

```
x = 987654321

print(round(x, -1))

# Output: 987654320

print(round(x, -2))

# Output: 987654300

print(round(x, -3))

# Output: 987654000
```

```
x = '''Alan Mathison Turing was an
English mathematician, computer scientist, logician,
cryptanalyst, philosopher and theoretical biologist.'''


import textwrap

print(textwrap.fill(x, 50))

print(textwrap.fill(x, 30))
```

```
Alan Mathison Turing was an  English
mathematician, computer scientist, logician,
cryptanalyst, philosopher and theoretical
biologist.


Alan Mathison Turing was an
English mathematician,
computer scientist, logician,
cryptanalyst, philosopher and
theoretical biologist.
```

```
x = '''Alan Mathison Turing was an
English mathematician, computer scientist, logician,
cryptanalyst, philosopher and theoretical biologist.'''

import textwrap
print(textwrap.fill(x, 50))
print(textwrap.fill(x, 50, initial_indent=' '))
print(textwrap.fill(x, 50, subsequent_indent=' '))

# Output:

Alan Mathison Turing was an  English
mathematician, computer scientist, logician,
cryptanalyst, philosopher and theoretical
biologist.

 Alan Mathison Turing was an  English
mathematician, computer scientist, logician,
cryptanalyst, philosopher and theoretical
biologist.

Alan Mathison Turing was an  English
 mathematician, computer scientist, logician,
 cryptanalyst, philosopher and theoretical
 biologist.
```

```
print(sum([1.23e+18, 1, -1.23e+18]))
# Output: 0.0

import math
print(math.fsum([1.23e+18, 1, -1.23e+18]))
# Output: 1.0
```

```
from fractions import Fraction

x = Fraction(7, 2)

y = Fraction(9, 18)

z = x * y

print(z) # Output: 7/4

print(z.numerator) # Output: 7

print(z.denominator) # Output: 4
```

```
import numpy as np

for i in np.arange(0, 4, 0.5): print(i)

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
```

Output

```
import numpy as np

print(np.matrix([[11,-22,23],[0,54,65],[87,98,-19]]))

# Output:

            [[ 11 -22  23]

             [  0  54  65]

             [ 87  98 -19]]
```

```
# prints the file size and modification date

import time, os

print(os.path.getsize('C:/Users/Manju/1.txt'))

print(time.ctime(os.path.getmtime('C:/Users/Manju/1.txt')))
```

```
25

Fri Jul  8 04:53:18 2022
```

```
import numpy as np

for i in np.arange(0, 1, 0.125): print(i)

# Output:

            0.0

            0.125

            0.25

            0.375

            0.5

            0.625

            0.75

            0.875
```

```
x = [11, 12, 13, 14]

for i in reversed(x):

  print(i, end=' ')

# Output:

      14 13 12 11
```

```
x = ['A', 'B', 'D']
from itertools import permutations
for i in permutations(x, 3):
   print(i)


# Output:
             ('A', 'B', 'D')
             ('A', 'D', 'B')
             ('B', 'A', 'D')
             ('B', 'D', 'A')
             ('D', 'A', 'B')
             ('D', 'B', 'A')
```

```
x = ['A', 'B', 'D']
from itertools import permutations
for i in permutations(x, 2):
   print(i)


# Output:
             ('A', 'B')
             ('A', 'D')
             ('B', 'A')
             ('B', 'D')
             ('D', 'A')
             ('D', 'B')
```

```
x = ['A', 'B', 'D']
from itertools import combinations
for i in combinations(x, 3):
   print(i)


# Output: ('A', 'B', 'D')
```

```
x = ['A', 'B', 'D']
from itertools import combinations
for i in combinations(x, 2):
   print(i)


# Output:
             ('A', 'B')
             ('A', 'D')
             ('B', 'D')
```

```
names = ['Albert', 'Alan', 'John']
age = [25, 56, 89]
for x, y in zip(names, age):
  print(x, '=', y)
```

```
# Output:
             Albert = 25
             Alan = 56
             John = 89
```

```
print('Papers', 1905, 1915)

# Output: Papers 1905 1915

print('Papers', 1905, 1915, sep=',')

# Output: Papers,1905,1915

print('Papers', 1905, 1915, sep=',', end='!!\n')

# Output: Papers,1905,1915!!
```

```
import io

print(io.StringIO().write('Albert'))

# Output: 6

print(io.StringIO().write('Alan'))

# Output: 4

print(io.StringIO().write('Joe'))

# Output: 3

print(io.StringIO('Albert').read(4))

# Output: Albe

print(io.StringIO('Albert').read())

# Output: Albert
```

```
# list temporary directories

import tempfile

print(tempfile.gettempdir())
```

```
# create temporary files and directories using mkstemp() and mkdtemp()

import tempfile

print(tempfile.mkstemp())

print(tempfile.mkdtemp())
```

```
# Output:

(10, 'C:\\Users\\Manju\\AppData\\Local\\Temp\\tmpbsmjzodf')

C:\Users\Manju\AppData\Local\Temp\tmp3d75il4w
```

Last-In, First-Out: A data structure processing method in which the newest element is handled first and the first element is handled last

Homogeneous data structures	Heterogeneous data structures
Can only store a single type of data	Can store more than one type of data

$$\textbf{Odds} = \frac{\text{probability of success}}{\text{probability of failure}}$$

$$\textbf{Logit Function} = \log\left(\frac{\text{probability of success}}{\text{probability of failure}}\right)$$

Sigmoid or **Logistic Function** →The inverse of the Logit Function

Normally distributed data

Data that defines with having Mean = Median = Mode

Suppose we have 25 samples, where 15 are positive and 10 are negative:

$$\textbf{Entropy} = -\frac{15}{25}\log_2\left(\frac{15}{25}\right) - \frac{10}{25}\log_2\left(\frac{10}{25}\right) \approx 0.97$$

A measure of uncertainty and the goal of ML model is to reduce uncertainty

```
a = 'AlbertEinstein1905Papers'
print(a[:3] + a[3:])

# Output: AlbertEinstein1905Papers
```

Prior probability

$$\frac{\text{Number of Desired Outcomes}}{\text{Total Number of Outcomes}}$$

Interpreted Language:

Does not require compilation before execution

Manifold Learning:

The geometry of the data is utilized to make the algorithms more efficient

Change Data Capture

The process of identifying and tracking changes made to data in a database

Data as a service: Providing data to users on-demand via the cloud

A/B testing

(Split testing or bucket testing)

A method of comparing 2 versions of a web application against each other to figure out

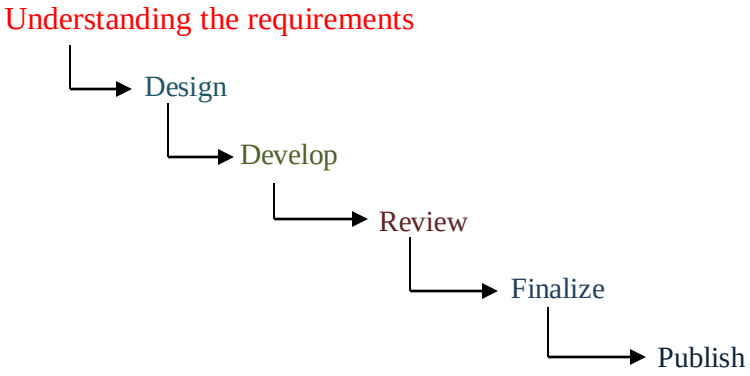
which one is more effective

which one performs better

- **Data in motion:** Any digital information that is being moved over a network from one location to another
- **Data at rest:** Any digital information that is not being moved over a network from one location to another and resides in a database or a file system

- **In-memory database:** Database that depends primarily on memory for data storage
- **In-memory processing:** Processing of data stored in an in-memory database
- **In-memory computing:** Running computer calculations entirely in computer memory

Documentation Development Life Cycle



Empirical Risk Minimization (ERM):

A learning method in which the model with the lowest average error over the training set is chosen

Holdout data: A sample of categorized historical data completely separated from the data sets used to train and test **supervised ML models**

Inter-rater agreement → The degree to which 2 or more persons agree

One hot encoding

A process of converting categorical variables to numerical values so that the ML model can understand them

Serial correlation: A pattern in which each value is influenced directly by the value next to it or preceding it in a sequence

Machine learning

Teaching **computing machines** how to learn from data to make predictions or decisions

Learning-to-Learn

The machines learning themselves how to learn

Loss = Actual value – Predicted value

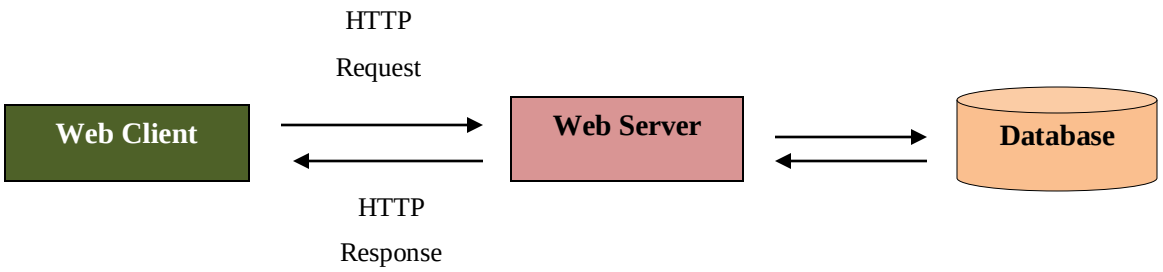
The lower the loss, the better a model

GIGO

(Garbage In, Garbage Out)

The quality of output is determined by the quality of the input

Request-Response Model



```
a = {1,2,3,4,5,6}
b = {5,6,7,8,9}
print(a.intersection(b))
```

Python code to print the common elements between the 2 sets

```
x,y=2,3
print("Alan") if x>y else print("Turing")

# Output: Turing
```

```
print('EINSTEIN'.isupper())
print('einstein'.islower())
```

Output on the screen:

True

True

- **Binary classification** → predicts one of 2 possible outcomes
  - **Example:** Is the condition true or false?
- **Multi-class classification** → predicts one of multiple possible
  - **Example:** Is this a picture of an apple, book, rat or mice?

Regression algorithms	Classification algorithms
Used to predict: what will be the temperature tomorrow?	**Used to predict: will it be hot or cold tomorrow?**

Jenkins Pipeline:

Start → Build → Test → Deploy → End

Ruby Program:

```
a = 26
unless a>=25
   puts "a is less than 25"
else
  puts "a is greater than 25"
end
```

Output:

a is greater than 25

$$\textbf{t-value} = \frac{\text{variance between groups}}{\text{variance within groups}}$$

Larger t-value → different groups

Smaller t-value → similar groups

Single-value attribute	Multi-valued attribute
Can have only one possible value	Can have many different values
Student → Roll number	Student → Address → zip code, street address, state, country, etc.

Dialog Management Systems (DMS):

Systems that analyze and contextualize human-like discussions between **Chatbots** and actual humans using natural language processing

Database Stress Testing:

One of the approaches for evaluating database performance is to put it under a significant strain that could cause the database system to fail at some time

- **Cold Migration:** The method of transferring a virtual machine that is shut down, together with its configuration and data, from one host to other

- **Hot Migration:** The method of transferring a virtual machine that is powered on, together with its configuration and data, from one host to other

Cross Site Request Forgery (CSRF):

A web security flaw that allows an attacker to persuade users to perform unwanted actions on a web application that they did not intend to do

Negative Evidence:

Evidence that predict the non-occurrence of event

Non-linearity:

The output is not proportional to the input

Prospective Data Analysis:

A mathematical approach for predicting future events based on previous performance or characteristics

Programming Assistant:

A programming framework that assists a programmer to handle the non-clerical aspects of programming

If 2 sets have no elements in common, they are said to be disjoint. **For example:** {2, 4, 5} and {6, 7, 9} are disjoint sets, while {2, 4, 5} and {1, 5, 8} are not disjoint.

Negative response → A request not processed successfully

- **Mandatory entry field:** A field in which a user must enter at least one character
- **Mandatory-fill field:** A field a user must fill in completely

Return code: A return statement that terminates a function's execution and returns control to the calling function

Run time:

The length of time that a computer program is running

Run-time environment → Execution environment

Nit: A minor flaw that can be easily corrected

Network congestion: An undesirable overload condition arises when a network is unable to appropriately handle the traffic pouring through it

Iterative server:

A server that can handle only one connection at a time

Lock: The method of ensuring data integrity by preventing several users from simultaneously accessing or modifying the same data

Maximize: An option that enlarges a window to its maximum size

Metadata: Information that describes the characteristics of a stored data

Foreground processes: Processes that need a user to initiate them or to interact with them

Background processes:

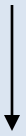
Processes that run without the intervention of a user

- **Log off:** Disconnect from a computer system, database or network
- **Log on:** Connect to a computer system, database or network

Mapping: The process of transforming data transmitted in one format by the sender into a format that the receiver can understand

Kludge

(A quick-and-dirty fix)



A quick awkward solution to a problem

Memory leak: An unintended kind of memory consumption occurs when a computer program fails to free an allocated block of memory when it is no longer needed

Memory is allocated, but not deallocated

```
c = lambda a,b: a+b
print(c(3,6))

# Output: 9
```

Traditional programming approach:

Input + Program → Output

Data science approach:

Input + Output → Program

Size of Test

(The probability of committing a type I error)

The probability of rejecting the null hypothesis when the null hypothesis is true

- Error = (observed value − actual value)
- Residual = (observed value − predicted value)

Power of Test

(The probability of avoiding a type II error)

The probability of rejecting the null hypothesis when the null hypothesis is false

Data Literacy

An individual's ability to extract meaningful information from data

Uplift modeling: A ML technique which predicts how every customer is likely to respond to marketing activities an organization does to boost the selling of a product or service to customers

Dependent Data Mart	Independent data mart
constructed with the use of a central data warehouse	constructed without the use of a central data warehouse

Monolithic kernel: An operating system framework in which the entire operating system runs in kernel space

Microkernel: A small piece of software capable of providing the processes required to implement an operating system

S3

(Simple Storage Service)

↓

Allows developers and IT Teams to store and retrieve any volume of data from any location on the internet at any time

AWS Availability Zone: An Amazon data center's physical location

AWS Region: A group of Amazon Data Centers

Data sparsity:

The concept of not perceiving enough data in a dataset

Can be used to learn an efficiently compressed and encoded representation of raw data

Autoencoder: A type of artificial neural network with the number of output units in the output layer is equal to the number of input units in the input layer

Enterprise Data Model:

A kind of data model that displays a unified perspective of all data used by the enterprise firm

BLEU

(BiLingual Evaluation Understudy)

↓

A method for automatic evaluation of **Machine Translation**

The process of **automatic translation** of text from one language to another without any human intervention

Cloud Enablement: The process of creating, deploying and operating

"the existing infrastructure, resources and services that a enterprise is currently using on-premises"

on the cloud

CAPTCHA

(Completely Automated Public Turing Test to tell Computers and Humans Apart)

↓

A system that distinguishes humans from robotic computer programs in order to detect fraudulent activities

Overheating

Most computer systems are configured to automatically shut down or reboot if the integrated circuits, graphics card, or other hardware components get too hot

DataOps

(Data Operations)

↓

The data management techniques that enable an enterprise organisation to quickly and precisely automate data flow between managers and data consumers within the company

Reverse ETL: A method of feeding data from a data warehouse that has been cleansed and processed back into business applications for operations and forecasting

Pivot table: A powerful tool for rearranging huge lists of data, as well as summarizing and analyzing them

Calibration: A comparison of a system's actual output and its expected output

Class-imbalanced dataset: Any dataset containing an unbalanced distribution of classes

For example: A class-imbalance problem exists in a **Covid 19** dataset in which 0.9997of cases have positive labels and 0.0003 cases have negative labels

ML pipeline: A series of automated steps for training and deploying a **ML model**

Bucketing → Data organizing technique

Given a weight survey data, weights of participants ranging from 50.0 to 60.0 are grouped together in one bucket, 60.1 to 70.0 in another, and so on.

```
print('in' in 'Hawking')
print('al' not in 'Hawking')
print({1,2,3}=={1,2,3})
print({1,3,5}=={1,2,3})
```

Output on the screen:

True
True
True
False

Go Program:

```
package main

import "fmt"


func main() {

fmt.Println(" ", 10 + 11)

}

# Output: 21
```

Ruby program:

```
$x = 1
$y = 4

while $x < $y do
   puts("#$x")
   $x = $x +1
end
```

Output:

1

2

3

```
from random import shuffle
a = ['Albert', 'Likes', 'To', 'Read', 'In', 'Garden']
shuffle(a)
print(a)

# Output:
                    'Garden', 'Read', 'In', 'Albert', 'Likes']
```

AndroidManifest.xml file

A XML file that contains information about an Android application

Android Asset Packaging Tool:

A development tool that allows developers to view, generate and update APK, as well as zip and jar files

Proof of Concept: A strategy used by enterprise organizations to test the viability of an idea or concept

Software prototype: A model of how an actual software product will function and appear

Feasibility study: A methodology of evaluating a software application in terms of how feasible software development will be for the company

SCD

(Slowly Changing Dimensions)

A dimension that saves and maintains data over time in a data warehouse

Test Data Management:

The process for generating, maintaining, and sending test data to application development teams

Data Analysis Platform:

A platform that combines services and technologies to store, manage and analyze data to address the needs of large data-driven enterprise organizations

If we use height, weight and sweet intake to forecast the probability of **Diabetes mellitus**:

Height, weight and sweet intake represent features

Binomial Distribution: The discrete probability distribution that yields only 2 possible outcomes in an experiment.

For example:

- If we throw a coin, there are only two possible outcomes: head or tail
- If a test is taken, there are only two possible outcomes: either pass or fail

Manifest variable	Latent variable
A variable that can be directly measured	A variable that cannot be directly measured but deduced through a mathematical model from other variables (that are directly measured)

Markov chain or Markov process: A stochastic model that predicts the probability of a particular event based on previous events

Digital Twin: A software program that generates stimulation that precisely replicates a physical object based on real-world data in order to predict how a physical object would behave

Dynamic model: A model that is trained online with the constantly updated data

Data blending: A method for combining data from various sources into a single dataset for data visualization and analysis

Data downtime:

Duration during which data is incomplete, wrong, missing and unreliable

- **Data Joining:** Combining the data from different tables within the same data source
- **Data fabric:** An architectural design that allows access to data regardless of its location

Stress testing

(How well a system will perform under unfavorable conditions)

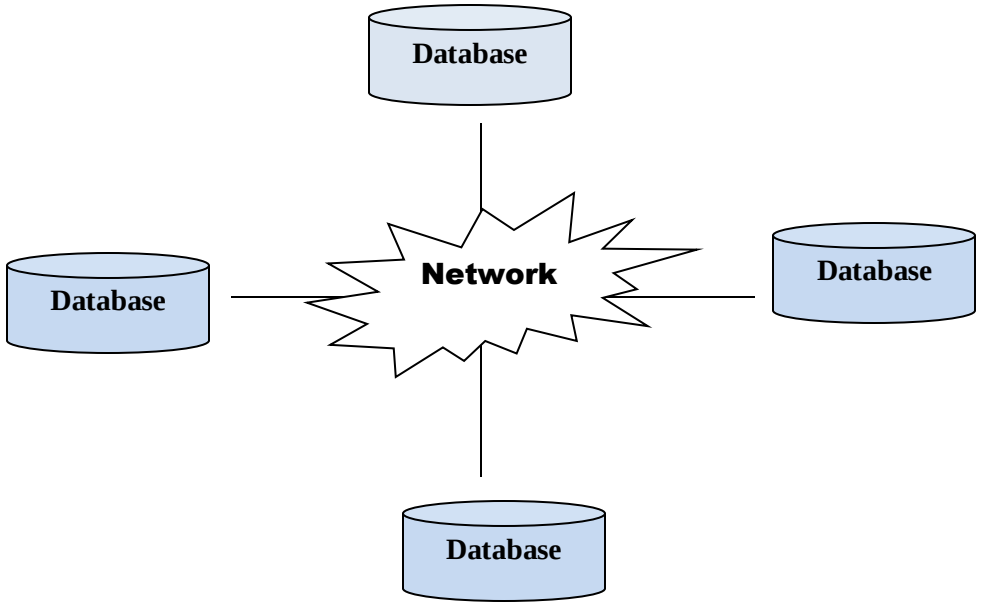
Evaluates whether or not the system is reliable and stable

A real-time operating system (RTOS):

An operating system with 2 main characteristics:

- **Predictability:** The ability to finish all tasks in a pre-determined time
- **Determinism:** The ability to predict how well a task will be completed

Distributed Database:



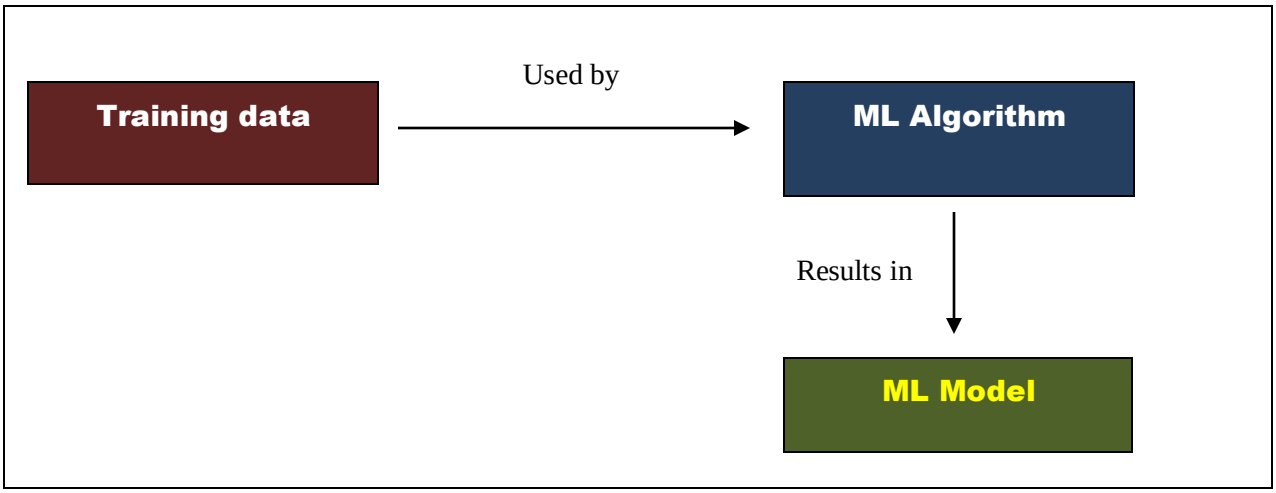
A type of database which consists of multiple databases that are connected with each other via network

Self-documenting code

Well-written code that can be easily understood by another programmer or even by the original author and a developer doesn't have to think about leaving **comments for the reader** to understand what the code is doing and why the code is written the way it is

Symmetric Encryption	Asymmetric Encryption
One key can be used for both encryption and decryption	To encrypt and decode data, two keys are required

Statistical Bias: The difference between the result obtained from the Statistics and the actual fact



Routing analysis

A kind of network analysis that discovers the most efficient path between two or more network locations

Transactional data: The unpredictably changing data gathered during transactions

Parallel query: A query that runs on numerous system processes to improve performance

Parallel method invocation

(PMI)

A programming concept that allows an application to simultaneously invoke numerous functions or methods

Parallel data analysis

Breaking down an analytical problem into smaller parts and simultaneously running algorithms on each of them

Parallel processing: The ability to perform numerous tasks at once

The revenue generated by selling the products and services

Operational Efficiency = $\frac{\text{Output gained from the business}}{\text{Input to run a business operation}}$

The costs of producing the products and services

NewSQL

A database language that combines and builds on the concepts and principles of both SQL and NoSQL

Mashup: A web application that combines data or functionality from multiple sources to create a single new service

Mahout → A data mining library

Location analytics: Associating geographic data with a company's data assets in order to gain more meaningful insights into spatial questions

Natural Language Processing

Make machines understand human speech

```
# Python code to print numbers ranging from 1 to 20
for i in range(1,21):
print(i)
```

SQL injection: A hacking technique that uses malicious SQL code to steal information from your database

Go program:

```
package main
import "fmt"

func main() {
   for i := 0; i < 2; i++ {
   fmt.Printf("Albert Einstein\n")
}
}
```

Output:

Albert Einstein

Albert Einstein

Market basket analysis:

A data mining technique used by retailers to identify purchases that commonly happen together. **For example,** customers who purchase pencil and rubber also purchase sharpener and scale. Or customers who purchase shoes might also purchase socks.

NoSQL database: A database technology that stores information in **JSON** documents

JavaScript Object Notation

Atomicity

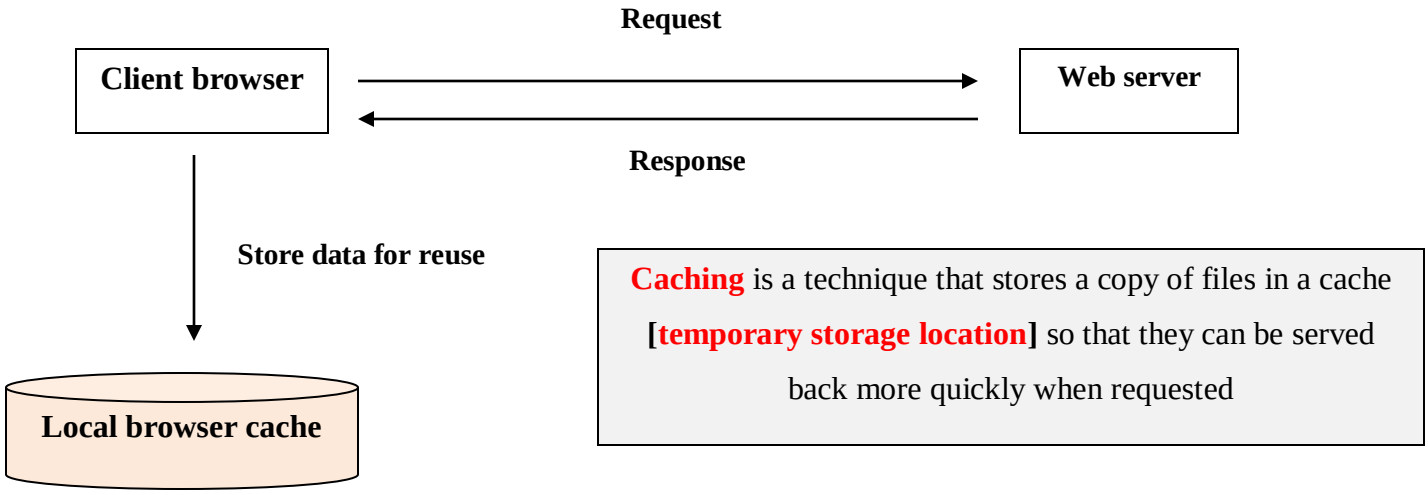
If any operation is performed on the data, either it should be performed completely or should not be performed at all

```
// Go program to demonstrate the use of an infinite loop

package main
import "fmt"


func main() {
   for {
     fmt.Printf("Albert Einstein\n")
}
}
```

Cache

A hardware or software component that stores data so that future requests for that data can be served faster

Web browsers use a cache to store web pages so that the user may view them again without reconnecting to the Web.



Caching is a technique that stores a copy of files in a cache **[temporary storage location]** so that they can be served back more quickly when requested

Data buffer

A region of a memory used to temporarily store data while it is being moved from one place to another

Computer cracker

A Person who attempts to access computer systems without authorization

Crash

An event wherein the **operating system** or a **computer application** stops functioning properly

Streaming

Watching any video content or listening to audio content that is stored on the **World Wide Web**

Directory

A folder where all files are kept on a computer

Firewall

A **network security system** that monitors incoming and outgoing network traffic and prevents unauthorized access to a network

HTTP	HTTPS
Hypertext Transfer Protocol	Hypertext Transfer Protocol Secure
Does not provide the security of the data	Ensures the security of the data

Internet service provider

Company that provides Internet connections and services to individuals and organizations

HTTP

http://www.abc.com/

Password: xyz@123

No password encryption

Hacker sees "xyz@123"

HTTPS

https://www.abc.com/

Password: xyz@123

Password encryption

Hacker cannot see "xyz@123"

Multimedia

A Software program that combines text and graphics with sound, video and animation

Freeware: software that anyone can download from the Internet and use for free.

Shareware give users a chance to try the software before buying it

Platform

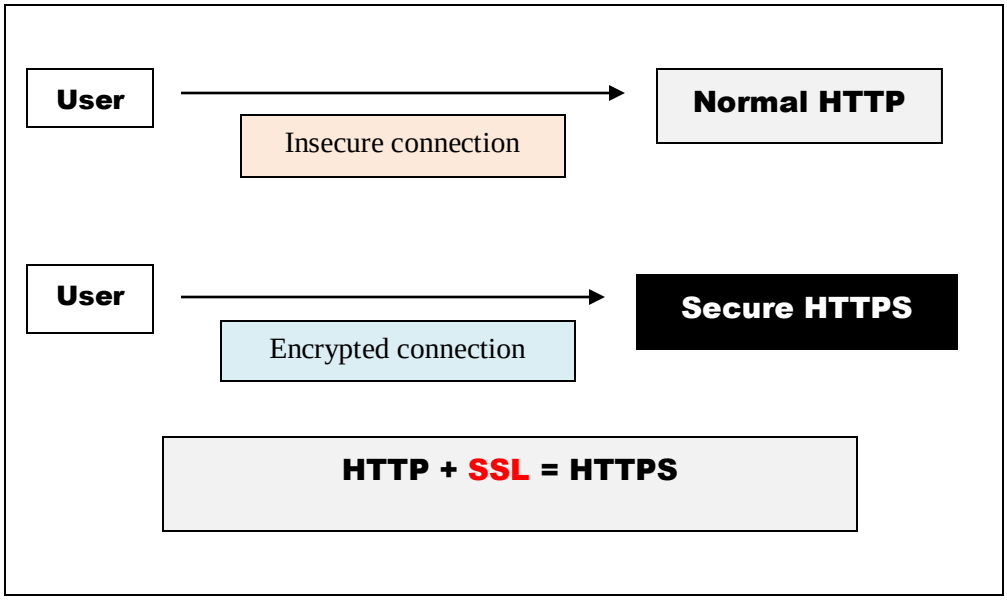
An environment in which a piece of software is executed

Virus

Software that infects other software by modifying it

Secure Sockets Layer (SSL)

Encryption-based Internet security protocol that allows Internet users to send encrypted messages across the Internet



Help our website or piece of content rank higher on Google

SEO

(search engine optimization)

The process of improving our website to increase its visibility when people search for **products** or **services** related to our business in major search engines

Spam

Unwanted, unsolicited email messages that gets sent out in bulk for marketing purposes

Wi-Fi

Wireless technology used to connect computers, tablets, smartphones and other devices to the internet via a wireless signal

Wired Equivalent Privacy (WEP)

A security protocol for Wi-Fi networks

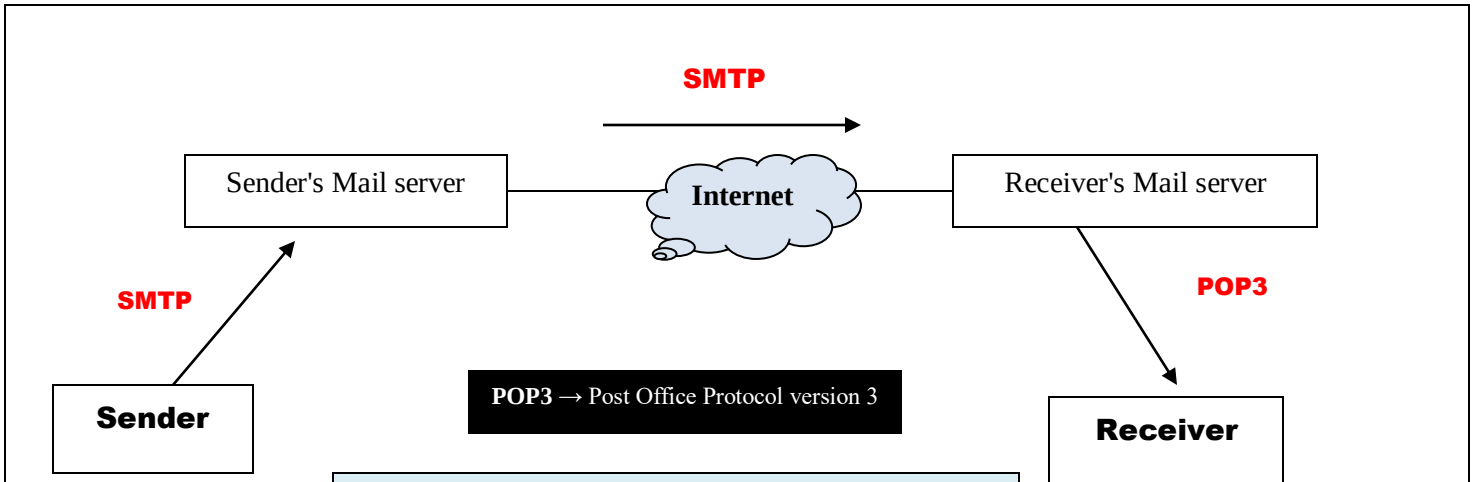
Add security to Wi-Fi networks by encrypting data

Wi-Fi Protected Access (WPA): security protocol designed to create secure wireless (Wi-Fi) networks. It provides better data encryption and user authentication than Wired Equivalent Privacy (WEP)

- DNS → Domain Name Service
- HTTP → Hyper Text Transfer Protocol
- SMTP → Simple Mail Transfer Protocol
- POP → Post Office Protocol
- FTP → File Transfer Protocol

Post office protocol (POP)

Message request protocol in the Internet world that provides **e-mail client** (receiver) the ability to fetch and receive email from an **e-mail server**



Cloud migration

The transfer of organization's data, IT resources, digital assets, services and other business elements into a cloud computing environment

SMTP

Internet standard communication protocol used for sending the emails efficiently and reliably over the internet

While sending the mail message, **SMTP** is used 2 times:

- Between the sender and the sender's mail server
- Between the sender's mail server and the receiver's mail server

Cloud Computing

Based on deployment:

- Private
- **Public**
- Hybrid

Based on service model:

- Infrastructure as a service (IaaS)
- **Platform as a service** (PaaS)
- Software as a service (SaaS)

Advantages:

- Easy Scalability
- Enhanced Security
- Easy Accessibility
- Cost Effective and Data is routinely saved in the cloud

Non SaaS Application	SaaS Application
Application logic runs on user's computer	Application logic runs in the cloud

Data migration

The process of moving data from one storage system or computing environment to another

Cloud Migration

Plan → Design → Migrate → Optimize → Assessment

Cloud portability

The ability to move applications and its associated data from one cloud computing environment to another with minimal disruption and downtime

Cloud interoperability

The ability of the **systems** to work efficiently and collaborate effectively across different cloud platforms

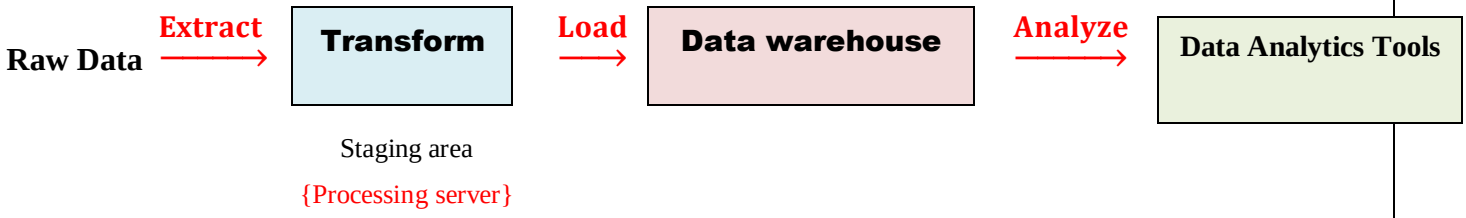
Data orchestration

{Organize the data → Transform the data → Make the data available to the tools that need it}

The process of **bringing data together** from multiple data storage locations – combining and organizing it – and preparing it for **data analysis**

ETL [Extract, Transform and Load]	ELT [Extract, Load and Transform]

ETL:



- ETL transforms data on a separate processing server
- ELT transforms data within the data warehouse itself

ELT:

- **ETL** does not transfer raw data into the data warehouse
- **ELT** sends raw data directly to the data warehouse

Automation	Orchestration
Setting up one task to run on its own {automating one task}	Automating many tasks together {automation not of a single task but an entire IT-driven process}

Database

↓

Designed to record data

{Capture and maintain the data}

Data Warehouse

↓

Designed to analyze data

{Explore the data}

- **Infrastructure as a service (IaaS):** Renting servers, virtual machines, storage, networks and operating systems
- **Platform as a service (PaaS):** Supplying an on-demand environment for developing, testing, delivering and managing software applications
- **Software as a service (SaaS):** Delivering software applications over the Internet

Data redundancy

Storing the same piece of data more than once in the database

Data exploration

A way to get to know data before working with it

Data Presentation

↓

The organization of data into tables, graphs or charts so that an informed decision can be made based on them

File transfer protocol (FTP)

Used to transfer files via the internet from one host to another host

Data mapping

The process of modeling or illustrating how data will move from a source data store to a target data store

Data cleaning

The process of correcting and deleting inaccurate records from a database or table

Data integration

↓

The process of bringing data from different sources together to provide users with a unified view

Null: a character with no value or missing value. If $a= " " and $b= "1", $a possess a null value.

Logical reasoning

Solving problems by using common sense and logic

Inductive research approach

↓

Aims at developing a theory

Deductive research approach

↓

Aims at testing an existing theory

5 steps to draw a conclusion

- Collect the information
- Analyze the information
- Form conclusion
- Support conclusion
- Defend conclusion

Generalization

↓

A quick way of solving new problems based on previous problems we have solved.

Version control

↓

The practice of tracking and managing changes to software code

Procedure	Function
A block of code that is called to perform a task	A block of code that is called to perform a task and will return one or more values

Usability testing

A method used to evaluate how easy and user-friendly a software application is

Plan the test → Recruit the desired number of testers → Prepare → Set up testing environment → Conduct the test → Analyze data → Report results

MEAN Stack	MongoDB + Express.js + Angular.js + Node.js
MERN Stack	MongoDB + Express.js + React.js + Node.js
MEVN Stack	MongoDB + Express.js + Vue.js + Node.js
LAMP Stack	Linux + Apache + MySQL + PHP

Pair programming

↓

2 developers use just one machine to work − meaning they work simultaneously on a single block of code. Person who writes code is called a **driver** and a person who observes and checks each line of the code is called **navigator**.

Command line

A text interface that takes in commands and passes on to the computer's operating system to run.

Object-oriented database

Object oriented programming

- Polymorphism
- Inheritance
- Encapsulation and abstraction

+

Relational database

- Integrity [the overall accuracy, completeness and reliability of data]
- Concurrency [the ability of the database to support multiple users]
- Query processing

MVC

[Model-View-Controller]

↓

An architectural pattern that separates an application into **3 main components**:

1. data (model)
2. user-interface (view)
3. application logic (controller)

Cybersecurity

↓

The practice of protecting systems, network data and programs from unauthorized access

Grayware

↓

Malicious software that is not as destructive as a **virus** – but can worsen the performance of computer and cause security risks

Hyperlink

A piece of text that when clicked takes the user to a webpage

Hypervisor

A crucial piece of software that makes virtualization possible

↓

Software that creates and runs virtual machines

Internet Connection Sharing (ICS)

A **Windows service** that allows several computers to share one Internet connection on a local area network

IP address

[Internet Protocol address]

A unique address that identifies a computer on the internet

Networking as a service (NaaS)

↓

Renting of network services to customers who don't want to build their own networking infrastructure

Nameserver

↓

A server that translate domain names into IP addresses

- **Network adapter** → A device that connects our computer to a network
- **SAN (storage area network)** → A network of data storage devices

QoS (Quality of Service)

↓

The technology that manage data traffic via the network

Serial Advanced Technology Attachment (SATA)

An interface for transferring data between a computer's motherboard and storage devices

Virtual memory is created when a computer is running many processes at once and RAM is running low.

Trojan horse

↓

Malicious software that damage, disrupt, steal − or in general inflict some other harmful action on your data or network

Voice over Internet Protocol (VoIP)

A t**echnology** that allows us to make voice calls using a broadband Internet connection

virtual hosting

The practice of hosting more than one website on a single server

- **Compression** → A technique that reduces file size
- **Software bug** → An error or flaw in a computer program
- **Word processor** → A software that allows users to create, edit and print documents
- **Modem** → A hardware device that is used to transfer information between computers via a phone line

Rendering

[image synthesis]

↓

The process of generating an image from the information contained in a file by means of a computer program.

Speech recognition

[speech to text]

↓

The ability of a program to **process human speech into a** readable text

Graphical User Interface

A **graphics-based operating system interface** that enables a person to communicate with a computer by using icons, menus and a **mouse** (to click on the icon or pull down the menus)

Attributes

↓

The characteristics of a file in the context of a file system

For example: if the file has a read-only attribute − the file cannot be modified and can only be read

Meta tags

↓

Pieces of information that tell the search engine more about your webpage and the information it contains

Plugin

A piece of software that gives the browser additional functionality

If we want to watch a video on a website, we may need a plugin to do so. If the plugin is not installed, our browser will not understand how to play the video.

Redirect

Automatic forward to a different URL from the one we originally requested

Responsive design

A web development approach that makes our web content adapt to the different screen sizes and viewports.

Sitemap

A blueprint of our website that help search engines find, crawl and index our entire website's content.

Data Bus

The connection between the **CPU** and **RAM** – literally a collection of electrical wires – used to send information between CPU and RAM

The fetch-decode-execute cycle

The sequence of steps that the CPU follows to process instructions

Fetch instruction

↓

Decode instruction

↓

Execute instruction

Repeat until there are no more instructions

Lossy Compression	Lossless Compression
Reduces the quality	Does not reduce the quality
Data reduction is higher	Data reduction is lower

Embedded testing

The process of verification and validation of both software and hardware

Anti-malware software

Software that detects and removes malicious code

Authorization

The process of granting the permission to user to access a system resource

Data Loss Prevention (DLP)

The practice of detecting and preventing unwanted destruction of sensitive data

Hardening

The process of making computer and network systems more resistant to malicious software

Software Patch

A modification to a program to resolve functionality issues – improve security and performance – and add new features

Root access

Most privileged access to a computer system allowing the use, change and deletion of any and all configuration data

Penetration test

[ethical hacking]

Helps us stay ahead of hackers

An authorized simulated cyber attack on a computer system to evaluate the security of the system

Sensitive information

Data that must be protected from unauthorized access to safeguard the privacy and security of an organization

Data Synchronization

The integration of data between 2 or more devices

Computer worm

A malicious malware that reproduces itself and spreads over network connections

Implementation model

A representation of how a system actually works

Information Base

The main repository of information about the software

Interaction Diagram

A diagram that represent the interactive behavior of a system

Legacy system

Outdated computing software or hardware that is still in use

Middleware

Software that mediates between an application program and a network

Process model

A graphical representation of a process

Workflow

A sequence of tasks that processes a set of data

Object-modeling technique (OMT)

An object modeling approach for software modeling and designing

Software Architecture (What we are building)	Software Design (How we are building)
A detailed plan to produce a software design	A detailed plan to implement software

Recursive self-improvement

↓

The property of software to make improvements on its own ability of making self-improvements

Technological singularity

A condition that could arrive in the near future where technological growth would be out of control and irreversible – ordinary human intelligence would be overtaken by artificial intelligence

Installer

↓

A piece of software that installs files such as applications, drivers or other software – onto a computer's hard disk

Bootstrap loader

The first piece of code that is executed when the computer system is started and is responsible for loading the rest of the operating system

Code refactoring

The process of restructuring computer code without changing its functionality

Evolutionary Computation

An area of computer science that uses a class of algorithms to solve computational problems with a process of trial and error

Heuristics

↓

Problem-solving strategies derived from previous experiences with similar problems – permits an individual to make a decision, come to a sensible conclusion or solve a problem quickly and with minimal mental effort

Ambiguity

Information that can be understood in more than one way

Human-level hardware

Hardware that matches the Human brain's information-processing ability

Human-level software

Software that matches the **Human brain's algorithmic efficiency** for performing the tasks the human brain performs

Computing overhang

↓

A situation where new algorithms can make use of **existing computing power** far more efficiently than before

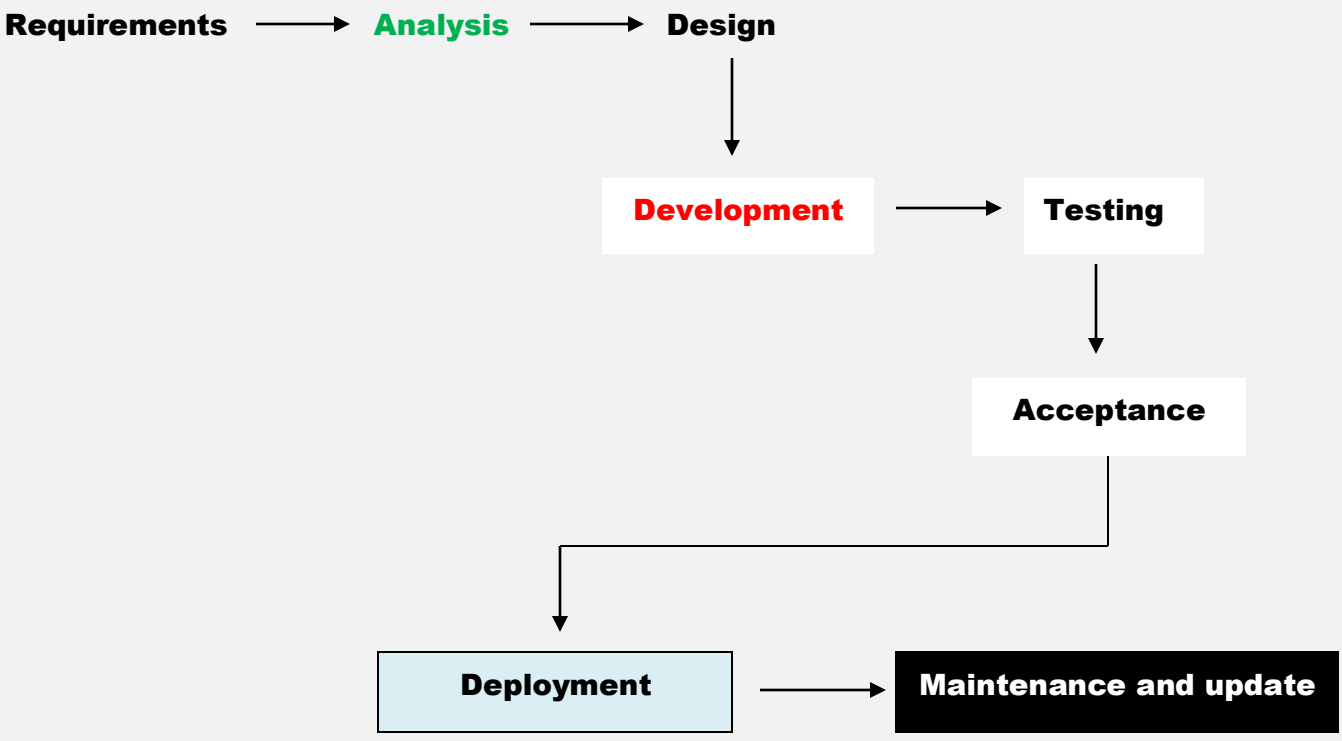
Hardware overhang

A situation where large amounts of hardware being used for other purposes become available for AI

Intelligence explosion

A hypothesized event in which an Artificial intelligence rapidly improves and machines will exceed human levels of intelligence and ability

Software Engineering Cycle:



Adversarial Machine Learning

A cyber-attack that aims to cause a machine learning model to make a wrong prediction with deceptive input

- **Black Box Attack:** The attacker has no knowledge of the machine learning model
- **Grey Box Attack:** The attacker has partial knowledge of the machine learning model
- **White Box Attack:** The attacker has complete knowledge of the machine learning model

Data sanitization

The process of intentionally, permanently and irreversibly deleting or destroying the data stored on a memory device – to ensure it cannot be recovered

Data corruption

↓

The process of data becoming unreadable or invalid

Adversarial Example

↓

A sample of input data which has been adjusted somewhat in a manner that is planned to cause an ML Model to misclassify it

Artificial General Intelligence

A **computational system** that can understand, perform or learn any intellectual task that a human being can

Artificial Superintelligence

↓

AI capabilities superseding human intelligence

Custom Training

↓

The process of training a model to make certain predictions

Facial recognition

The method of identifying or verifying the identity of a person using their face

False Positive (Type I error)	False Negative (Type II error)
ML Model predicts something is true when it is actually false	ML Model predicts something is false when it is actually true

True Positive	True Negative
ML Model predicts something is true when it is actually true	ML Model predicts something is false when it is actually false

False positive rate = $\frac{\text{Number of false positives}}{\text{Number of false positives+ Number of true negatives}}$

False negative rate = $\frac{\text{Number of false negative}}{\text{Number of false negatives+ Number of true positives}}$

True positive rate = $\frac{\text{Number of true positives}}{\text{Number of true positives+ Number of false negatives}}$

True negative rate = $\frac{\text{Number of true negatives}}{\text{Number of true negatives+ Number of false positives}}$

Feature Extraction

↓

The transformation of raw data into the inputs that a particular Machine Learning algorithm requires

Feature selection →

↓

The process of identifying and selecting the most appropriate information from the raw data and removing the irrelevant or less important information when developing a predictive model to reduce overfitting in the model

↓

The model performs well on the training data but does not perform accurately on the test data

Data augmentation

The process of enlarging a dataset with additional data

Cross-validation

A statistical method used to evaluate the performance, effectiveness and accuracy of machine learning models

Regularization

A technique that is used to adjust machine learning model in order to prevent overfitting or underfitting and improve the performance of machine learning model

Ensembling

A technique that combines two or more machine learning models to produce one best predictive model

In Sample Error	Out of Sample Error
The error we get on the same data with which the machine-learning model was built	The error we get on a unknown data

Bias

The difference between the prediction of the values by the Machine learning model and the actual value

Variance

How much a model changes when we train it using a different dataset

Machine learning model with high variance will have the adjustability to match any dataset that is provided to it

Generative Adversarial Network

A machine learning model in which 2 neural networks contest with each other to become more accurate in their predictions

Graphics Processing Unit

↓

A specialized electronic circuit designed to render graphics for display on an electronic device

Image Recognition

The ability of software to identify objects in digital images or video

Optical character recognition: The ability of software to identify the text inside images – such as scanned documents and photos

Autonomous

Machines make their own decisions and potentially take their own actions to solve a problem – without needing human intervention

Backward chaining

(Goal driven approach)

We have a decision and based on the decision we fetch data

Decision ⟶ Data

Forward chaining

(Data driven approach)

We have some data and we make a decision based on the data

Data ⟶ Decision

Big data

Massive collection of data that is too huge in volume or complex to be handled and managed by traditional data processing softwares

Computational learning theory

A branch of artificial intelligence that is primarily concerned with the design and analysis of machine learning algorithms

Corpus

A large collection of text and speech data that can be used to train AI and machine learning systems to perform linguistic tasks

Entity annotation

The process of labeling sentences lacking structure or organization with some information so that a machine can read them

Entity extraction

↓

Information extraction technique that identifies and classifies specific data from unstructured text into predefined categories so that a machine can read it

Natural language generation

The process by which a machine turns highly organized information into text or speech that human beings can understand

Natural language processing

A branch of **artificial intelligence** that is concerned with giving computer program the ability to understand text and speech in much the same way humans can

Natural language understanding

A branch of artificial intelligence that is concerned with giving computer program the ability to understand and interpret human language

Predictive analytics

A branch of statistics that deals with extracting information from data and using it to make predictions about unknown future events

Sentiment analysis

(Emotion artificial intelligence)

↓

The process of determining whether a given text contains negative, positive or neutral emotions

Image Segmentation

The process of identifying and separating different objects in a 3D image

ImageNet

A large database of over 14 million images used for developing computer vision algorithms

$$\textbf{Misclassification Rate} = \frac{\text{Number of incorrect predictions}}{\text{Total number of predictions}}$$

- Misclassification Rate = 0 → zero incorrect predictions
- Misclassification Rate = 1 → completely incorrect predictions

Accuracy = (1 – Misclassification Rate)

Reactive machines

↓

The most basic functionality of AI where the machine can only react to the input it receives – with no ability to use its past experience or previous data to make decisions

Limited Memory

↓

The ability of AI to store previous data and use it to make better predictions

Theory of mind

The ability of AI to understand human thoughts and emotions and use this to build intermediating technology for machine-human interaction

Self-awareness

(Machine has its own consciousness)

↓

The **most advanced type of AI** will be when we witness a machine with human-level consciousness and intelligence

Reinforcement Learning

Machine learning technique in which ML models learn to make better predictions after repeated cycles of trial and error

Short Term Memory	Long Term Memory
A type of memory in a computer that allows storing data temporarily **(RAM)**	A type of memory in a computer that allows storing data permanently **(Hard drive)**

Recurrent Neural Networks

A class of artificial neural networks where the output from previous step are fed as input to the current step

Supervised learning (Task driven)	Unsupervised learning (Data driven)
The output is known while training the ML model **Input and output data is given** **(Highly accurate and used for prediction)**	The output is unknown while training the ML model **Only Input data is given** **(Less accurate and used for analysis)**

Transfer learning: Learning of a new tasks relying on the previous learned tasks

→ Faster, more accurate and need less training data

AI Assurance

The process of testing the behavior of algorithms to protect AI applications from attack or malfunction

Explainability	Interpretability
The process of explaining why and how a machine learning model made a decision	The extent to which a human can understand the cause of a decision made by machine learning model

Commonsense Reasoning

A branch of AI that is concerned with giving computer the human ability to make presumptions about the events and situations humans encounter every day

Visual recognition

The ability of software to identify an object visually

Visual Search

The ability of software to search for information using images

Object Detection

↓

A computer vision technique that identify and locate objects within an image or video

Object Tracking

A **computer vision technique** that track the displacement of a particular object across the entire video

Zero-shot learning

Teach machines to recognize something they have not seen

One-shot learning

Teach machines to do a lot more with a lot less

On-premises Software

↓

The practice of installing and running software on your own hardware on your company's premises

$$\text{Precision} = \frac{\text{Number of true positives}}{\text{Number of true positives} + \text{Number of false positives}}$$

A measure of how often a model predicts something is true when it is actually true

The precision of a machine learning model will be low when

Number of true positives + Number of false positives > Number of true positives

The precision of a machine learning model will be high when

Number of true positives + Number of false positives < Number of true positives

$$\textbf{Positive Predictive Value} = \frac{\text{Number of true positives}}{\text{Number of true positives} + \text{Number of false positives}}$$

$$\textbf{Negative Predictive Value} = \frac{\text{Number of true negatives}}{\text{Number of true negatives} + \text{Number of false negatives}}$$

Search Query

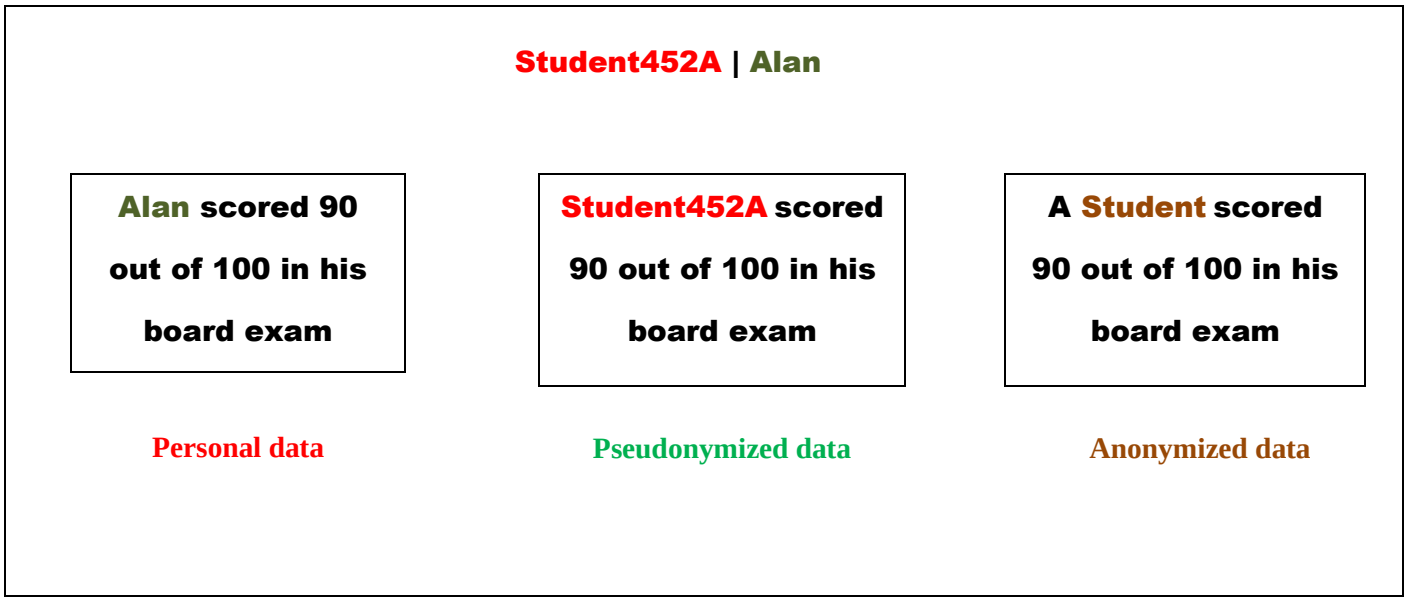
A list of phrases and keywords **users enter in search engines** to find information about things of interest

Selective Filtering

↓

A ML model ignores unwanted information to focus on valuable information

Target Function ⟶ **The end goal of an algorithm**



[Pseudonymization and anonymization of data]

Database Firewalls

A protection technology that effectively protects the sensitive information stored in the databases from malware attacks, data loss and thievery

Augmented intelligence

A subfield of AI machine learning that is designed to enhance human intelligence through the power of machine generated insights rather than replacing it

Edge computing

The method of capturing, processing, analyzing and storing information at or near the physical location where that information is produced or consumed

Deepfake

The AI based technique of creating fake images and videos that we cannot distinguish them from authentic ones

Deployed AI

The method of integrating an AI model into an existing production environment to make real-time business decisions based on data

F1-score

A measure of the model's accuracy

$$\frac{\text{Number of true positives}}{\text{Number of true positives} + \frac{\text{Number of false positives} + \text{Number of false negatives}}{2}}$$

- F1-score = 1 ⟶ Model is considered perfect
- F1-score = 0 ⟶ Model is a total failure

Bayes Theorem:

$$\textbf{P (H | E)} = \frac{P(E|H) \bullet P(H)}{P(E)}$$

H represents a hypothesis and E the evidence

- **P (H | E) – the probability of H given E is true**
- **P (E | H) – the probability of E given H is true**
- **P (E) – the probability of E**
- **P (H) – the probability of H**

Binarization

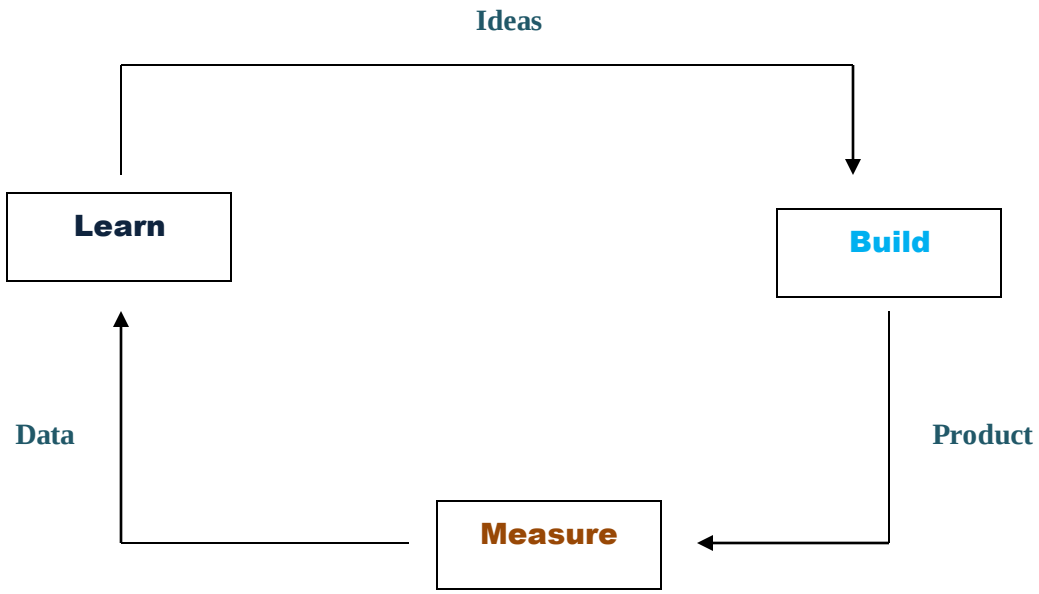
The process of transforming data into binary numbers

Data discretization

The method of converting a large number of data values into smaller ones so that assessment and management of data becomes unchallenging

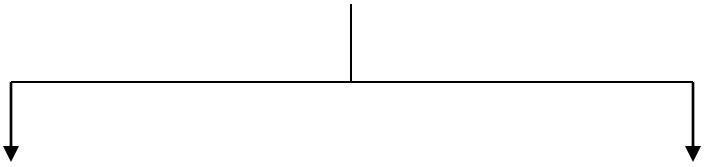
Build-Measure-Learn cycle

(BML cycle)



1. **Build idea into product**
2. **Measure the response of the customers against the product**
3. **Learn from it to improve the product**

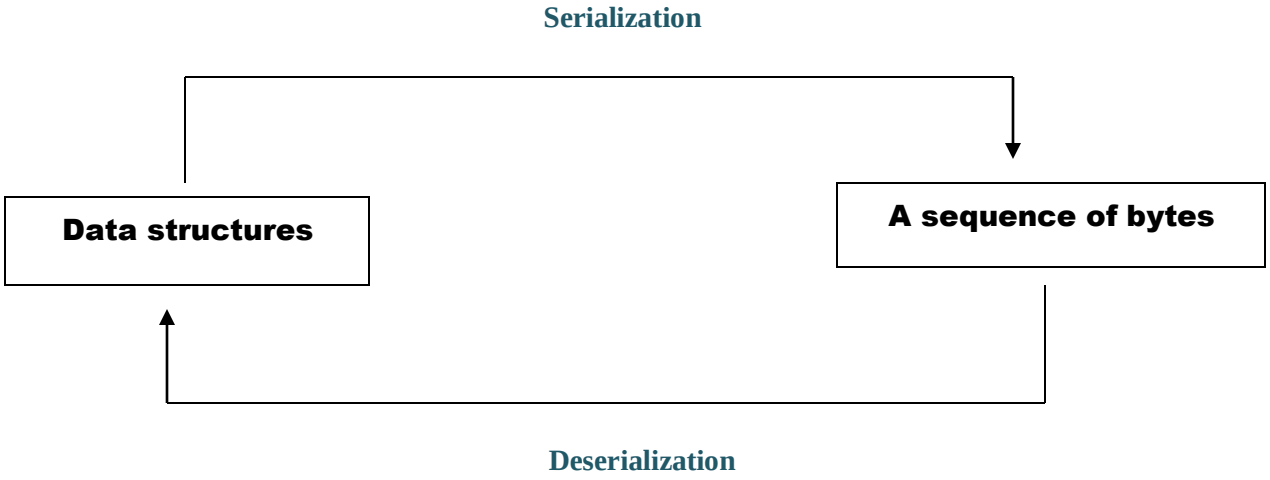
Recommendation system



Collaborative Filtering	Content-based filtering
Suggest a user the products or services based on the likings of the other users with similar tastes. **For example**, if you are watching a movie on **Netflix**, then it is likely that the movie liked by the other users with the similar taste will be suggested to you	Suggest a user the relevant products or services based on their preferred features of other products or services. **For example** , if you search for white clothes frequently on **Shopclues**, then other clothes of the same colour will be suggested to you

Data Preparation Process

Data collection → Data Preprocessing → Data Transformation



Rule-Based Approach

↓

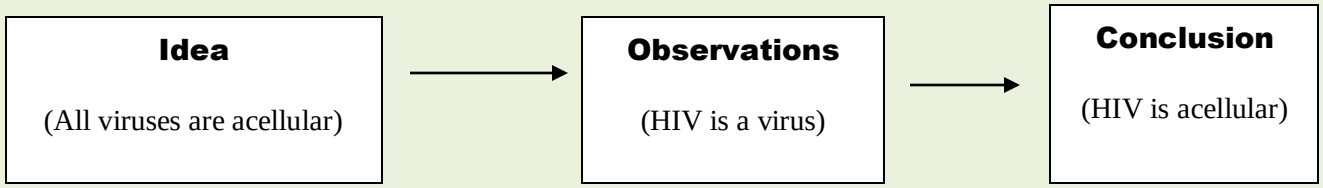
Approach to achieve AI via a model merely based on predetermined rules

Rule-Based Approach	Machine Learning
↓	↓
High Precision	Low Precision
Low Recall	High Recall

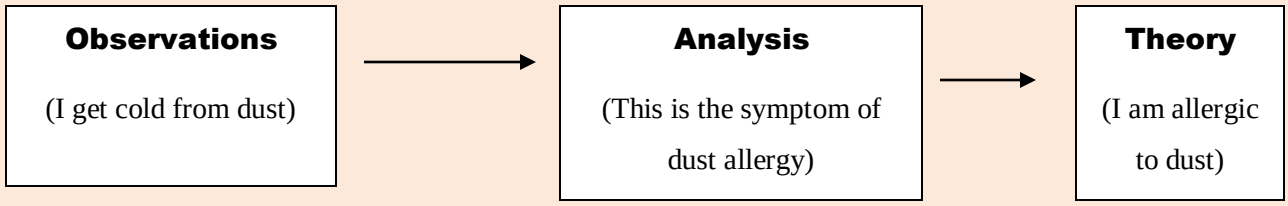
- **Precision =** $\frac{\text{Number of true positives}}{\text{Number of true positives + Number of false positives}}$

- **Recall =** $\frac{\text{Number of true positives}}{\text{Number of true positives + Number of false negatives}}$

Deduction:



Induction:



Stages of Data Mining

Data source → Data exploration → Data modeling → Model deployment

AI winter

↓

Periods of time [1974 –1980 and 1987–1993] during which AI experienced reduced funding for researches and low interest from the people

C → Consistency

All users see the same data at the same time

A → Availability

System continues to function even with node failures

P → Partition Tolerance

System continues to function even if the communication fails between nodes

CAP Theorem: A distributed database system can only support 2 out of these 3 properties: Consistency, Availability, and Partition Tolerance

- AP → Availability + Partition Tolerance
- CA → Consistency + Availability
- CP → Consistency + Partition Tolerance

Centralized Database

↓

Data is stored, located as well as maintained at a single location only

Structured data ✓→ Rules of Inference ✓→ **Conclusion**

Unstructured data ×→ Rules of Inference ×→ **Conclusion**

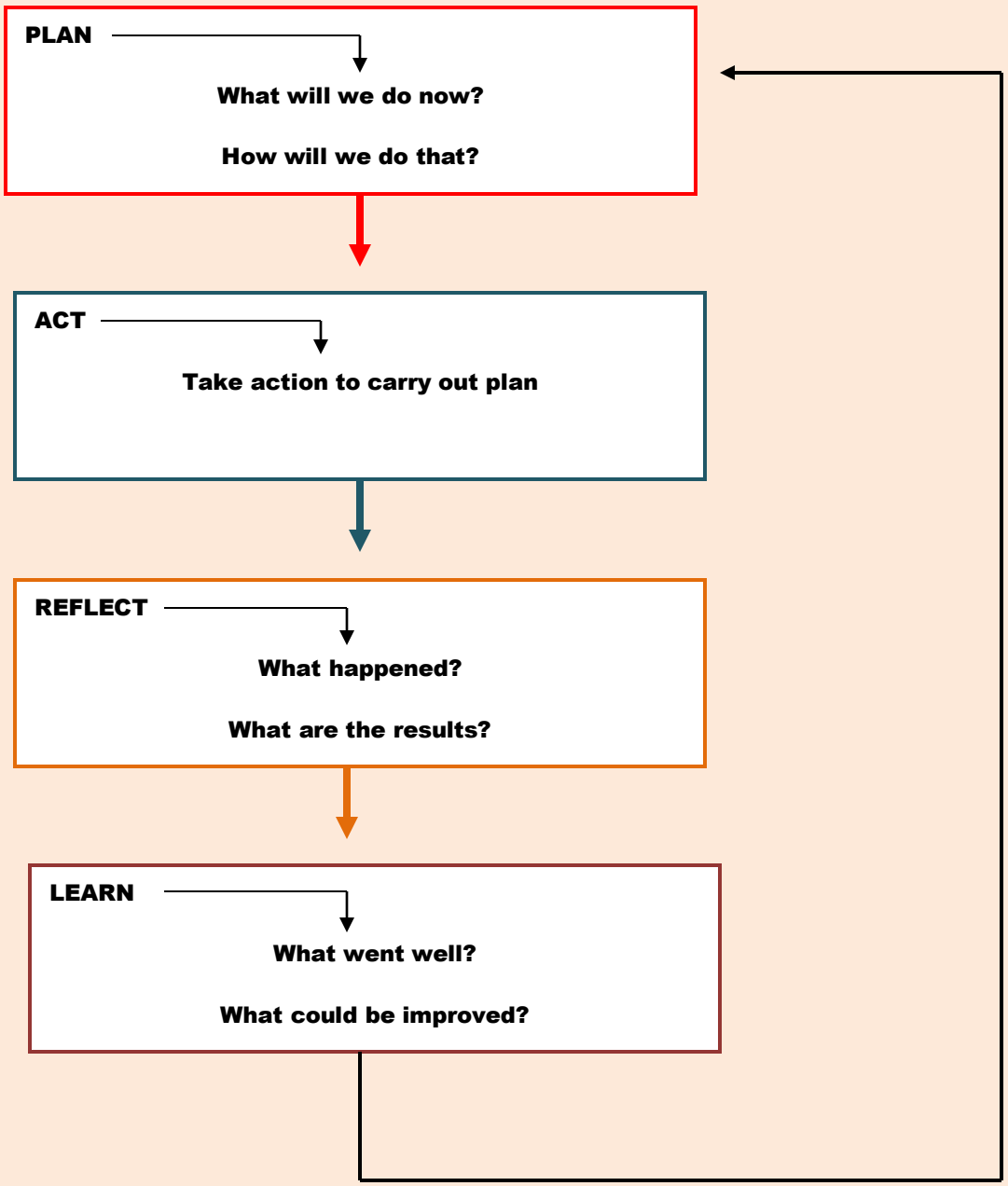
Unstructured data → Deep Learning (CV + NLP) → Structured data ✓→ Rules of Inference ✓→ Conclusion

- CV → Computer Vision
- NLP → Natural Language Processing

Social Network Analytics:

The use of network analytics to the study of social networks such as Facebook

Action Learning Cycle:



Active Experimentation

↓

Trying out what we have learned

Abstract Conceptualization

↓

Learning from the experience

- **Dynamic Data:** Data that change often
- **Static Data:** Data that does not change

Machine learning as a service

↓

Providing ML tools as a part of cloud computing services

Simpson's Paradox:

A > C

B > D

But A + B < C+ D

A huge storage of business information for the purpose of reporting and analysis

Data mart: The access layer of a **data warehouse** used to provide information to users

Binary Variable

↓

A variable that can only take one of two values

For example:

- True or False
- Yes or No
- 0 or 1
- Male or Female

PDSA

↓

Plan-Do-Study-Act

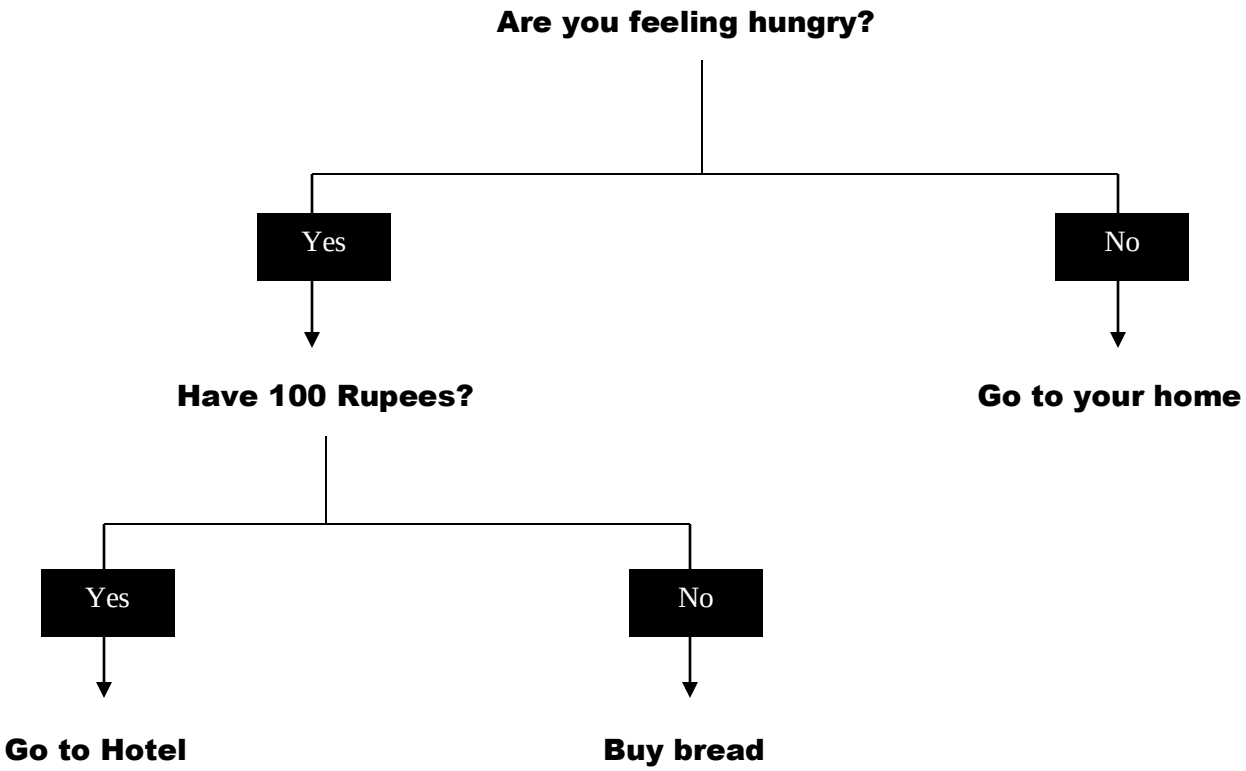
- **Plan:** What will be tested?
- **Do:** Carry out the test
- **Study:** What did you learn?
- **Act:** Take action based on what you learned

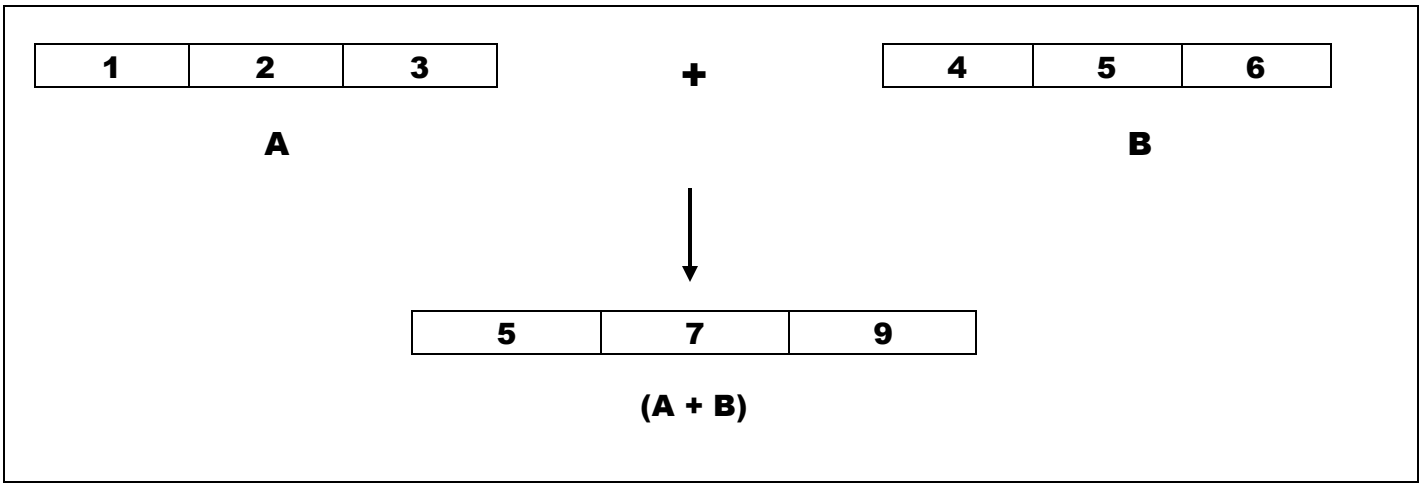
Evaluation Metrics: A set of metrics used to evaluate the quality and performance of a ML model.

Examples of **evaluation metrics** are:

- Recall = $\frac{\text{Number of true positives}}{\text{Number of true positives + Number of false negatives}}$
- Precision = $\frac{\text{Number of true positives}}{\text{Number of true positives + Number of false positives}}$
- Accuracy = $\frac{\text{Number of true positives +Number of true negatives}}{\text{Number of true positives + Number of true negatives + Number of false positives +Number of false negatives}}$
- F-measure = $\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision + Recall}}$

Decision-Tree:

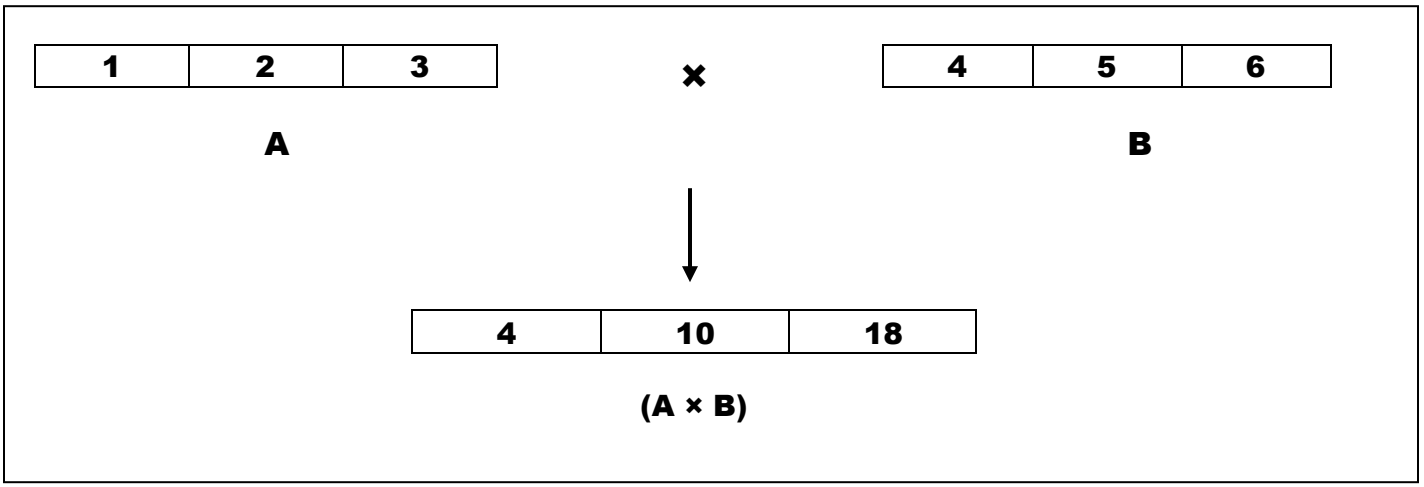





```
import numpy as np
A = np.array([1, 2, 3])
B = np.array([4, 5, 6])
print(A + B)
```

Output on the screen:

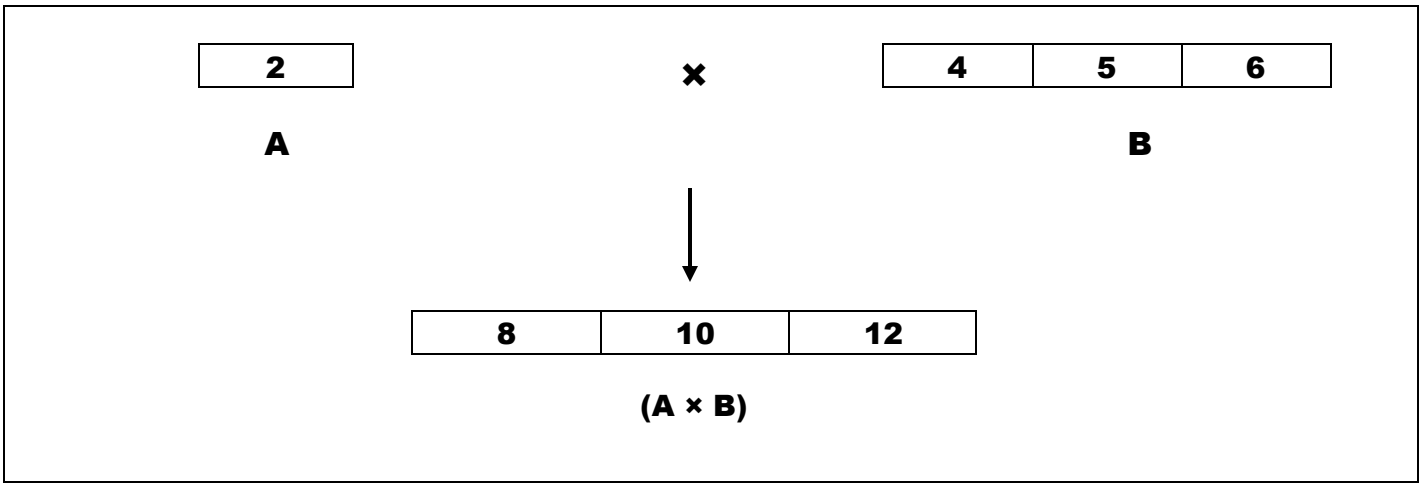
[5 7 9]




```
import numpy as np
A = np.array([1, 2, 3])
B = np.array([4, 5, 6])
print(A * B)
```

Output on the screen:

[4 10 18]




```
import numpy as np
A = np.array([2])
B = np.array([4, 5, 6])
print(A * B)
```

Output on the screen:

[8 10 12]

Code:

```
import numpy as np

A = np.array([1, 2, 3])

B = A.copy()

print(B)
```

Output on the screen:

[1 2 3]

Code:

```
import numpy as np

A = np.array([1, 2, 3])

B = A.view()

print(B)
```

Output on the screen:

[1 2 3]

Code:

```
import numpy as np

A = np.array([1, 2, 3])

A[0] = 6

B = A.view()

print(B)
```

Output on the screen:

[6 2 3]

Code:

```
import numpy as np

A = np.array([1, 2, 3])

A[1] = 6

B = A.view()

print(B)
```

Output on the screen:

[1 6 3]

Code:

```
import numpy as np

A = np.array([1, 2, 3])

A[1] = 6

B = A.copy()

print(B)
```

Output on the screen:

[1 6 3]

Code:

```
public class MyClass {

    public static void main(String[] args) {

        String a = new String("Albert");

        String b = new String("Einstein");

        System.out.println(a.equals(b));

    }

}
```

Output on the screen:

false

Code:

```
public class MyClass {

    public static void main(String[] args) {

        String a = new String("Albert");

        String b = new String("Albert");

        System.out.println(a.equals(b));

    }

}
```

Output on the screen:

true

Feature selection:

Set of all features → Selecting the best feature → ML Algorithm → Model performance

Hold-out method

Technique of splitting the data into

- **Training Dataset** → used for training the model
- **Validation Dataset** → used for validating the model
- **Testing Dataset** → used for evaluating the model

Imputation:

Name	Age
Alan	21
Stephen	
Albert	25

Incomplete data: The second row contains a missing value

Name	Age
Alan	21
Stephen	23
Albert	25

Complete data: The missing value is filled with mean of all ages

i.e. $\frac{21+25}{2} = 23$

A metric used to assess the performance of a classification model

- TP → Number of true positives
- TN → Number of true negatives
- FP → Number of false positives
- FN → Number of false negatives

Matthews Correlation Coefficient (MCC):

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

The value for MCC ranges from −1 to 1 where:

- −1 indicates total disagreement between predicted values and actual values **(inverse prediction)**
- 0 indicates average random prediction **(completely random guessing)**
- 1 indicates total agreement between predicted values and actual values **(perfect prediction)**

Fowlkes-Mallows index (FMI):

$$FMI = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP+FN}}$$

Determine the similarity between 2 clusters

A **high value** indicates a good similarity between 2 clusters

Java code	JavaScript code

public class MyClass {
  public static void main(String[] args) {
  System.out.println("Hello, World!");
  }
}
	<script>
	  document.write("Hello, World!");
	</script>

Output on the screen:

Hello, World!

Code:

```
public class MyClass {
    public static void main(String[] args) {
      // Here str[0] points to String Alan
      // Here str[1] points to String Albert
      // Here str[2] points to String Hawking
       String[] str = { "Alan", "Albert", "Hawking" };
      // Print the length of the String Albert
       System.out.println(str[1].length());
    }
}
```

Output on the screen:

6

```
x = "100"
def myfunc():
  print(x)
myfunc()
```

Global variable declaration

```
def myfunc():
  x = "100"
  print(x)
myfunc()
```

Local variable declaration

Name	Height	Weight
Alan	172	86
Albert	179	92

Wide format data

Name	Attribute	Value
Alan	Height	172
Alan	Weight	86
Albert	Height	179
Albert	Weight	92

Long format data

Confounder

A variable that falsely appears to be the cause of the outcome instead of the true cause

Decision	Null Hypothesis is true	Null hypothesis is false
Reject the Null Hypothesis	Type I error	Correct Decision
Fail to reject Null Hypothesis	Correct Decision	Type II error

Pareto principle:

80% of results are produced by 20% of causes

Examples include:

20% of convicts commit 80% of offences

20% of bus drivers cause 80% of all road accidents

Append	Extend
x = [1,2,3,4,5,6] x.append(7) print (x) [1, 2, 3, 4, 5, 6, 7]	x = [1,2,3,4,5,6] y = [7] x.extend(y) print (x) [1, 2, 3, 4, 5, 6, 7]

Python Code:

```
x = ['Stephen', 'William']
x.append('Hawking')
print (x)
```

Output on the screen:

```
['Stephen', 'William', 'Hawking']
```

Python Code:

```
x = ['Stephen', 'William']
x.extend('Hawking')
print (x)
```

Output on the screen:

```
['Stephen', 'William', 'H', 'a', 'w', 'k', 'i', 'n', 'g']
```

When a statistical model is overfed with data → Results in → **Overfitting**

Bias and variance → prediction errors of a ML model

$$\text{Bias} \propto \frac{1}{\text{variance}}$$

- An oversimplified model → High bias, Low variance
- A more complex model → Low bias, High variance

Low bias and Low variance → The goal of any supervised ML algorithm to achieve strong prediction performance

Therefore, a balance is maintained between the bias and variance and this is called the **bias-variance trade-off**

Total Sum of Squares = (Explained Sum of Squares + Residual Sum of Squares)

Python code:

```
x = lambda a,b,c : (a**b) +(c**a)

print (x (2,4,3))
```

Output on the screen:

25

Positive correlation

- High scores on A are associated with High scores on B
- Low scores on A are associated with Low scores on B

Negative correlation

- High scores on A are associated with Low scores on B
- Low scores on A are associated with High scores on B

If we want to study the diabetes among the people in **South Africa** and we study 500 people then those 500 became sample whereas the all the people in the **South African country** is the population

Machine learning algorithm → performs well on small or medium datasets and needs less time to train as compared to deep learning

Deep learning algorithm → performs well on large datasets and needs much more time to train

Python code:

```
x = [1,2,3,4]

y= [1,2,5,6]

print(list(set(x+y)))
```

Output on the screen:

```
[1, 2, 3, 4, 5, 6]
```

Python code:

```
x = [1,2,3,4]

y= [5,6,7,8]

print(list(set(x+y)))
```

Output on the screen:

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

Empirical Rule:

(68–95–99.7 Rule)

- 68% of the observations in a normal distribution lie within (mean ± standard deviation)
- 95% of the observations in a normal distribution lie within (mean ± 2 × standard deviation)
- 99.7% of the observations in a normal distribution lie within (mean ± 3 ×standard deviation)

Fuzzy Matching

↓

A technique of identifying 2 strings that are approximately similar but are not exactly the same

Confusion matrix

A table which contains information about predicted and actual values of a classification problem − used for evaluating the performance of a classification model

It has four parts namely:

- **True Positive:** The actual value is positive and the model predicts a positive value
- **True Negative:** The actual value is negative and the model predicts a negative value
- **False Positive (Type 1 error):** The actual value is negative but the model predicts a positive value
- **False Negative (Type 2 error):** The actual value is positive but the model predicts a negative value

It can be used to calculate:

- **Accuracy** = $\frac{\text{Number of true negatives} + \text{Number of true positives}}{\text{Number of true negatives} + \text{Number of true positives} + \text{Number of false negatives} + \text{Number of false positives}}$
- **Precision** = $\frac{\text{Number of true positives}}{\text{Number of true positives} + \text{Number of false positives}}$
- **Recall** = $\frac{\text{Number of true positives}}{\text{Number of true positives} + \text{Number of false negatives}}$

Python code:

```
import numpy as np

A = np.array([[2, 4, 8], [5, 7, 1]])

B = np.array([[4, 5, 6], [2, 3, 4]])

print(A * B)
```

Output on the screen:

```
[[ 8 20 48]

[10 21  4]]
```

	Bagging	Boosting	Stacking
Goal to achieve	Reduce variance	Reduce bias	Improve performance

```
# Python code to convert uppercase string to lowercase

str = 'ALBERT'
print(str.lower())
```

```
a = ["Stephen", "William", "Hawking"]
print(" ".join(a))
```

Output on the screen:

Stephen William Hawking

```
# Python code to delete 1.pdf file

import os
os.remove("1.pdf")
```

Local variables

Variables that are defined within a function

Global variables

Variables that are defined outside a function

```
# Python code to print the file path of imported modules

import os;
import socket;
print(os)
print(socket)
```

Output on the screen:

```
<module 'os' from 'C:\\Users\\Manju\\Anaconda3\\lib\\os.py'>
<module 'socket' from 'C:\\Users\\Manju\\Anaconda3\\lib\\socket.py'>
```

```
print(5//2)
print(5/2)
```

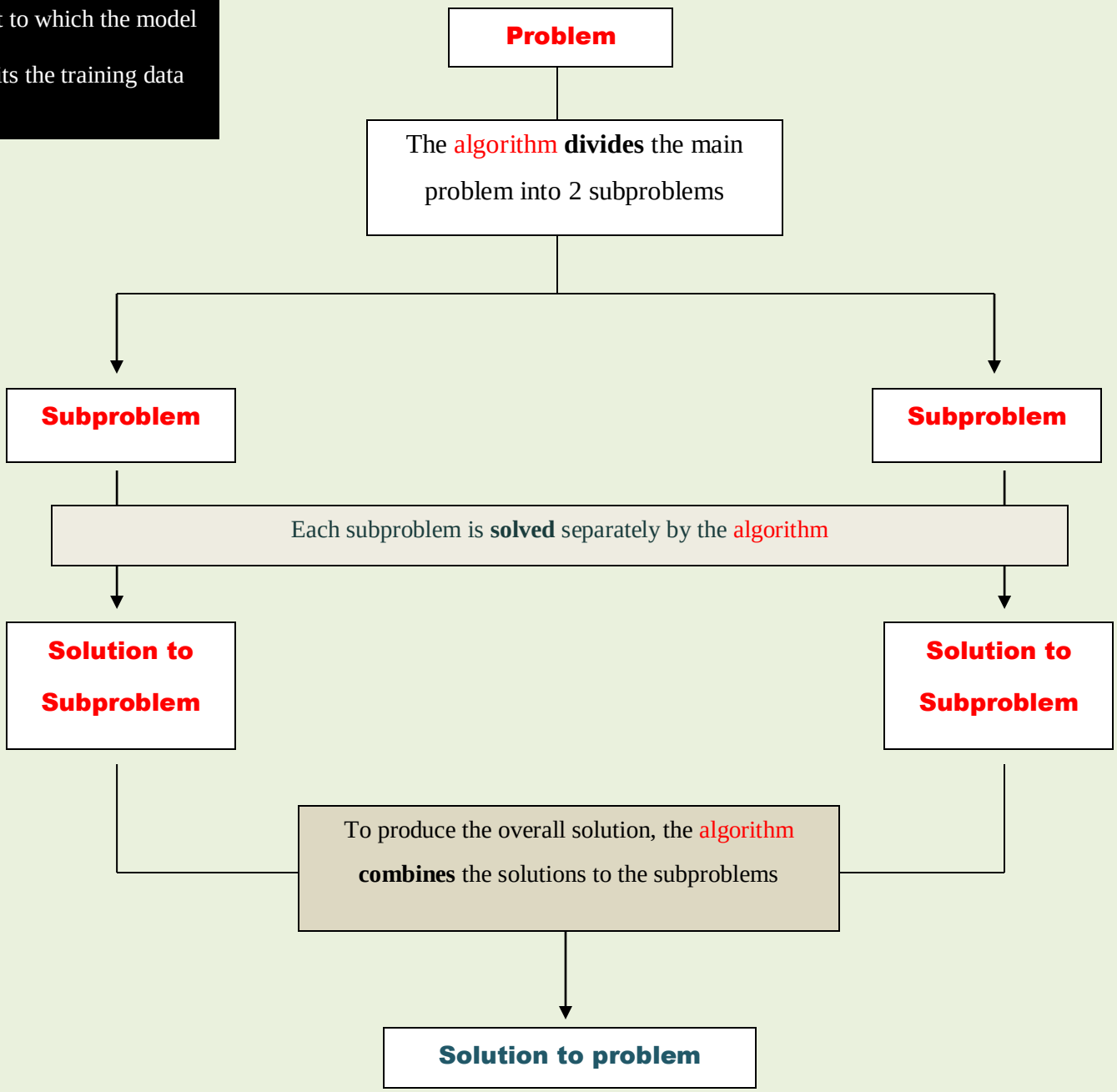
Output on the screen:

```
2
2.5
```

Degree of fit

The extent to which the model closely fits the training data

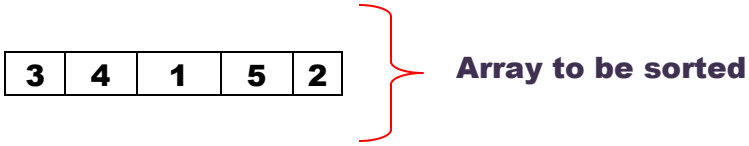
Divide and conquer technique



Chi squared = $\sum \frac{(\text{Observed} - \text{Expected})^2}{\text{Expected}}$

A metric for comparing observed results with expected results

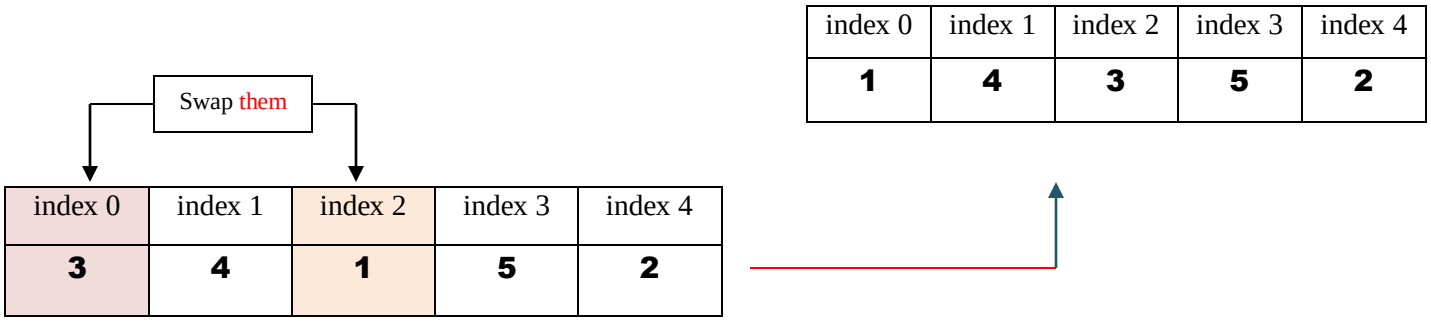
Selection Sort



index 0	index 1	index 2	index 3	index 4
3	**4**	**1**	**5**	**2**

Sorting at index 0

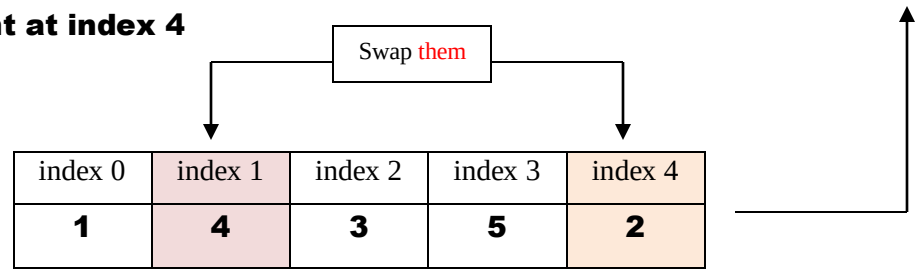
Smallest element at index 2



index 0	index 1	index 2	index 3	index 4
1	**2**	**3**	**5**	**4**

Sorting at index 1

Smallest element at index 4



Sorting at index 2

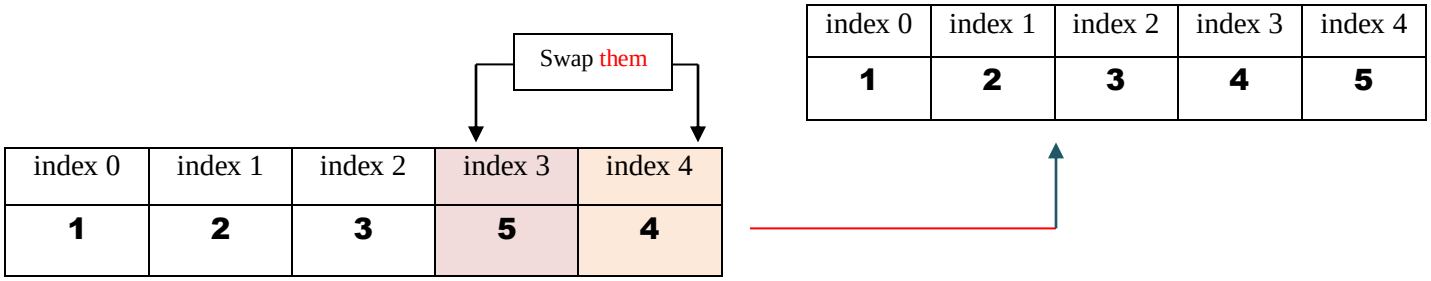
Smallest element at index 2

Leave it

index 0	index 1	index 2	index 3	index 4
1	**2**	**3**	**5**	**4**

Sorting at index 3

Smallest element at index 4



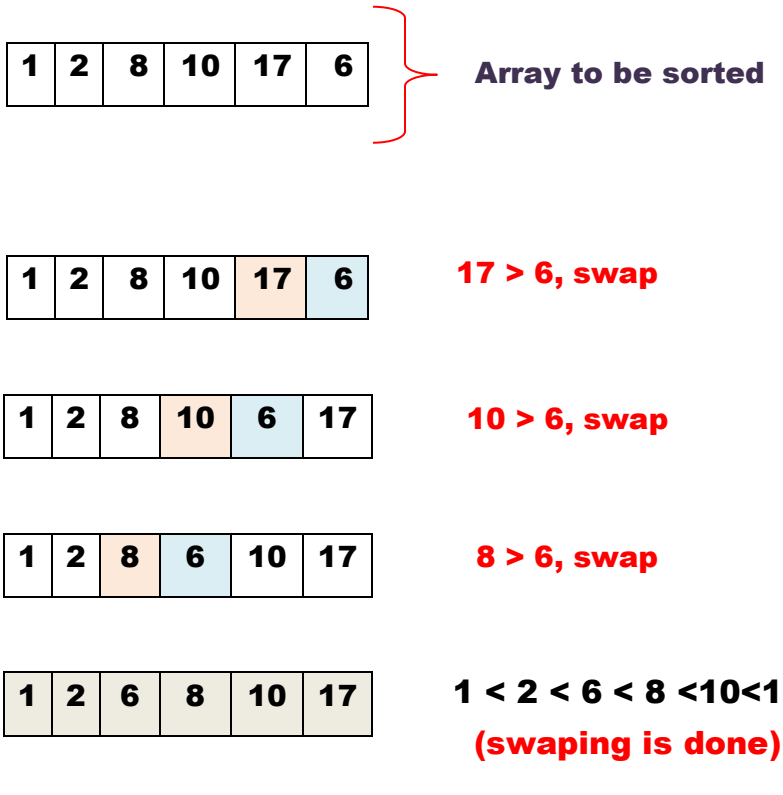
Array is now sorted

1	2	3	4	5

```
[manju@localhost ~]$ cat 1.txt
hi
Alan
Alan
Alan

[manju@localhost ~]$ sort 1.txt | uniq | tee s.txt
Alan
hi

[manju@localhost ~]$ cat s.txt
Alan
hi
```

```
find . -mtime -1 -type f -print0 | xargs -0 tar rvf "12.tar"

# Backup each file or directory that have been modified within the last 24 hour


find . -type f -size +100000 -exec ls -al {} \;

# List all the files having more than 100000 bytes
```

Insertion Sort



Array is now sorted

1	2	6	8	10	17

```
[manju@localhost ~]$ for i in 1 2 3 4 5 ; do echo "$i" ; done
1
2
3
4
5
[manju@localhost ~]$ for i in {1..5..1};do echo "$i" ; done
1
2
3
4
5
[manju@localhost ~]$ for i in {1..5..2};do echo "$i" ; done
1
3
5
[manju@localhost ~]$ for i in {1..5..3};do echo "$i" ; done
1
4
[manju@localhost ~]$ for i in {1..5..4};do echo "$i" ; done
1
5
[manju@localhost ~]$ for i in {1..5..5};do echo "$i" ; done
1
```

```
[manju@localhost ~]$ for i in `seq 1 6`;do echo $i ;done
1
2
3
4
5
6
[manju@localhost ~]$ b="x y z"; for i in $b; do echo $i; done
x
y
z
[manju@localhost ~]$ b=(x y z); for i in ${b[@]};do echo $i; done
x
y
z
```

- `while :; do echo "Hello World"; done`
- `while true; do echo "Hello World"; done`

Infinite Loop

```
[manju@localhost ~]$ cat 2.txt
Hi
Alan
John
```

```
[manju@localhost ~]$ cat 2.txt | while read LINE; do echo "${LINE}"; done
Hi
Alan
John
```

Linux command: `ss -tunlp4`

Description: **Get statistics about your network connections**

Linux command: `netstat -anpl`

Description: **Check all the network connections**

Linux command: `find . -mtime +1`

Description: **Find files modified before 1 day**

Linux command: `find . -mmin -10`

Description: **Find files modified in the last 10 minutes**

Linux command: `find . -mmin +10`

Description: **Find files modified in the before 10 minutes**

Linux command: `find . -type f -mmin -40`

Description: **Find only the files (but not directories) modified in the last 40 minutes**

Parameter	**Statistic**
↓	↓
The whole population	**A sample of the population**

Statistical significance

↓

A finding is statistically significant when it is unlikely to have occurred by chance

Alternative Data

Data gathered from non-traditional information sources

Blogs, social media and unpublished websites

Continuous Improvement Cycle

- Plan the improvement
- Implement the improvement
- Evaluate the outcome of the improvement
- If the outcome of the improvement is successful
- If the outcome of the improvement is failure

Implement it on a wider scale

```
[manju@localhost ~]$ ls
1.txt  Desktop    Downloads   empty2.txt  Pictures  Templates  test.txt  Videos
2.txt  Documents  empty1.txt  Music       Public    test.php    txt.pdf
[manju@localhost ~]$ find . -name "empty1.txt" -exec rm {} \;
[manju@localhost ~]$ ls
1.txt  Desktop    Downloads   Music     Public     test.txt   txt.pdf
2.txt  Documents  empty2.txt  Pictures  Templates  test.php  Videos
```

```
[manju@localhost ~]$ a=22; echo $a; # Prints 22
22
[manju@localhost ~]$ unset a; echo $a # Prints nothing
```

Standard score

$$\textbf{Z-Score} = \frac{\text{Observed value} - \text{Mean of the sample}}{\text{Standard deviation of the sample}}$$

If **Z-Score** = 0:

$$\text{Observed value} = \text{Mean of the sample}$$

Alpha value

The probability of making a Type I error in a statistical test

p value

The probability of the result obtained by chance

If p value > alpha value:

Result is not statistically significant

If p value < alpha value:

Result is statistically significant

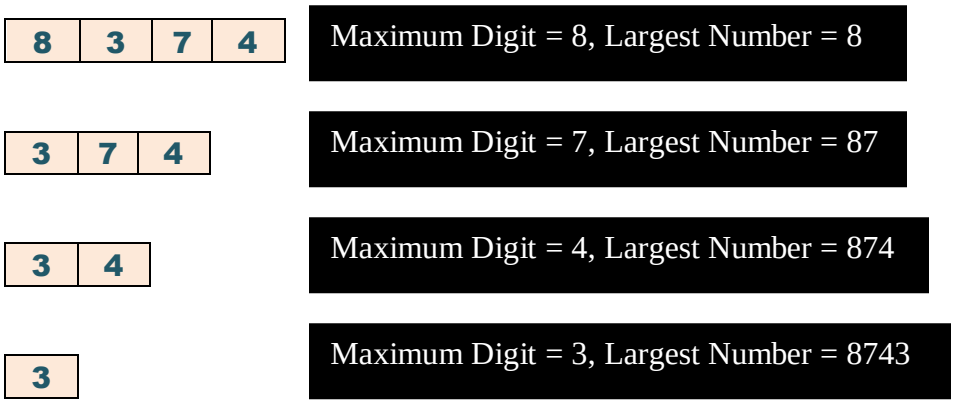
Alpha value = 0.05 → 5%

which implies:

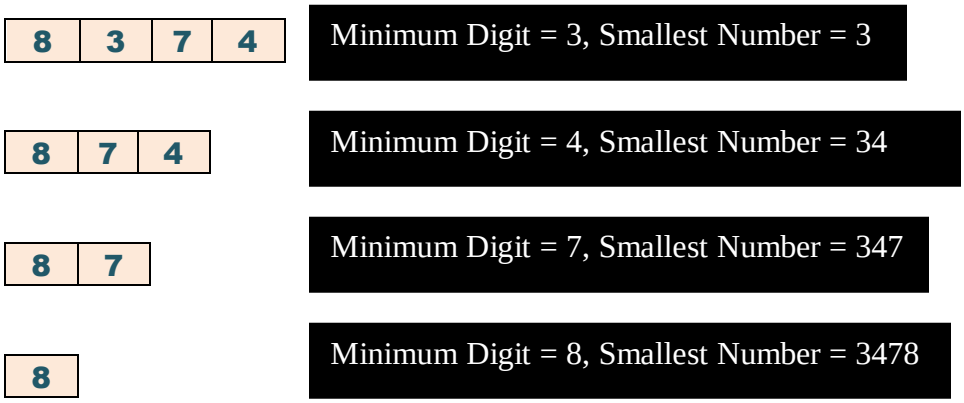
We are 95% confident that we won't make a Type I error

Greedy Algorithmic Approach:

- **Find the largest number that can be formed with the given digits:**



- **Find the smallest number that can be formed with the given digits:**



- **Largest number** → 8743 (The arrangement of the given digits in descending order)
- **Smallest number** → 3478 (The arrangement of the given digits in ascending order)

De Morgan's Laws:

not (X or Y) = not X and not Y

not (X and Y) = not X or not Y

A powerful tool to solve various kinds of Boolean algebraic expressions

Distributive Property:

X × (Y + Z) = XY + XZ

X × (Y − Z) = XY − XZ

- If Sham is in Bangalore, then he is in India
- If Sham is not in Bangalore, then he is not in India

Logically equivalent statements

If X = {1, 2} and Y = {3, 5, 7}, then the Cartesian Product of X and Y is {(1, 3), (1, 5), (1, 7), (2, 3), (2, 5), (2, 7)}

$$\textbf{Lift} = \frac{\text{The success rate given the classifier}}{\text{The normal success rate}}$$

Describe how a classifier improves prediction

$$\textbf{Positive likelihood ratio} = \frac{\text{True positive rate}}{\text{False positive rate}} = \frac{\text{Sensitivity}}{(1-\text{Specificity})}$$

$$\textbf{Negative likelihood ratio} = \frac{\text{False positive rate}}{\text{True positive rate}} = \frac{(1-\text{Sensitivity})}{\text{Specificity}}$$

Akaike information criterion:

AIC = −2ln (L) + 2k } **Does not depends on sample size**

→ Better at choosing predictive model

Schwarz Bayesian Criterion:

SBC = −2ln (L) + k ln (n) } **Depends on sample size**

→ Better at choosing explanatory model

- L = likelihood function
- k = number of parameters to be estimated
- n = number of observations (sample size)

Negation	**¬x** **(not-x)**	**If x is true, then ¬x is false**
Double Negation	**¬¬x** **(not not-x)**	**If x is true, then ¬¬x is true**
Conjunction	**x ∧ y** **(x and y)**	**x ∧ y is true only if x and y are both true**
Disjunction	**x ∨ y** **x or y (or both)**	**x ∨ y is true if any of x, y is true and also when both are true**

Exclusive Or	**$x \oplus y$ (either x or y, but not both)**	**$x \oplus y$ is true only if exactly one of x and y is true**
Biconditional	**$x \leftrightarrow y$ (x if and only if y)**	**$x \leftrightarrow y$ is true only if both x and y are true or if both x and y are false**
Conditional	**$x \rightarrow y$ (if x then y)**	**$x \rightarrow y$ is false only if x is true, and y is false; otherwise, it is always true**

- $(4^2 < 0) \rightarrow (2 < 4)$ is true because $(4^2 < 0)$ is false and $(2 < 4)$ is true
- $(2 < 4) \rightarrow (4^2 < 0)$ is false because $(2 < 4)$ is true and $(4^2 < 0)$ is false

Text: "Einstein was the greatest physicist of the twentieth century. Einstein was German"

- **Sentence 1:** Einstein was the greatest physicist of the twentieth century
- **Sentence 2:** Einstein was German

Tokenization: Breaking the text into sentences

	Einstein	was	the	greatest	physicist	of	twentieth	century	German
Einstein was the greatest physicist of the twentieth century	1	1	2	1	1	1	1	1	0
Einstein was German	1	1	0	0	0	0	0	0	1

A bag of words

The number of times each word appears in each sentence

Kolmogorov's 3 Axioms of Probability Every Data Scientist Should Know:

- **Axiom 1:**

$$0 \leq P(A) \leq 1$$

P(A) → Probability of an event A

- **Axiom 2:**

The **sample space** is the set of all possible experiment outcomes

Experiment 1: When a coin is tossed:

Possible outcomes are head or tail

sample space = {head, tail}

Experiment 2: When a dice is thrown:

Possible outcomes are the numbers 1, 2, 3, 4, 5, and 6

sample space = {1, 2, 3, 4, 5, 6}

Probability of the sample space =1

- **Axiom 3:**

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

If $P(A \cap B) = 0$:

$$P(A \cup B) = P(A) + P(B)$$

- P(A) → The probability of event A occurring
- P(B) → The probability of event B occurring
- P(A U B) → P(A or B) → The probability of either one of the events A or B occurring
- P(A ∩ B) → P(A and B) → The probability of event A and event B occurring together

R-Squared:

$$R^2 = \frac{\text{Total sum of square} - \text{Residual sum of square}}{\text{Total sum of square}}$$

Adjusted R-Squared:

$$\text{Adjusted } R^2 = 1 - \frac{(1-R^2)\,(\text{Number of samples} - 1)}{\text{Number of samples} - \text{Number of features used} - 1}$$

Clustering Technique

↓

The technique for grouping or clustering the dataset

Associations Rule Technique

↓

The technique for finding the frequently occurring items

Pre-Pruning	Post-Pruning
Stop the growth of the decision tree before it completely classifies the training set	Stop the growth of the decision tree after it completely classifies the training set

Python Pandas package — Used for ↓ Data analysis

Python Numpy package — Used for ↓ Numerical calculation

Python Plotly package — Used for ↓ Data visualization

Python Scikit-learn package ↓ **Used for** Modeling the ML Algorithms

Blue-green deployment: A methodology for deploying software upgrades that minimizes downtime by managing active traffic between **2 identical production environments**

One (blue)

Running the current version of application

Other (green)

Running the new version of application

Box diagram:

A rectangle is subdivided to illustrate:

- **if-then-else conditions**
- **repetition and case conditions**

in a control flow diagram

A **1 teraFLOPS computer system** can do one trillion (10^{12}) floating-point operations per second

Data definition language (DDL)

Modify the structure of the database

CREATE, ALTER, DROP statements

Data manipulation language (DML)

Query or modify the data of the database

SELECT, INSERT, UPDATE, DELETE statements

Bandwidth:

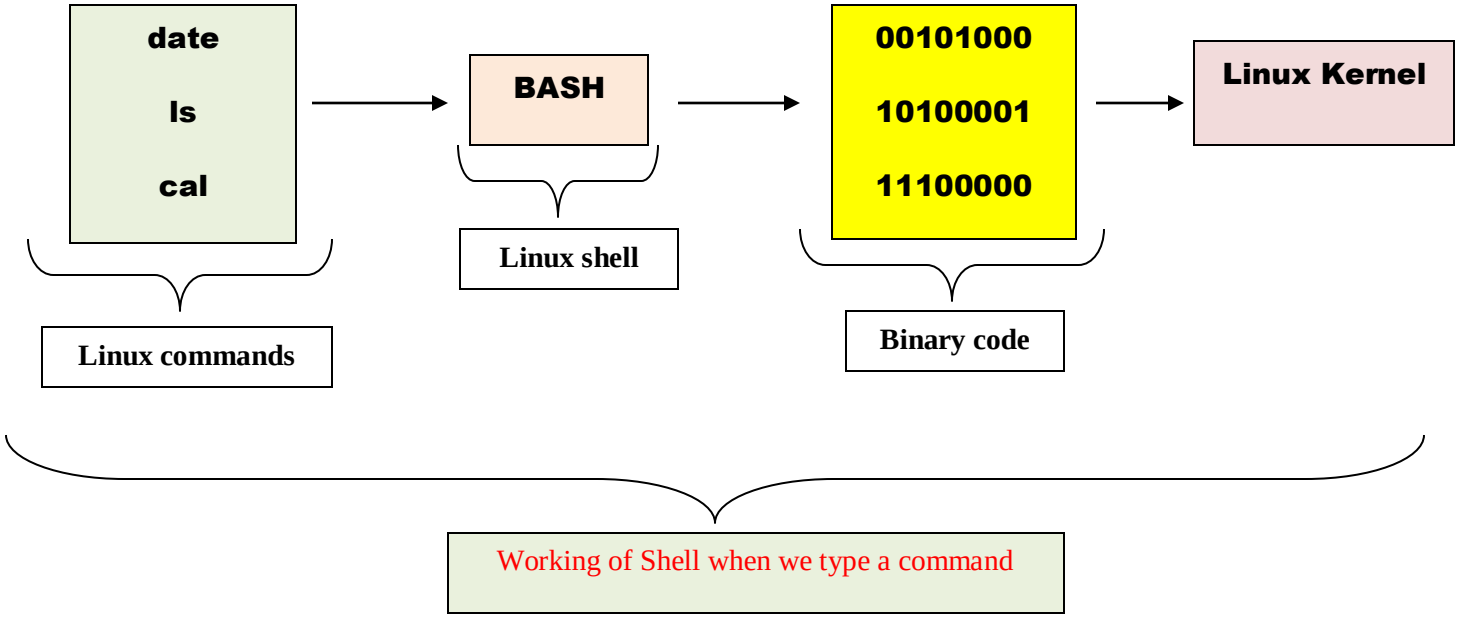
The maximum amount of information that can be sent from one point to another through an internet connection in a specific amount of time

```
[manju@localhost ~]$ for ((x=0,y=50; x<=5; x++,y-=1)); do printf "x=%03d y=%03d\n" $x $y; done
x=000 y=050
x=001 y=049
x=002 y=048
x=003 y=047
x=004 y=046
x=005 y=045

[manju@localhost ~]$ ls
1.txt    ab.cpp   ab.txt  bc.txt  CD.txt     Downloads  IO.txt  Pictures   testtxt
1.zip    abc.txt  AB.txt  BC.txt  Desktop    files.txt  log     Public     txt
abc.cpp  ab.pdf   bc.pdf  cd.txt  Documents  io.txt     Music   Templates  Videos

[manju@localhost ~]$ ls *
1.txt  abc.cpp  abc.txt  ab.txt  bc.pdf  BC.txt  CD.txt     io.txt  log      txt
1.zip  ab.cpp   ab.pdf   AB.txt  bc.txt  cd.txt  files.txt  IO.txt  testtxt

Desktop:
Documents:
Downloads:
Music:
Pictures:
Public:
Templates:
Videos:
```

```
[manju@localhost ~]$ ls .*

.bash_history  .bash_logout  .bash_profile  .bashrc  .esd_auth  .ICEauthority

.:

1.txt    ab.cpp   ab.txt  bc.txt  CD.txt     Downloads  IO.txt  Pictures   testtxt

1.zip    abc.txt  AB.txt  BC.txt  Desktop    files.txt  log     Public     txt

abc.cpp  ab.pdf   bc.pdf  cd.txt  Documents  io.txt     Music   Templates  Videos

..:

manju  users

.cache:

abrt                                                                  gstreamer-1.0

event-sound-cache.tdb.localhost.localdomain.x86_64-redhat-linux-gnu  imsettings

evolution                                                             tracker

gdm                                                                   webkit

gnome-shell


.config:

abrt       gnome-initial-setup-done  ibus        run-welcome-tour  yelp

dconf      gnome-session             imsettings  tracker

evolution  goa-1.0                   nautilus    user-dirs.dirs

gconf      gtk-3.0                   pulse       user-dirs.locale


.local:

Share


.mozilla:

extensions  plugins
```

```
[manju@localhost ~]$ ls

1.txt    ab.cpp   ab.txt  bc.txt  CD.txt     Downloads  IO.txt  Pictures   testtxt

1.zip    abc.txt  AB.txt  BC.txt  Desktop    files.txt  log     Public     txt

abc.cpp  ab.pdf   bc.pdf  cd.txt  Documents  io.txt     Music   Templates  Videos


[manju@localhost ~]$ ls a[a-z]c.txt

abc.txt


[manju@localhost ~]$ ls -l *d*.txt

-rw-r--r--. 1 root root 0 May 29 19:33 cd.txt


[manju@localhost ~]$ for i in a* b*; do md5sum $i; done

d41d8cd98f00b204e9800998ecf8427e  abc.cpp

d41d8cd98f00b204e9800998ecf8427e  ab.cpp

d41d8cd98f00b204e9800998ecf8427e  abc.txt

d41d8cd98f00b204e9800998ecf8427e  ab.pdf

d41d8cd98f00b204e9800998ecf8427e  ab.txt

d41d8cd98f00b204e9800998ecf8427e  bc.pdf

d41d8cd98f00b204e9800998ecf8427e  bc.txt


[manju@localhost ~]$ wc -l /etc/hosts || echo "The command wc -l /etc/hosts failed to read /etc/hosts"

2 /etc/hosts
```

```
[manju@localhost ~]$ names=( Albert Alan John Hawking Stephen Steve Balmer ); echo ${names[@]:0:1}

Albert

[manju@localhost ~]$ names=( Albert Alan John Hawking Stephen Steve Balmer ); echo ${names[@]:1:1}

Alan

[manju@localhost ~]$ names=( Albert Alan John Hawking Stephen Steve Balmer ); echo ${names[@]:2:1}

John
```




```
[manju@localhost ~]$ expr 2 \* 13
26
[manju@localhost ~]$ echo `expr 2 + 13`
15
[manju@localhost ~]$ echo -e "\e[1;33m This is yellow text \e[0m"
 This is yellow text
[manju@localhost ~]$ name=Albert; number=1905; echo "${name} Einstein's $number Papers"
Albert Einstein's 1905 Papers

[manju@localhost ~]$ date --date "Feb 20 2020" +%A
Thursday
```

```
find . -print
# Print the list of files and folders
```

```
[manju@localhost ~]$ touch file1.txt file2.txt FILE.txt
[manju@localhost ~]$ find . -iname "file*" -print
./file1.txt
./file2.txt
./FILE.txt
```

```
[manju@localhost ~]$ find . \( -name "*.cpp" -o -name "*.txt" \) -print
./.cache/tracker/db-version.txt
./.cache/tracker/db-locale.txt
./.cache/tracker/miner-applications-locale.txt
./.cache/tracker/last-crawl.txt
./.cache/tracker/first-index.txt
./1.txt
./ab.txt
./bc.txt
./cd.txt
./io.txt
./AB.txt
./BC.txt
./CD.txt
./IO.txt
./abc.txt
./abc.cpp
./ab.cpp
./files.txt
./file1.txt
./file2.txt
./FILE.txt
```

```
[manju@localhost ~]$ cat ab.txt
56
66
[manju@localhost ~]$ cat ab.txt | echo $[ $(tr '\n' '+' ) 0 ]
122
[manju@localhost ~]$ echo $[ 11+12+13+14+15+20 ]
85
```

```
[manju@localhost ~]$ find . -regex ".*\(\.c\|\.java\)$"
./ab.c
./ab.java

[manju@localhost ~]$ find . -iregex ".*\(\.cpp\|\.c\)$"
./abc.cpp
./ab.cpp
./ab.c
```

```
find . -type f -atime -5 -print

# Print all the files that were accessed within the last five days


find . -type f -amin +15 -print

# Print all the files that have an access time older than 15 minutes
```

```
find . -type f -name "*.cpp" -delete

# Remove all the .cpp files from the current directory


find . -type f -perm 744 -print

# Print files having permission 744
```

```
[manju@localhost ~]$ find . -type f -name "*.pdf" -exec printf "PDF FILE: %s\n" {} \;

PDF FILE: ./bc.pdf

PDF FILE: ./ab.pdf


[manju@localhost ~]$ echo "albert einstein" | tr 'a-z' 'A-Z'

ALBERT EINSTEIN


[manju@localhost ~]$ echo 45678 | tr '0-9' '9876543210'

54321


[manju@localhost ~]$ echo 54321 | tr '9876543210' '0-9'

45678


[manju@localhost ~]$ echo "Albert 1905 Papers 1921" | tr -d '0-9'

Albert  Papers
```

```
[manju@localhost ~]$ echo Albert 1905 Einstein 1921 Alan 1933 | tr -d -c '0-9 \n'

 1905  1921  1933
```

```
[manju@localhost ~]$ echo albert einstein was a German | sed 's/\w\+/[&]/g'
[albert] [einstein] [was] [a] [German]

[manju@localhost ~]$ echo albert EINSTEIN | sed 's/\([a-z]\+\) \([A-Z]\+\)/\2 \1/'
EINSTEIN albert
```

Physical Topology

The way the network is established

- how the cables are laid up
- how the computers are linked

Transmission Delay = $\frac{\text{The number of bytes in the data packet}}{\text{Network Bandwidth}}$

Transmission delay $\propto$ The number of bytes in the data packet

Transmission delay $\propto \frac{1}{\text{Network Bandwidth}}$

Propagation delay = $\frac{\text{Distance between sender and receiver}}{\text{Transmission speed}}$

Propagation delay $\propto$ Distance between sender and receiver

Propagation delay $\propto \frac{1}{\text{Transmission speed}}$

A piece of hardware that controls how data is transferred across a network

Digital Subscriber Line

Used to send digital data via a phone connection

Handshake

The process for connecting two networking devices

MAC Address

(Media Access Control Address)

The manufacturer's issued unique identifier for a piece of network hardware

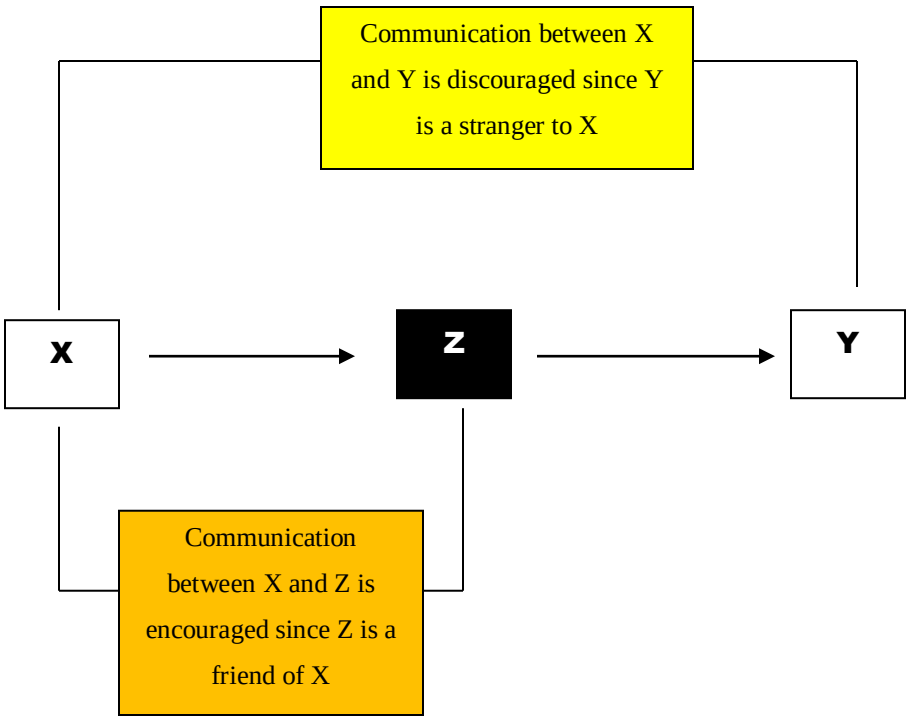
The Law of Demeter

Selection Bias

↓

A situation in which a wrong data selection can inadvertently lead to imprecise predictions

- **Z is a friend of X**
- **Y is a friend of Z but a stranger to X**



Confidence interval = sample mean ± confidence level value $\frac{\text{sample standard deviation}}{\sqrt{\text{sample size}}}$

The standard deviation of the sample determines the confidence interval. When outliers are introduced into the sample, the standard deviation increases, and the confidence interval increases with it

With the introduction of **outliers**, the confidence interval will increase

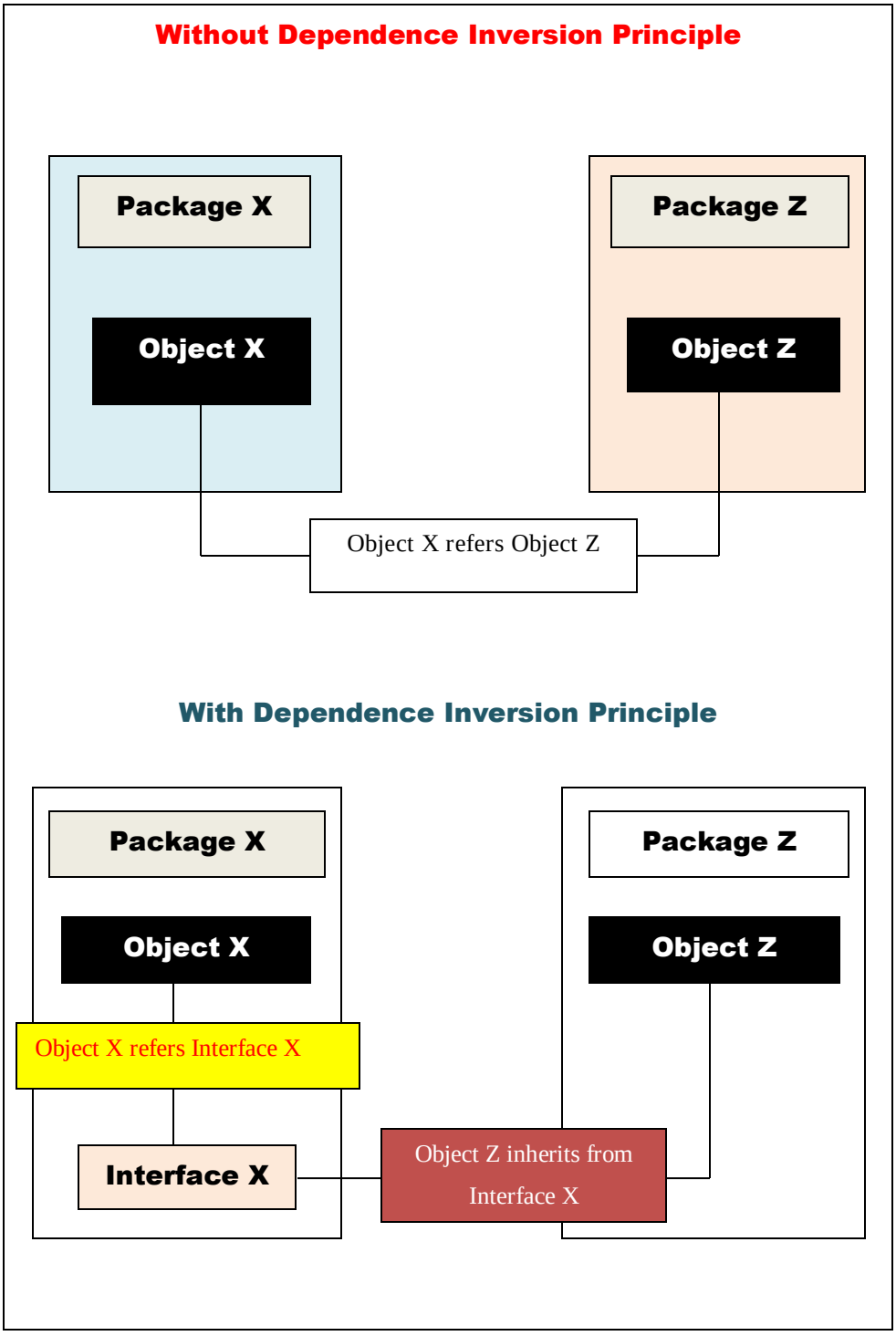
Measurement Bias:

Bias is created when the information gathered isn't totally correct (imprecise)

If a server-side web application contains multiple versions of the same function

→ Used thousands of times per day

→ **Technical debt** soon accumulates in the form of poor performance and wasted CPU time



An underweight person takes a new weight-gain powder and the weight gained is measured after 1 week

weight-gain powder

↓

Explanatory variable

weight gained

↓

Response variable

Benchmarking

↓

A method of determining the best performance − whether in a specific organization, by a competitor company or by a completely different enterprise sector

Continuous Deployment	Continuous Delivery
The deployment to the production environment is completely automated, requiring no human intervention	The deployment to the production environment is not automated, requiring manual or human intervention

Logical data warehouse:

An architectural layer that sits on top of persistent data and allows data to be viewed without having to be changed

HTTP GET	HTTP POST
HyperText Transfer Protocol method used to obtain data from a specific server via specific URL	HyperText Transfer Protocol method used to used to send data to a specific server

Spoofing

A method in which a cybercriminal attempts to steal the original identity of authorized users and then impersonates them

You are a target of phone spoofing when a caller fraudulently portrays himself as a representative of your bank and requests for your credit card details

Traits

A method that allows developers to create reusable code in PHP and other languages that don't support multiple inheritances

Cloud Elasticity	Cloud Scalability
The ability of a cloud to expand or contract its infrastructure resources in response to an unanticipated or abrupt increase in demand	The ability of a cloud to scale up or down its infrastructure resources in response to expected increase in demand

- **Software Engineering** → Practical approach required to create a software application
- **Computer Engineering** → (Practical + Theoretical) approach required to create a software application

P(6, 2): (6) + (2) ≥ 7 8 ≥ 7	True propositional statement
P(4, 1): (4) + (1) ≥ 7 5 ≱ 7	False propositional statement

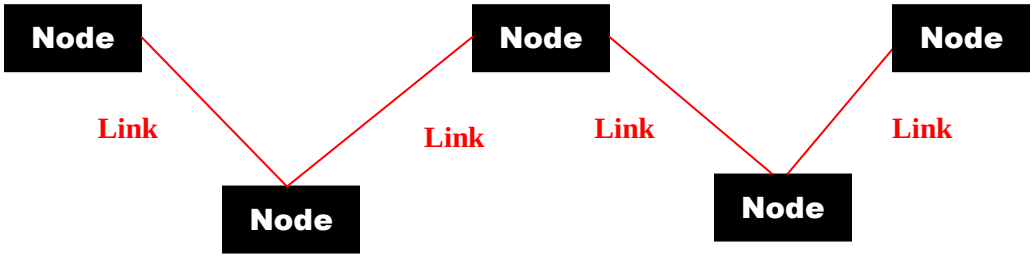
REST API Methods

GET	Get information about API resource
POST	Create API resource
PUT	Update API resource
DELETE	Delete API resource

Multitasking	**Multiprocessing**
Uses a single processor to accomplish multiple tasks at once	Uses multiple processors to accomplish multiple tasks at once
More cost-effective and less efficient than multiprocessing	Less cost-effective and more efficient than multiprocessing

Node: Any device that can communicate via network

Link or edge: The connectivity between two communicating devices in the network



In Microsoft operating systems:

ipconfig

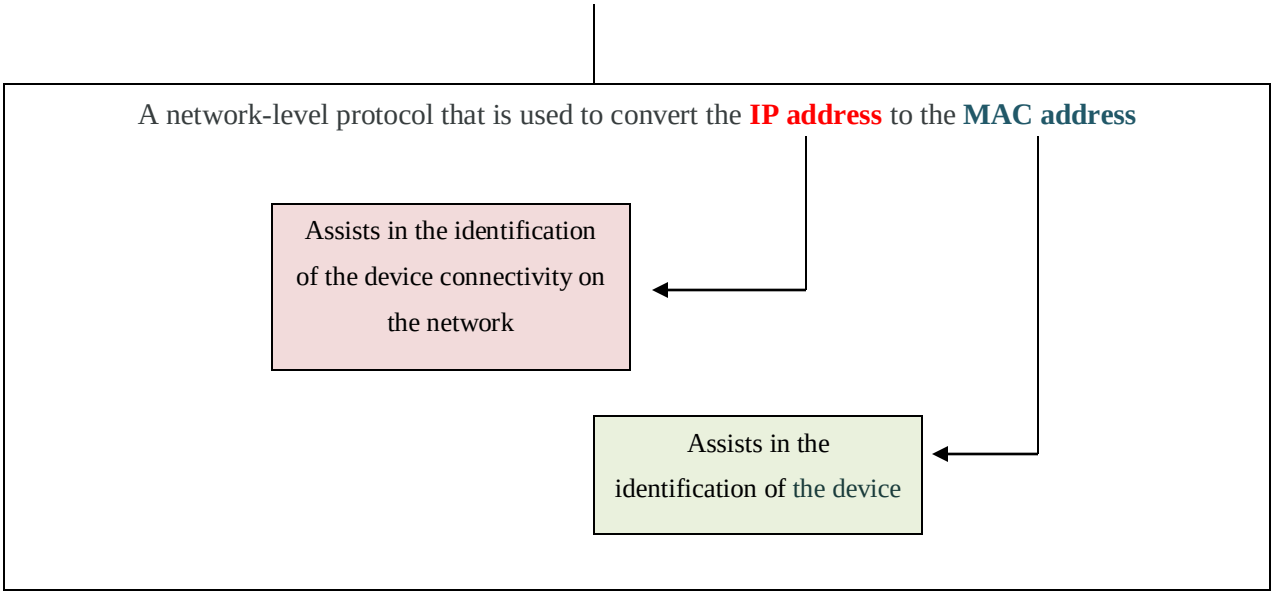
command is used to examine and configure network interfaces.

In MAC, Linux, UNIX operating systems:

ifconfig

command is used to examine and configure network interfaces.

IP address → Address Resolution Protocol → MAC address



Parametric Methods	Non-Parametric Methods
Uses a fixed number of parameters to build the model	Uses a flexible number of parameters to build the model

Epoch

The number of times the machine learning algorithm has passed through the entire training dataset

dataset size × number of epochs = number of iterations × batch size

If batch size = dataset size:

The number of epochs equals the number of iterations

Number of N-gram in a text = T – (N − 1)

- T → The total number of words in a text

text = "Albert Einstein was a famous theoretical physicist"

If N = 1:

Number of 1-gram in a text = 7 – (1 − 1) = 7

(unigram)

If N = 2:

Number of 2-gram in a text = 7 – (2 − 1) = 6

(bigram)

If N = 3:

Number of 3-gram in a text = 7 – (3 − 1) = 5

(trigram)

1-gram	2-gram	3-gram
Albert	Albert Einstein	Albert Einstein was
Einstein	Einstein was	Einstein was a
was	was a	was a famous
a	a famous	a famous theoretical
famous	famous theoretical	famous theoretical physicist
theoretical	theoretical physicist	
physicist		

- **Defect -rejection ratio** = $\frac{\text{Number of defects rejected}}{\text{Total number of defects of software}} \times 100$

- **Defect -leakage ratio** = $\frac{\text{Number of defects missed}}{\text{Total number of defects of software}} \times 100$

Fixed defects percentage = $\frac{\text{Number of defects fixed}}{\text{Total number of defects reported}} \times 100$

Passed test cases percentage = $\frac{\text{Number of passed tests}}{\text{Total number of tests}} \times 100$

Failed test cases percentage = $\frac{\text{Number of failed tests}}{\text{Total number of tests}} \times 100$

Blocked test cases percentage = $\frac{\text{Number of blocked tests}}{\text{Total number of tests}} \times 100$

3 Types of Relationship in Database Table:

One to One Relationship	When a single row in table A is linked to a single row in table B, this type of relationship is used
One to Many Relationship	When a single row in table A is linked to many rows in table B, this type of relationship is used
Many to Many Relationship	When many rows in table A is linked to many rows in table B, this type of relationship is used

Ruby program:

```
a, b, c = 42, 20, 13

puts "The value of a is #{ a }."

puts "The sum of a and b is #{ a + b }."

puts "The product of a, b and c is #{ a * b * c}."
```

Output:

The value of a is 42.

The sum of a and b is 62.

The product of a, b and c is 10920.

```
for x in 0..3

   puts "#{x}"

end
```

Output:

0

1

2

3

Shallow neural network:

{Input Layer} {Hidden Layer} {Output Layer}

Deep neural network:

{Input Layer} {Hidden Layer 1} {Hidden Layer 2} {Hidden Layer 3} … {Hidden Layer N} {Output Layer}

Deep neural networks include multiple hidden layers between the input and output layers, whereas **shallow neural networks** have only one hidden layer

Armstrong number:

- $1^3 + 5^3 + 3^3 = 153 \rightarrow$ Three Digits Armstrong number

The sum of the cubes of each digit of 153 is equal to the number itself

- $1^4 + 6^4 + 3^4 + 4^4 = 1634 \rightarrow$ Four Digits Armstrong number

The sum of the fourth power of each digit of 1634 is equal to the number itself

Python code:

```
num = 153

sum = 0

d = num

while d > 0:

   digit = d % 10

   sum += digit ** 3

   d //= 10

if num == sum:

   print(num, "is an Armstrong number")

else:

   print(num, "is not an Armstrong number")
```

Output:

153 is an Armstrong number

```
num = 1634

sum = 0

d = num

while d > 0:

   digit = d % 10

   sum += digit ** 4

   d //= 10

if num == sum:

   print(num, "is an Armstrong number")

else:

   print(num, "is not an Armstrong number")
```

Output:

1634 is an Armstrong number

IDL

(Interface Definition Language)

A language that allows a program written in one language to interact with another program written in an unknown language

World readable files

Any user can view files stored on a file system. Files stored on web servers, for example, can only be seen by Web users if their permissions are set to global accessible.

Wrapper: An object that encapsulates and transfer control to another object in order to change its interface or behavior

`print("hello"*3)`	**hellohellohello**	repetition
`print("hello"+"world")`	**helloworld**	concatenation
`print("hello"[0])`	**h**	indexing
`print("hello"[-1])`	**o**	from end
`print("hello"[1:4])`	**ell**	slicing
`print(len("hello"))`	**5**	size

Analytics Models:

Descriptive Analytics	What happened?
Diagnostic Analytics	Why did it happen?
Predictive Analytics	What will happen?
Prescriptive Analytics	How can we make it happen?

The Law of Large Numbers:

The average outcome is likely to approach the value anticipated if, by chance, the number of trails in any such random process increases

- Reduce development time
- Reduce code length
- Easy to learn and use
- Easy to understand codes
- Easy to do team projects
- **Easy to extend to other languages**

Advantages of Python

Jeff Hammerbacher's Model

- Identify problem
- Instrument data sources
- Collect data
- Prepare data:
  - Integrate
  - Transform
  - Clean
  - Filter
  - Aggregate
- Build model
- Evaluate model

$$\textbf{Signal to Noise Ratio} = \frac{\text{Mean}}{\text{Standard Deviation}}$$

Raw Score = Mean + (Standard deviation × Z score)

$$\textbf{F statistic} = \frac{\text{Explained variance}}{\text{Unexplained variance}} = \frac{\text{variance between groups}}{\text{variance within groups}}$$

$$\textbf{Standard deviation} = \sqrt{\frac{\sum (\bar{x} - x_i)^2}{(n-1)}}$$

$\bar{x}$: mean

x_i: i^{th} data value

n: number of data values

2, 4, 6 and 8

The mean of this data ($\bar{x}$) is 5

$$(5-2)^2 + (5-4)^2 + (5-6)^2 + (5-8)^2 = 20$$

$$\textbf{Standard deviation} = \sqrt{\frac{20}{(4-1)}} = \sqrt{6.66666666667} = 2.5819$$

Python code:

```
print('Albert Einstein'.partition('E')) # Output: ('Albert ', 'E', 'instein')

print('Albert Einstein'.partition('abc')) # Output: ('Albert Einstein', '', '')
```

```
print('Alan'.rjust(10))

print('Mathison'.rjust(20))

print('Turing'.ljust(10))
```

Output:

```
      Alan
             Mathison
Turing
```

```
print('Albert'.rjust(10, '/')) # Output: ////Albert

print('Albert'.ljust(10, '+')) # Output: Albert++++
```

```
print('Albert'.center(10, '*')) # Output: **Albert**
```

```
# prints the ASCII value of 'A'

print(ord("A"))

# Output: 65


# prints the ASCII value of 'C'

print(ord("C"))

# Output: 67


print(ord('A') < ord('C')) # Output: True


print((ord('A') + 5)) # Output: 70
```

```
# pip install pyperclip


import pyperclip

pyperclip.copy('Alan Turing')

print(pyperclip.paste()) # Output: Alan Turing
```

```
import re

x= re.compile(r'Chat(bot|room|man)')

print(x.search('Chatman is a surname').group()) # Output: Chatman

print(x.search('Chatman is a surname').group(1)) # Output: man
```

```
# Create an iterator 'x' from iterable 'ALAN'

x = iter('ALAN')

while True:

 try:

# To get the next item, repeatedly call next on the iterator

   print(next(x))

# When there are no more items, the iterator 'x' raises the StopIteration

 except StopIteration:

    del x # delete the iterator 'x'

    break # exit the loop
```

Output:

A

L

A

N

```
print(abs(50) + abs(-50)) # Output: 100

print(-50 + abs(-50)) # Output: 0
```

```
import time

year = time.localtime()[0]

month = time.localtime()[1]

date = time.localtime()[2]

print(year) # Output: 2022

print(month) # Output: 7

print(date) # Output: 5
```

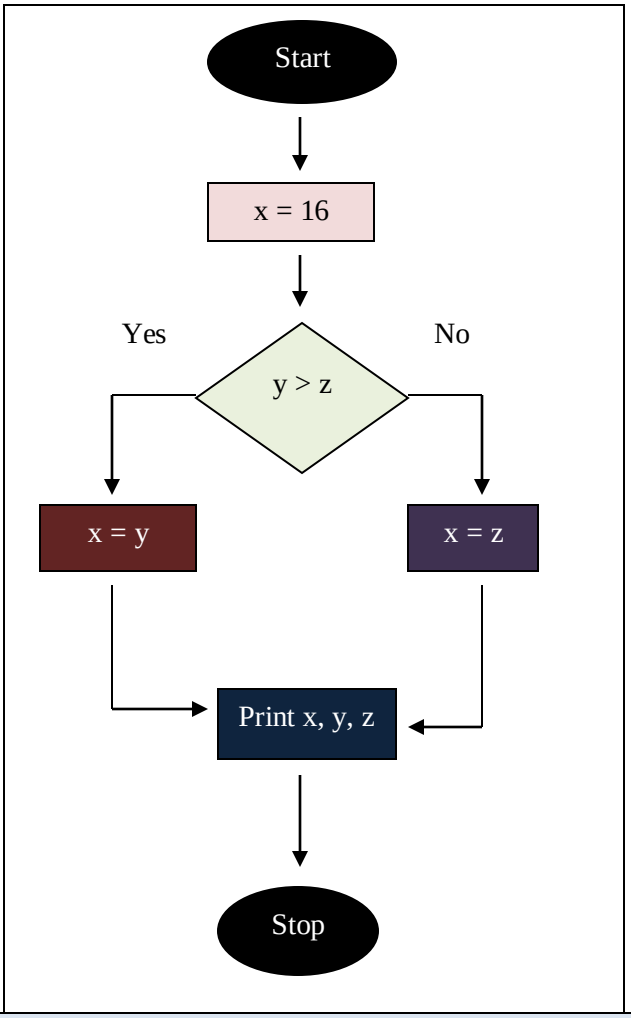
Dead code (waste of time)	Unreachable code (waste of space)

`int foo (int a, int b) {`
`  int c = a*b;`
`  return a+b;`
`}`
Executed but never used (Dead)
	`int foo (int a, int b) {`
	`  return a+b;`
	`  int c = a*b;`
	`}`
	Cannot be executed (Unreachable)

Cyclomatic Complexity = E − N + (2 × P)

- E = number of edges
- N = number of nodes
- P = number of nodes that have exit points

A software statistic that measures program complexity



The above graph shows seven shapes (nodes), seven lines (edges), hence **Cyclomatic Complexity** = 7−7 + (2×1) = 2

Dark Launch

Delivering software features that are ready for production to a small group of users in order to get their opinion and then modify the features in response

Lead Time: The length of time from the beginning of a process to its completion

Python Code:

Case 1: If there is no exception

```
try:
 print("Try")
 print(10/2)
except:
 print("Exception")
finally:
 print("Finally")
```

Output:

```
Try
5.0
Finally
```

Case 2: If there is an exception raised but handled

```
try:
 print("Try")
 print(10/0)
except ZeroDivisionError:
 print("Exception")
finally:
 print("Finally")
```

Output:

```
Try
Exception
Finally
```

Case 3: If there is an exception raised but not handled

```
try:
 print("Try")
 print(10/0)
except NameError:
 print("Exception")
finally:
 print("Finally")
```

Output:

```
print(10/0)
ZeroDivisionError: division by zero
```

```
import re

i="Alan Turing never described himself as a philosopher"

x=re.search("^Alan", i)

if x != None:

 print("The given string starts with Alan")

else:

 print("The given string does not start with Alan")


# Output: The given string starts with Alan
```

```
import re

i="Alan Turing never described himself as a philosopher"

x=re.search("philosopher$",i)

if x != None:

 print("The given string ends with philosopher")

else:

 print("The given string does not end with philosopher")


# Output: The given string ends with philosopher
```

Application Whitelisting:

The process of creating a list of trustworthy software applications and allowing only those applications to run on authorized systems

ISP Performance Monitoring:

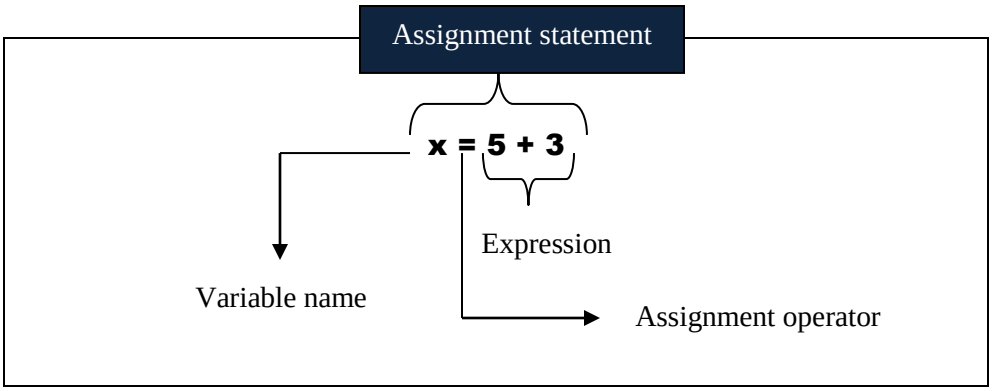
Observing and visualizing how internet services connect our webpages with our viewers and cloud systems

```
import re

i="Alan Turing never described himself as a philosopher"

x=re.search("Philosopher$", i)

if x != None:

 print("The given string ends with philosopher")

else:

 print("The given string does not end with philosopher")



# Output: The given string does not end with philosopher
```

```
import re

i="Alan Turing never described himself as a philosopher"

x=re.search("Philosopher$",i, re.IGNORECASE)

if x != None:

 print("The given string ends with philosopher")

else:

 print("The given string does not end with philosopher")



 # Output: The given string ends with philosopher
```

Incident Management:

The methodology used by the DevOps and IT Operations staff to react to a service interruption or outrage and return the service to working order




```
x='Albert'

print(x[0]) # Output: A

print(x[1]) # Output: l

print(x[2]) # Output: b
```

A	l	b	e	r	t
0	1	2	3	4	5

String Indexes

The string 'Albert' and its indexes

```
x='James'

print(x[-1]) # Output: s

print(x[-2]) # Output: e

print(x[-3]) # Output: m

print(x[-4]) # Output: a

print(x[-5]) # Output: J
```

J	a	m	e	s
-5	-4	-3	-2	-1

String Negative indexes

The string 'James' and its negative indexes

```
print('Albert\'s 1905 Papers')

# Output: Albert's 1905 Papers

print('Albert\tEinstein')

# Output: Albert  Einstein

print('Alan\\Turing')

# Output: Alan\Turing
```

```
print(len('Alan')) # Output: 4

print(len('')) # Output: 0

print(len('Alan'+'Turing')) # Output: 10

print('Alan' == 'Alan') # Output: True

print('Alan' == 'ALAN') # Output: False

print(43 == 43) # Output: True

print(43 == '43') # Output: False

print(len('Alan') + len('Alan')) # Output: 8

print(len('Alan') *2) # Output: 8
```

```
x = 'Albert Einstein grew up in Munich'

y = ''

i = len(x) - 1

while i >= 0:

   y = y + x[i]

   i = i - 1

print(y)

# Output: hcinuM ni pu werg nietsniE treblA

print(x)

# Output: Albert Einstein grew up in Munich
```

```
x = 0

while x < 2:

   while x < 4:

      x = x + 1

      print(x)

# Output:

            1

            2

            3

            4
```

```
x = 0

while x < 2:

    print('Alan')

    x = x + 1

# Output:

             Alan

             Alan

x = 0

y = 'Alan'

while x < 4:

    y = y + y[x]

    x = x + 1

print(y)

# Output: AlanAlan
```

```
i = 0

while i < len('Alan'):

      x = 'Alan'[i]

      print(x)

      i = i + 1

# Output:

             A

             l

             a

             n
```

```
if 50 > 25:

   if 12 < 14:

    print('Alan')

# Output: Alan

if 50 > 25 and 12 < 14:

   print('Alan')

# Output: Alan
```

```
if 14 != 14:
  print('Albert')
elif 50 > 25:
  print('Einstein')
# Output: Einstein


if 14 != 14 or 50 > 25:

   print('Albert')
# Output: Albert
```

```
import math

x=16

for i in range(2, int(math.sqrt(x)) + 1):

    print('Alan')
```

```
# Output:

      Alan

      Alan

      Alan
```

```
x = {'name': 'Alan', 'salary': 9900}

for i in x:

  print(i, x[i])

# Output:

                  name Alan

                  salary 9900
```

```
x = {'albert': 1905}

y = x

y['albert'] = 1915

print(x)

# Output: {'albert': 1915}
```

```
for x in range(2):
    print(x)
    print('Alan')

# Output:
                0
               Alan
                1
               Alan
```

```
for x in range(2):
    print(x)
    continue
    print('Alan')

# Output:
                0

                1
```

```
x = [['alan', 'john'], [11, 12, 13]]

print(x[0]) # Output: ['alan', 'john']

print(x[0][0]) # Output: alan

print(x[0][1]) # Output: john

print(x[1][0]) # Output: 11

print(x[1][1]) # Output: 12
```

```
x = 'Albert Einstein!'

x= x[:6] + '+' + x[7:]

print(x)

# Output: Albert+Einstein!
```

```
def myfunc(x):

   print('David ' + x)

print('Albert Einstein')

myfunc('Bowman')

print('Stephen Hawking')

myfunc('Hilbert')

print('Robert Hess')
```

```
# Output:
     Albert Einstein
      David Bowman
     Stephen Hawking
      David Hilbert
       Robert Hess
```

```
import itertools

print(itertools.product('XY', repeat=4))

# Output: <itertools.product object at 0x0000027760609750>

print(list(itertools.product('XY', repeat=2)))
# Output: [('X', 'X'), ('X', 'Y'), ('Y', 'X'), ('Y', 'Y')]

print(list(itertools.product(range(2), repeat=2)))
# Output: [(0, 0), (0, 1), (1, 0), (1, 1)]
```

```
print(31 % 6) # Output: 1

print((50 + 400) % 24) # Output: 18

print(60 % 60) # Output: 0

print(60 % 20) # Output: 0
```

```
x = "alan turing"
print(x.title())
# Output: Alan Turing
print(3 + 4*2) # Output: 11
print((3 + 4) * 2) # Output: 14

x = ['alan', 'john', 'mary', 'james']
print(x[0].title())
# Output: Alan

x = 'Alan'
print(x.upper() == 'ALAN')
# Output: True

x = {}
x['name'] = 'alan'
x['age'] = 35
print(x)
# Output: {'name': 'alan', 'age': 35}
```

```
x = {'name': 'alan'}
print("My name is: " + x['name'] + ".")
x['name'] = 'albert'
print("My name is changed to: " + x['name'] + ".")

# Output:
My name is: alan.
My name is changed to: albert.
```

```
i = '1.txt'

with open(i, 'w') as x:

 x.write("Albert Einstein\n")

 x.write("Hans Albert Einstein\n")

# Output:

            Albert Einstein

            Hans Albert Einstein
```

1.txt

```
def myfunc(x):

 print("Albert " + x.title() + "!")

myfunc('einstein')

# Output: Albert Einstein!
```

```
x = input("Please enter a number: ")

x = int(x)

if x % 2 == 0:

 print("\nThe number " + str(x) + " is even.")

else:

 print("\nThe number " + str(x) + " is odd.")


# Output:

Please enter a number: 2 # entered number

The number 2 is even.
```

```
import pandas as pd
df = pd.read_csv('1.csv')
print(df.to_string())

import pandas as pd
df = pd.read_csv('1.csv')
print(df)
```

Load the 1.csv into a DataFrame

```
print(sorted({x//2 for x in range(5)}))
# Output: [0, 1, 2]

print({x:x//2 for x in range(4)})
# Output: {0: 0, 1: 0, 2: 1, 3: 1}
```

```
i = [0, 'albert', 4]
for i[2] in 'abc': print(i)
# Output:
            [0, 'albert', 'a']
            [0, 'albert', 'b']
            [0, 'albert', 'c']
```

```
x= 'Church'

print('{place!r} is {b} km long'.format(place=x, b=len(x)))

# Output: 'Church' is 6 km long

print(f'{x!r} is {len(x)} km long')

# Output: 'Church' is 6 km long
```

```
x = {'a':24, 'b':5.89}
x['d'] = 73
print(x)
# Output: {'a': 24, 'b': 5.89, 'd': 73}


i = [11, 12, 13, 14]
print(i[1:3]) # Output: [12, 13]
print(i[1:]) # Output: [12, 13, 14]
print(i[:2]) # Output: [11, 12]


i = [11, 12, 13, 14]
i[1:3] = [52, 53, 54]
print(i) # Output: [11, 52, 53, 54, 14]
i[1:4] = [18, 19]
print(i) # Output: [11, 18, 19, 14]


i = [11, 12, 13, 14, 15]
del i[1]
print(i) # Output: [11, 13, 14, 15]
del i[::2]
print(i) # Output: [13, 15]
```

```
def myfunc(a, b):
 b.append(55)
x = 105
y = [100]
myfunc(x, y)
print(x, y)
# Output: 105 [100, 55]
```

```
x = [
'Albert Einstein'.split(),
'was a German'.split(),
'born theoretical physicist'.split(),
]
print(x, (19, 1, 5))

# Output:
 [['Albert', 'Einstein'], ['was', 'a', 'German'], ['born', 'theoretical', 'physicist']] (19, 1, 5)
```

```
a = 15; b = 17
c = 18; d = 14
if a < b or c > d:
  print('Alan')

# Output: Alan
```

```
for x in 4, 10:

   for y in 6, 7, 8, 9:

     print(f'{5.1459:{x}.{y}}')

# Output:

5.1459

5.1459

5.1459

5.1459

    5.1459

    5.1459

    5.1459

    5.1459
```

```
import re

x = re.compile(r'stand')

if x.match('understand'):

 print('matched..')

else:

 print('Failed to match..')


# Output: Failed to match..
```

```
# Contents of '2.txt '

Albert Einstein 1879

Hans Albert Einstein 1904

Elsa Einstein

Pauline Einstein
```

```
import re
with open('2.txt') as x:
if re.compile(r'\d$', re.MULTILINE).search(x.read()):
print('Some lines end in numbers.')
else:
print('No lines end in numbers.')

# Output: Some lines end in numbers.
```

Output

```
      A    B
0    alan  2
1  albert  2
2    alan  4
3  albert  4
4    alan  4
5  albert  6
6  albert  6
```

```
import pandas as pd

x = pd.DataFrame({'A': ['alan', 'albert'] * 3 + ['albert'], 'B': [2, 2, 4, 4, 4, 6, 6]})

print(x)
```

```
# pip install click

import click

def myfunc():

   click.echo("Albert Einstein")

if __name__=="__main__":

    myfunc()

# Output: Albert Einstein
```

```
import shutil

# create an archive of the directory 'Music'

# music.zip is created

shutil.make_archive('music','zip','Music')

# unzip the file 'music.zip'

shutil.unpack_archive('music.zip')

# prints a list of supported archive formats

print(shutil.get_archive_formats())

# Output:

[('bztar', "bzip2'ed tar-file"), ('gztar', "gzip'ed tar-file"),

('tar', 'uncompressed tar file'), ('xztar', "xz'ed tar-file"),

('zip', 'ZIP file')]
```

```
# pip install fire

import fire

def myfunc(x):

    return "Alan " + x

fire.Fire(myfunc("Turing"))

# Output: Alan Turing


from random import choices

x = list(range(1,5))

y = ["Albert", "Einstein"]

for i in x:

  print(f"{i} .... {choices(y)}")

# Output:

            1 .... ['Albert']

            2 .... ['Albert']

            3 .... ['Albert']

            4 .... ['Einstein']
```

```
import pandas as pd

x = pd.Series([11, 13, -14, 16], index=['w', 'y', 'x', 'z'])

print(x)

y = x.reindex(['w', 'x', 'y', 'z'])

print(y)

# Output:

                    w    11

                    y    13

                    x   -14

                    z    16

                    dtype: int64

                    w    11

                    x   -14

                    y    13

                    z    16

                    dtype: int64
```

```
# prints the size of a list
import sys
print(sys.getsizeof([11, 12, 13, 14]))
# Output: 96
print(sys.getsizeof([11, 12]))
# Output: 80
```

```
import pandas as pd

import numpy as np

x = pd.DataFrame(np.arange(16).reshape((4, 4)),

          index=['apple', 'orange', 'mango', 'grapes'],

          columns=['a', 'o', 'm', 'g'])

print(x)
```

```
          a   o   m   g
apple     0   1   2   3
orange    4   5   6   7
mango     8   9  10  11
grapes   12  13  14  15
```

```
import pandas as pd

import numpy as np

x = pd.DataFrame(np.arange(16).reshape((4, 4)),

         index=['apple', 'orange', 'mango', 'grapes'],

         columns=['a', 'o', 'm', 'g'])

print(x['o'])
```

```
apple      1
orange     5
mango      9
grapes    13
Name: o, dtype: int32
```

```
print(x[['m', 'a']])
```

	m	a
apple	2	0
orange	6	4
mango	10	8
grapes	14	12

```
print(x[:2])
```

	a	o	m	g
apple	0	1	2	3
orange	4	5	6	7

```
print(x[x['m'] > 5])
```

	a	o	m	g
orange	4	5	6	7
mango	8	9	10	11
grapes	12	13	14	15

```
# pip install PyYAML
import yaml
with open('1.yaml', 'r') as x:
   x = yaml.safe_load(x)
   print(x) # print the contents of '1.yaml'
```

```
import bs4
x = bs4.BeautifulSoup('<h1>Alan', 'html.parser')
y = bs4.BeautifulSoup('<h1>Alan', 'xml')
z = bs4.BeautifulSoup('<h1>Alan', 'lxml')
w = bs4.BeautifulSoup('<h1>Alan', 'html5lib')
print(x, x.is_xml)
# Output: <h1>Alan</h1> False
print(y, y.is_xml)
# Output: <?xml version="1.0" encoding="utf-8"?>
          <h1>Alan</h1> True
print(z, z.is_xml)
# Output: <html><body><h1>Alan</h1></body></html> False
print(w, w.is_xml)
# Output: <html><head></head><body><h1>Alan</h1></body></html> False


print(bs4.BeautifulSoup('<h1>Alan', 'html.parser').prettify())
# Output:
             <h1>
             Alan
             </h1>
```

```
import json
x = {
'name' : 'Alan',
'age' : 45,
'salary' : 54200
}
print(json.dumps(x))
# Output: {"name": "Alan", "age": 45, "salary": 54200}
print(json.dumps(True))
# Output: true
```

Java Program:

```
import java.util.Arrays;

import java.util.HashSet;

import java.util.Set;

public class MyClass {

public static void main(String[] args) {

Set i = new HashSet();

for(int x = 1; x <= 5; x++) {

i.add(x * x);

}

System.out.println(Arrays.toString(i.toArray())); // Output: [16, 1, 4, 9, 25]

System.out.println(i.contains(25)); // Output: true

}

}
```

```
import static java.lang.System.out;

public class MyClass {

public static void main(String args[]) {

 for(int i = 1; i < 5; i++) {

 out.print("I've visited ");

 out.print(i);

 out.println(" times.");

 }

 out.println("4 times! Whoopee!");

 out.println("I am excited!");

 }

}
```

Output:

```
I've visited 1 times.

I've visited 2 times.

I've visited 3 times.

I've visited 4 times.

  4 times! Whoopee!

    I am excited!
```

```
import java.util.ArrayList;

import java.util.LinkedList;

import java.util.List;

import java.util.Vector;

public class MyClass {

public static void main(String[] args) {

List<String> planets = new ArrayList<>();

planets.add("Earth");

planets.add("Uranus");

planets.add("Jupiter");

planets.add(0,"Mars");

System.out.println(planets);

List<String> stars = new LinkedList<>();

stars.add("Polaris");

stars.add(1,"Sirius");

stars.add("Betelgeuse");

System.out.println(stars);

}

}
```

```
[manju@localhost ~]$ x=15; echo '$x' = $x

$x = 15

[manju@localhost ~]$ echo albert einstein | tee -

albert einstein

albert einstein
```

Output:

```
[Mars, Earth, Uranus, Jupiter]

[Polaris, Sirius, Betelgeuse]
```

```
[manju@localhost ~]$ echo "11" >> 1.txt; echo "12" >> 1.txt; echo "13" >> 1.txt;

[manju@localhost ~]$ cat 1.txt

11

12

13

[manju@localhost ~]$ cat 1.txt | echo $[ $(tr '\n' '+' ) 0 ]

36 #11+12+13=36
```

```
import java.util.Scanner;

public class MyClass {

public static void main(String args[]) {

Scanner keyboard = new Scanner(System.in);

String marks = keyboard.next();

switch(marks) {

case "85":

 System.out.println("Distinction");

 break;

case "35":

 System.out.println("Pass");

 break;

case "25":

 System.out.println("Fail");

 break;

}

}

}
```

```
[manju@localhost ~]$ echo $[ 11+12+13 ]

36
```

Output:

```
35 # entered number

 Pass
```

```
[manju@localhost ~]$ touch 1.py 2.py 3.py 1.sh

[manju@localhost ~]$ find . -iregex ".*\(\.py\|\.sh\)$"

./1.sh

./1.py

./2.py

./3.py
```

```
import java.util.Scanner;

import static java.lang.System.out;

import static java.lang.System.in;

public class MyClass {

public static void main(String args[]) {

Scanner keyboard = new Scanner(in);

String x = "john";

 out.print("Enter your password: ");

String i = keyboard.next();

 out.print("You entered: ");

 out.println(i);

 out.print("But the password is: ");

 out.println(x);

 if (x == i) {

 out.println("You are Permitted!");

 } else {

 out.println("Sorry, Try Again!");

 }

 keyboard.close();

 }

}
```

Output:

```
Enter your password: alan # entered password

You entered: alan

But the password is: john

Sorry, Try Again!
```

```
find . -type f -size +5M

# Find all files having size > 5 megabytes

find . -type f -size -5M

# Find all files having size < 5 megabytes

find . -type f -size 5M

# Find all files having size = 5 megabytes

find . -type f -size +5G

# Find all files having size > 5 gigabytes

find . -type f -size -5G

# Find all files having size < 5 gigabytes

find . -type f -size 5G

# Find all files having size = 5 gigabytes
```

```
find . -type f -name "*.txt" -delete

# Remove all the .txt files from the current directory

find . -type f -user manju -print

# Print the list of all files owned by the user manju

find . -type f -perm 644 -print

# Print files having permission 644

find . -type f -name "*.pdf" ! -perm 644 -print

# Print pdf files having permission 644
```

```
find /usr -type f -size +4G -exec ls -lh {} \; | awk '{ print $9 "|| Size : " $5 }'

# Print the file paths which are greater than 4GB


find . -type d -print

# List only directories


find . -type f -print

# List only regular files


find . -type l -print

# List only symbolic links
```

```
import java.util.Random;

public class MyClass {

public static void main(String args[]) {

 Random x = new Random();

 int i = x.nextInt(5) + 1;

 if (i == 2 || i == 5 ) {

 System.out.println("Albert Einstein");

 }

 System.out.println(i);

 }

}
```

If Random number = 2 or 5

Output:

Albert Einstein

2 or 5

```
import java.io.File;

import java.io.FileNotFoundException;

import java.io.PrintStream;

public class MyClass  {

public static void main(String args[]) throws FileNotFoundException {

 PrintStream diskWriter = new PrintStream(new File("1.txt"));

 diskWriter.print ('H');

 diskWriter.println('i');

 diskWriter.close();

 }

}
```

- Deletes existing 1.txt file on the hard drive.
- Creates an empty file named 1.txt.
- The print method call inserts the letter H into the 1.txt file.
- The println method call adds the letter i into the 1.txt file.
- So, after running the code, the 1.txt file contains 2 letters: the letters "H" "i".

```java
import java.util.Scanner;

public class MyClass  {

public static void main(String args[]) {

Scanner keyboard = new Scanner(System.in);

char x, y, z, w;

System.out.print("Enter a string: ");

 x = keyboard.findWithinHorizon(".", 0).charAt(0);

 y = keyboard.findWithinHorizon(".", 0).charAt(0);

 z = keyboard.findWithinHorizon(".", 0).charAt(0);

 w = keyboard.findWithinHorizon(".", 0).charAt(0);

 System.out.print(w);

 System.out.print(z);

 System.out.print(y);

 System.out.print(x);

 System.out.println();

 keyboard.close();

 }

}
```

Output:

```
Enter a string: alan # entered string
nala
```

```
cat /proc/partitions
```

Lists out the partitions information available on the system

```
[manju@localhost ~]$ echo "Albert Einstein." | tr 'a-zA-Z' 'n-za-mN-ZA-M'
Nyoreg Rvafgrva.
[manju@localhost ~]$ echo Nyoreg Rvafgrva. | tr 'a-zA-Z' 'n-za-mN-ZA-M'
Albert Einstein.
[manju@localhost ~]$ echo "Albert        Einstein  ....." | tr -s ' '
Albert Einstein .....
```

```
for i in {1..5}.txt
do
 touch $i
done
```

Creates empty files: 1.txt, 2.txt, 3.txt, 4.txt and 5.txt

```
[manju@localhost ~]$ echo -e "Alan Mathison \nTuring" | grep Mathison
Alan Mathison

[manju@localhost ~]$ echo Alan Mathison Turing. | egrep -o "[a-z]+\."
uring.

[manju@localhost ~]$ echo Alan Mathison Turing | grep -e "Alan" -e "Turing" -o
Alan
Turing
```

```
[manju@localhost ~]$ echo -e "Math\nison" | awk 'BEGIN{ print "Alan" } { print } END { print "Turing" }'

Alan

Math

ison

Turing
```

```
[manju@localhost ~]$ seq 4 | awk 'BEGIN { getline; print"****", $0 } {print $0 }'

**** 1

2

3

4
```

```
f r-- = 4  # Read permission

f -w- = 2  # Write permission

f --x = 1  # Execute permission

f rw- = 4 + 2 = 6  # Read + write  permission

f r-x = 4 + 1 = 5  # Read + execute permission
```

```
f rwx = 4 + 2 + 1 = 7

f rw- = 4 + 2 = 6

f r-- = 4

rwx rw- r-- is equal to 764


# setting the permission '764' to the file "myfiles.txt"

chmod 764 myfiles.txt
```

C Program:

```c
#include <stdio.h>

#include <math.h>

int main() {

double x = 6 * atan(2.0);

printf("%.0f %.1f %.2f %.3f %.6f %.10f\n", x, x, x, x, x, x);

printf("%.0e %.1e %.2e %.10e\n", x, x, x, x);

return 0;

}
```

Output:

```
7 6.6 6.64 6.643 6.642892 6.6428923068

7e+00 6.6e+00 6.64e+00 6.6428923068e+00
```

```c
#include <stdio.h>

int main() {

printf("%.1g %.2g %.4g %.8g\n", 10/3.0, 10/3.0, 10/3.0, 10/3.0);

return 0;

}


// Output: 3 3.3 3.333 3.3333333
```

Python code:

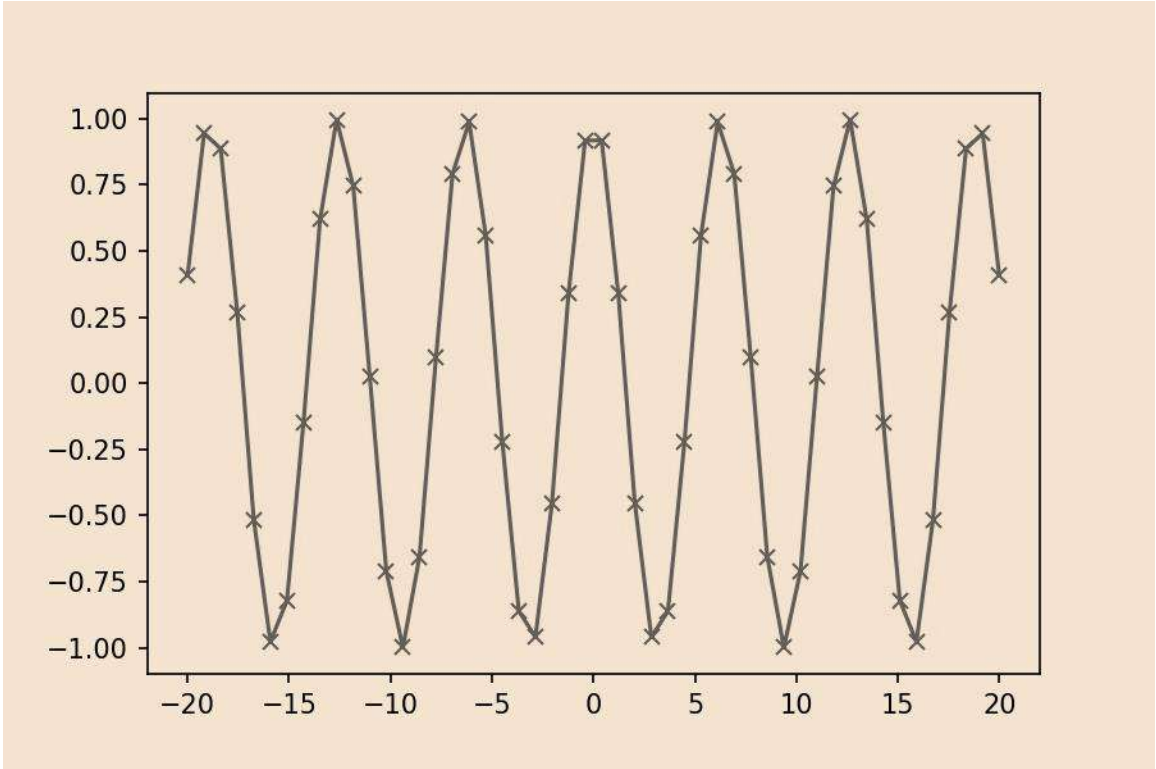
```python
import numpy as np

i = np.array([[11, 12, 13], [14, 15, 16]])

print("i:\n{}".format(i))
```

Output:

```
i:

[[11 12 13]

 [14 15 16]]
```

```
import matplotlib.pyplot as plt

import numpy as np

# Generate a sequence of numbers from -20 to 20 with 50 steps in between

a = np.linspace(-20, 20, 50)

# Create a second array using cosine

b = np.cos(a)

# The plot function makes a line chart of one array against another

plt.plot(a, b, marker="x")
```

Output:



Information Architecture:

The technique of logically and clearly arranging complex information

```
import pandas as pd

# create a simple dataset of students

x = {'Name': ["James", "Mary", "Albert", "Alan"],

 'Subject' : ["Physics", "Chemistry", "Maths", "Biology"],

 'Marks' : [84, 53, 73, 93]

 }

print(pd.DataFrame(x))


// Output:


     Name    Subject  Marks

0   James    Physics     84

1    Mary   Chemistry    53

2   Albert    Maths      73

3    Alan    Biology     93
```

Minification:

The practice of deleting unneeded or redundant information without altering how the content is handled by the browser, such as formatting and code comments, getting rid of useless code, utilizing shorter variable and function names, etc.

Mobile-first

A design approach that prioritizes the smallest screens and starts the design process with mobile devices in consideration in order to improve customer experiences

Accessibility Conformance Testing

Define and record guidelines for determining whether web content complies with accessibility standards like the **Web Content Accessibility Guidelines** (WCAG)

```
import sys

import pandas as pd

import matplotlib

import numpy as np

import scipy as sp

import IPython

import sklearn


print("Python version: {}".format(sys.version))

# Output: Python version: 3.7.3 (default, Mar 27 2019, 17:13:21) [MSC v.1915 64 bit (AMD64)]


print("Pandas version: {}".format(pd.__version__))

# Output: Pandas version: 0.24.2


print("Matplotlib version: {}".format(matplotlib.__version__))

# Output: Matplotlib version: 3.0.3


print("NumPy version: {}".format(np.__version__))

# Output: NumPy version: 1.16.2


print("SciPy version: {}".format(sp.__version__))

# Output: SciPy version: 1.2.1


print("IPython version: {}".format(IPython.__version__))

# Output: IPython version: 7.4.0


print("Scikit-learn version: {}".format(sklearn.__version__))

# Output: Scikit-learn version: 0.20.3
```

Accessibility Object Model:

An accessibility tree for an HTML page can be altered and eventually examined by programmers using a JavaScript API.

Agglomerative Clustering:

The most popular approach of hierarchical clustering that groups objects into clusters according on how similar they are

```
import numpy as np

print("{}".format(np.linspace(-2, 2, 11)))

# Output: [-2.  -1.6 -1.2 -0.8 -0.4  0.   0.4  0.8  1.2  1.6  2. ]
```

numpy.linspace(start, stop, num=11)

```
np.linspace(-2, 2, 11)
```

1	2	3	4	5	6	7	8	9	10	11
-2	-1.6	-1.2	-0.8	-0.4	0.	0.4	0.8	1.2	1.6	2.

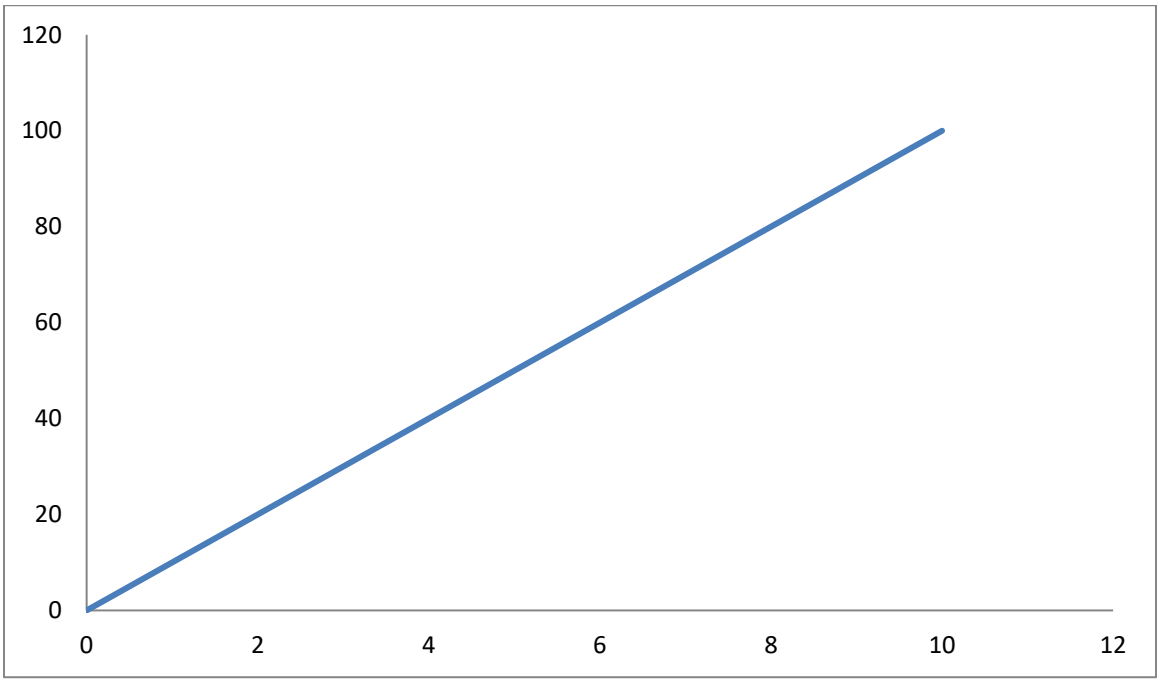
American Standard Code for Information Interchange:

(ASCII)

A standard collection of codes used consistently to represent characters. There are 128 letters, digits, punctuation, and symbols over all. A distinct binary string has been assigned to each character.

```
import matplotlib.pyplot as plt

import numpy as np


x = np.array([0, 10])

y = np.array([0, 100])

# Draw a line in a diagram from position (0,0) to position (10,100)

plt.plot(x, y)
```

Output:

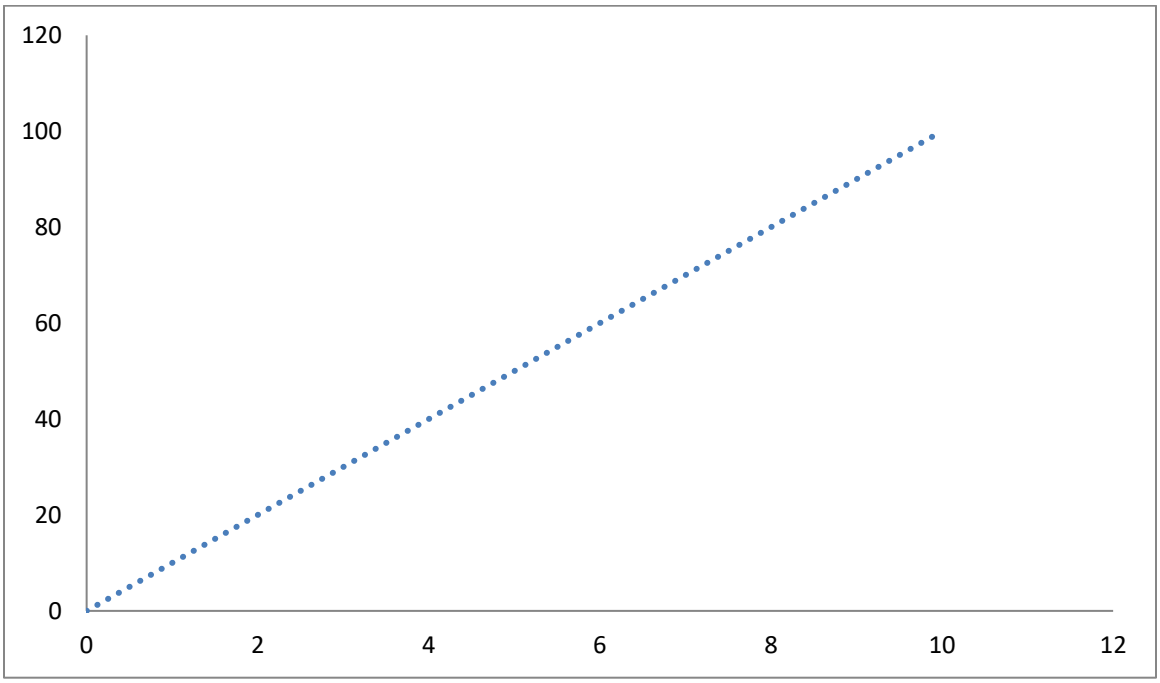


Concurrent Versions System

A tool that allows programmers to save and retrieve various development versions of source code

```
import matplotlib.pyplot as plt

import numpy as np



x = np.array([0, 10])

y = np.array([0, 100])

# Draw a dotted line in a diagram from position (0,0) to position (10,100)

plt.plot(x, y, linestyle = 'dotted')
```

Output:

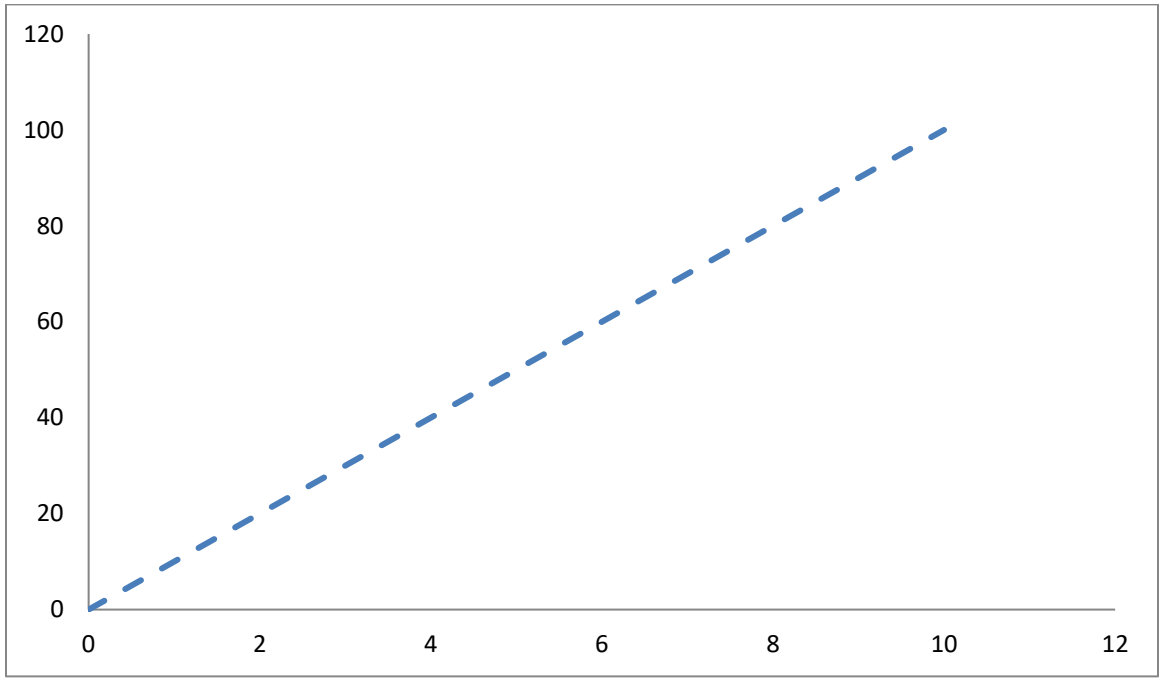


Cone of Uncertainty

A visual representation of the rising accuracy of predictions as more information about a project becomes available over time

```
import matplotlib.pyplot as plt

import numpy as np


x = np.array([0, 10])

y = np.array([0, 100])

# Draw a dashed line in a diagram from position (0,0) to position (10,100)

plt.plot(x, y, linestyle = 'dashed')
```

Output:

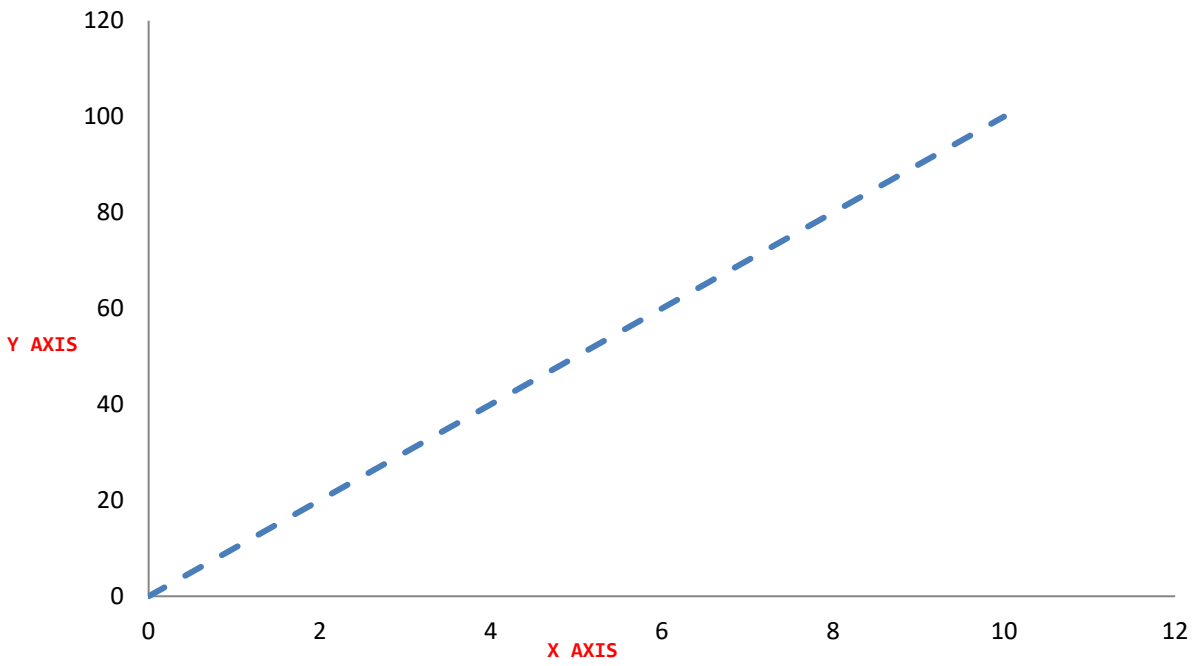


Convention over Configuration:

A method of writing computer programs that adheres to accepted programming rules rather than the configurations that the programmer specifies. It enables quick and easy software production while maintaining fundamental software requirements

```
import matplotlib.pyplot as plt

import numpy as np


x = np.array([0, 10])

y = np.array([0, 100])

# Draw a dashed line in a diagram from position (0,0) to position (10,100)

plt.plot(x, y, linestyle = 'dashed')

plt.xlabel("X AXIS")

plt.ylabel("Y AXIS")
```

Output:



Duck Typing

A property of some modern computer languages that allows for dynamic data types

Automatic Semicolon Insertion:

JavaScript does not need the use of a semicolon at the end of a statement, in contrast to other C-like languages. Instead, the semicolon is optional, and when the JavaScript interpreter runs the code, it will "sensibly" add them

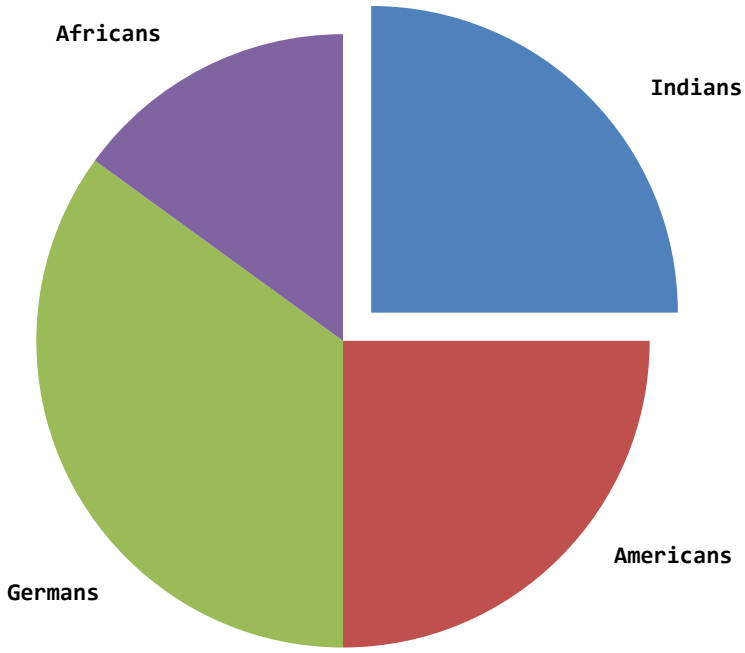
Bad Neighborhood

A collection of websites that are connected and have bad reputations that search engines have severely devalued

Command-Query Separation: A design approach that specifies a method can only ever be either a COMMAND that executes an operation OR a QUERY that gives information to the caller

```
import matplotlib.pyplot as plt

import numpy as np


x = np.array([25, 25, 35, 15])

y = ["Indians", "Africans", "Germans", "Americans"]

z = [0.6, 0, 0, 0]

# Pull the "Indians" wedge 0.6 from the center of the pie

plt.pie(x, labels = y, explode = z)
```

Output:



The law of conservation of complexity:

(Tesler's Law or Waterbed Theory)

A postulate in human-computer interaction that asserts that each application has a certain level of inherent complexity that can be neither reduced nor hidden

First Contentful Paint:

(FCP)

The period of time it takes for a page to load completely − from the time the request is received until any content appears on the screen. The content loads more slowly the higher the FCP score

IndexedDB:

Maintaining a NoSQL database of JSON objects with a JavaScript application programming interface (API) offered by web browsers

Self-supervised learning:

A new area of study that lies in between supervised learning and unsupervised learning and uses automatically labelled data to eliminate the need for human annotation. It is a method of machine learning where a **ML model** teaches itself to learn one feature of the input from another feature of the input

Sentiment analysis:

A method based on natural language processing that aids in identifying human emotions (positive, negative, neutral, etc.) in a document

Sequence to sequence:

The potential to train a model to translate input sequences from one domain (for example, an English sentence) to output sequences in another domain (for example a sentence in Hindi)

```
[manju@localhost ~]$ cat names.txt
Albert
Alan
John
James
Mary
Yovan
Hawking
Stephen
```

```
[manju@localhost ~]$ awk 'length($1) > 6 {print $0}' names.txt
Hawking
Stephen

[manju@localhost ~]$ awk 'length($1) == 4' names.txt
Alan
John
Mary

[manju@localhost ~]$ awk '!(length($1) == 4)' names.txt
Albert
James
Yovan
Hawking
Stephen

[manju@localhost ~]$ awk '(length($1) == 3) || (length($1) == 4)' names.txt
Alan
John
Mary
```

Print all names that are longer than 6 characters

Print all names that have 4 characters

Print all names that do not have 4 characters

Shape of a tensor:

The number of elements in each dimension of the tensor

Siamese neural network: A group of neural network topologies that have two or more subnetworks that are exactly the same

```
[manju@localhost ~]$ awk 'length($1) > 0  {print $1, "has", length($1), "chars"}' names.txt
Albert has 6 chars
Alan has 4 chars
John has 4 chars
James has 5 chars
Mary has 4 chars
Yovan has 5 chars
Hawking has 7 chars
Stephen has 7 chars
```

Silhouette Coefficient:

A statistic used to determine the effectiveness of a clustering method. Its value is between -1 and 1

- 1 → clusters are clearly distinguished and spaced far apart
- 0 → clusters are undifferentiated or that there is no statistically significant difference across clusters
- -1 → clusters are assigned incorrectly

Similarity learning:

A domain of **supervised machine learning** where the objective is to develop a similarity function that assesses the degree of resemblance or relationship between two things and returns a similarity score. When the things are similar, the similarity score is higher; when they are distinct, the similarity score is lower

$$\textbf{Sparsity} = \frac{\text{The number of zero elements in a vector}}{\text{The total number of elements in the vector}}$$

A Sparsity of 18% means that 18% of elements in the vector contain zeros

Spatial-temporal reasoning:

A branch of AI where the ultimate goal is to create a system with the ability to manipulate objects, navigate through time and space, and solve issues

Stacking:

An ensemble machine learning technique that combines predictions from various effective machine learning models in the most effective way possible

Strong classifier:

The classifier that is responsible for combining the findings from the weak classifiers into a single appropriate result

Structural similarity:

The degree to which two or more objects' shapes overlap; the more overlap there is, the more similar the objects are

Subsampling:

The outcome of 2 samplings, the first of which was taken from the population and the second from the sample

Synthetic feature:

A feature that was developed during data preparation but was not present in the input data at the time of the computation

Topic modeling

An approach for unsupervised machine learning that scans a collection of documents, finds word and phrase patterns within them, and automatically classifies words and expressions that best describe the collection

```
[manju@localhost ~]$ cat marks.txt
Albert 95
Alan 65
John 92
James 66
Mary 77
Yovan 98
Hawking 56
Stephen 44

[manju@localhost ~]$ awk '$2 >= 60 { print $0 }' marks.txt
Albert 95
Alan 65
John 92
James 66
Mary 77
Yovan 98
```

Print all names with marks ≥ 60

Tensor Processing Unit (TPU):

An application-specific integrated circuit for neural network machine learning that Google developed using its own Tensorflow software

Translational invariance:

A machine learning model's potential to disregard the translation of an image or its content. In other words, the model can still identify an animal in an image even if it is rotated or relocated to the top corner

Trend analysis:

A method used in technical analysis that makes predictions about future changes in stock prices using trend data that has just been observed

Universal function approximation theorem:

There is already a neural network available for any function that we want to compute since neural networks have some kind of universality in them

Watson studio:

A data science platform developed by **IBM** that offers the environment and resources necessary for us to work with others to solve business challenges using data

Weight sharing:

A method of neural network model compression that lowers the number of weights in a network that must be trained

XGboost:

A free software package that offers a regularizing gradient boosting framework for the programming languages C++, Java, Python, R, Julia, Perl, and Scala

```
[manju@localhost ~]$ ls -l

total 4

-rw-rw-r--. 1 manju manju   0 Jul 31 07:49 1.c

-rw-rw-r--. 1 manju manju   0 Jul 31 07:49 1.cpp

-rw-rw-r--. 1 manju manju  47 Jul 31 10:03 1.html

-rw-rw-r--. 1 manju manju   0 Jul 31 03:56 1.pdf
```

```
[manju@localhost ~]$ ls -l | awk '{print $6 " " $9}'
Jul 1.c
Jul 1.cpp
Jul 1.html
Jul 1.pdf
```

```
[manju@localhost ~]$ cat names.txt
Albert, Physicist, USA, M, 03/14/1879
Charles, Biologist, Great Britain, M, 02/28/1889
John, Chemist, Germany, M, 05/21/1890
Alan, Mathematician, Russia, M, 05/28/1905

[manju@localhost ~]$ awk -F, '{print $1 " is a" $2}' names.txt
Albert is a Physicist
Charles is a Biologist
John is a Chemist
Alan is a Mathematician

[manju@localhost ~]$ awk 'BEGIN {FS=","}  {print $3}' names.txt
 USA
 Great Britain
 Germany
 Russia
```

```
echo "1,2,3,4,5" | awk '{ split($0,x,",");for (i in x) {sum+=x[i]} } END {print sum}'
```

Go through the array elements and calculate their sum

```
# Output: 15 (1 + 2 + 3 + 4 + 5)
```

"Simplicity is the soul of efficiency."

– Austin Freeman

SAP stands for **System Applications and Products in Data Processing**. One of the world's top software developers for business process management, **SAP** helps companies of all sizes manage their finances, logistics, human resources, and other business areas.

1. Introduction:

SAP is a market leader in offering **ERP** (**Enterprise Resource and Planning**) solutions and services to enterprises across industries for improved work and data management. In 1972, **SAP** was established in Walldorf, Germany. It stands for **Systems, Applications and Products in Data Processing**. It has developed and grown over time to become one of the two leading competitors in the ERP software market as well as the leading global provider of client / server business solutions, for which it is now renowned.

In 180 countries, **SAP** serves more than **335,000** clients, **80%** of whom are small- and medium-sized businesses, according to its **2016** corporate fact sheet. By assisting in better management of **customer needs** and nurturing long-lasting and fruitful customer relationships, **SAP** is a **German multinational software company** whose products enable businesses to track customer and business interactions. SAP also builds a strong technical support for organizations. One of the basic goals of all corporate companies is to satisfy their customers, and **SAP** is particularly well-known for its data management tools that help them do this.

2. What is SAP?

The **technological revolution** of today is being led by **SAP**. **SAP**, the industry leader in enterprise application software, supports businesses in overcoming the negative effects of complexity, creating new chances for innovation and growth, and maintaining a competitive edge. This has helped **SAP** rise to the top of the list of the biggest providers of **ERP** and other enterprise applications.

3. History of SAP

In order to continuously transform the information technology industry, **five** former **IBM** employees founded **SAP** in **1972** with the goal of developing standard application software for the business potential of technology and real-time business processes. Since its founding, **SAP** has released a number of versions, including **SAP R/1**, **SAP R/2**, and **SAP R/3**. The **R** in these releases stands for **"Real-Time"** data processing. The number **3** in the **R/3** stands for **three-tier client-server architecture**.

4. Features of SAP

- Monitors Human Costs
- Documents Progress of Process
- Process of Invoices
- Records External Employee's Services
- Confirms Service Management Orders
- Maintains Planning and Progress
- Collects Data Plans and Sheets

5. Technical details

ERP

The essential business operations of a firm are included in **SAP ERP**. Almost every functional area of a company activity, including the purchase of products and services, sale and distribution, **finance**, accounting, **human resources**, manufacturing, production planning, logistics, and **warehouse management**, is supported and integrated by the **ERP** package. The most recent Enhancement Package (**EHP8**) for **SAP ERP 6.0** was released in **2016**.

SAP R/3

It is **client / server** version of the software and it is **3-tier architecture** which includes **3** layers installed in **3** separate servers. The **3** layers are:

- **Presentation Layer:** that contains the software components that make up the **SAPGUI**.
- **Application Layer:** that consists of one or more application servers and a message server.
- **Data Base layer:** that consists of a central database system containing all of the data in the **R/3**.

SAP R/2

It is a main frame version of software and it is **2-tier architecture** in which **3** layers are installed in **2** separate servers.

SAP R/1

It is the first version and it is **one-tier architecture** in which **3** layers are installed in one server.

6. Functional modules of SAP include:

- **FICO** - Finance & Control
- **PP** - Production Planning
- **MM** - Material management
- **SD** - Sales and distribution
- **HR** - Human resource
- **CRM** - Customer Relationship Management

7. How is SAP in terms of career growth?

SAP provides a faster career growth whether it is functional or technical area, provided the candidate has good fundamental knowledge in **IT** domain and a very sound knowledge of **SAP**. Certification in **SAP** can increase your job prospects. **Fortune 500 Firms** has implemented **SAP R/3** and being successful because of its process integrations among the different organizational functionalities in real time data accessing under a **single common database**. There are various sub-modules of **SAP (**such as Material Management (**MM**), Project System (**PS**), Sales and Distribution (**SD**), Advanced Planner and Optimizer (**APO**), Quality Management (**QM**) etc.**)** and every module in **SAP** has its own demand and uniqueness and has opened up new job opportunities to execute on an aggressive acquisition strategy to fill the technology gaps. Owing to the various advantages, more and more organizations are moving towards adopting **SAP** to impact business in an important and far-reaching way.

Non-parametric learning algorithm:

Algorithms that get their learning from data rather than a mathematical model

Linear Regression:

A **regression** is a technique for figuring out how one variable **"y"** and another variable **"x"** is related. A **linear regression** is a method used to represent a linear relationship between the independent variable **"x"** and the dependent variable **"y"** in statistics. In **Machine Learning**, a **Linear Regression** is a **supervised machine learning algorithm**. When there is a linear relationship between two variables, the dependent variable's value changes in response to changes in the independent variable's value (**increase or decrease**). With the use of the following equation, the relationship can be mathematically represented as:

$$y = mx + c$$

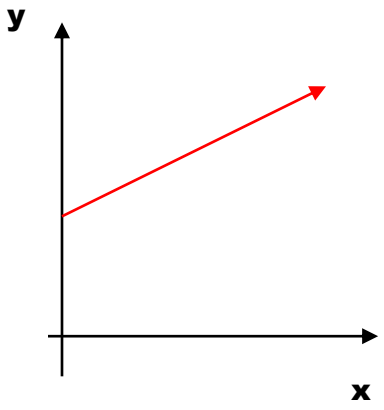
where:

- The dependent variable **"y"** is what we are attempting to predict.
- **"x"** serves as the independent variable for our predictions.
- The slope of the regression line **"m"** shows how much of an impact **"x"** has on **"y"**.
- **"c"** often known as the y-intercept, is a constant. **"y"** would be equal to **m** if **"x"** = 0.

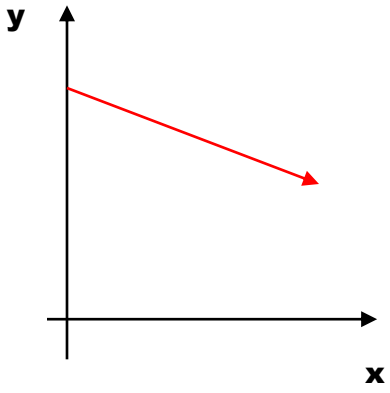
Furthermore, the **linear relationship** can be positive or negative:

Positive Linear Relationship:

If both the **independent** and **dependent** variables increase, the **linear relationship** is said to be positive. The following graph can help you understand it:



Positive Linear Relationship



Negative Linear Relationship

Negative Linear Relationship:

If both the **independent** and **dependent** variables decrease, the **linear relationship** is said to be negative. The above graph can help you understand it:

Random Forest Algorithm:

1. Choose the samples at random from the provided dataset.
2. The second step of this technique is to build a decision tree for each sample. The forecast outcome from each decision tree will then be obtained.
3. Voting will be done for each predicted outcome in this step.
4. Finally, choose the prediction result that received the majority of votes as the final prediction result.

Conducting feature selection before to data modeling will:

- Decrease overfitting
- Speed up training
- Improve model correctness

Epistemology: An exploration of the kind of knowledge needed to address global issues

Dempster Shafer's theory:

A logical theory of evidence based on credible arguments and belief functions. It is used to calculate the likelihood of an event by combining various pieces of evidence

Rule-Based Expert Systems:

The most basic type of Artificial Intelligence, which follows predetermined knowledge-based rules to address issues

PL/SQL Code Block Structure:

[DECLARE]

```
declaration statements;
```

[BEGIN]

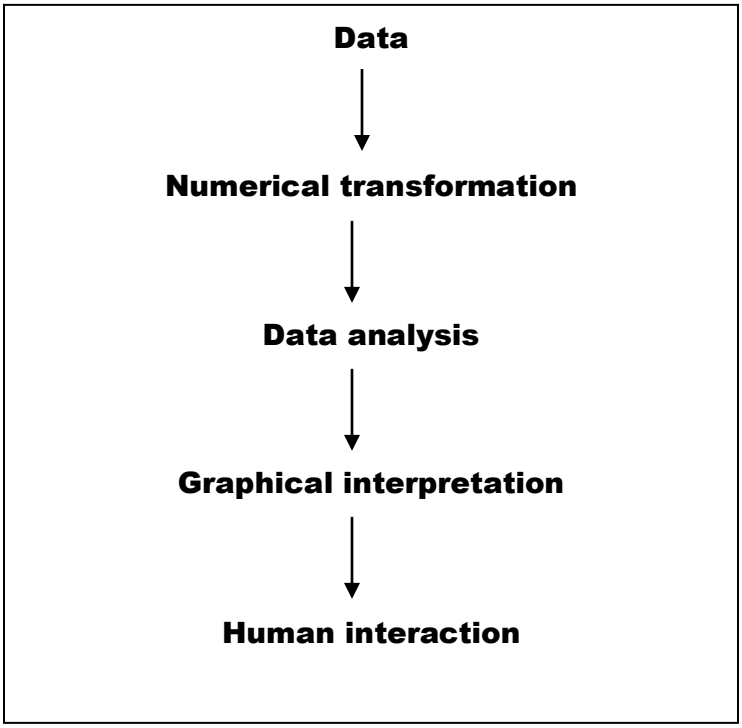
```
execution statements;
```

[EXCEPTION]

```
exception statements;
```

END;

Defined by the keywords DECLARE, BEGIN, EXCEPTION, and END



Transforming Data into Visuals

Data flow diagram	System flow chart
Graphical representation of the information system ↓ The reader of a data flow diagram cannot interpret the result from it easily	Graphical representation of the information system by describing the process step-by-step ↓ The reader of a system flow chart can easily understand the process

Natural language

↓

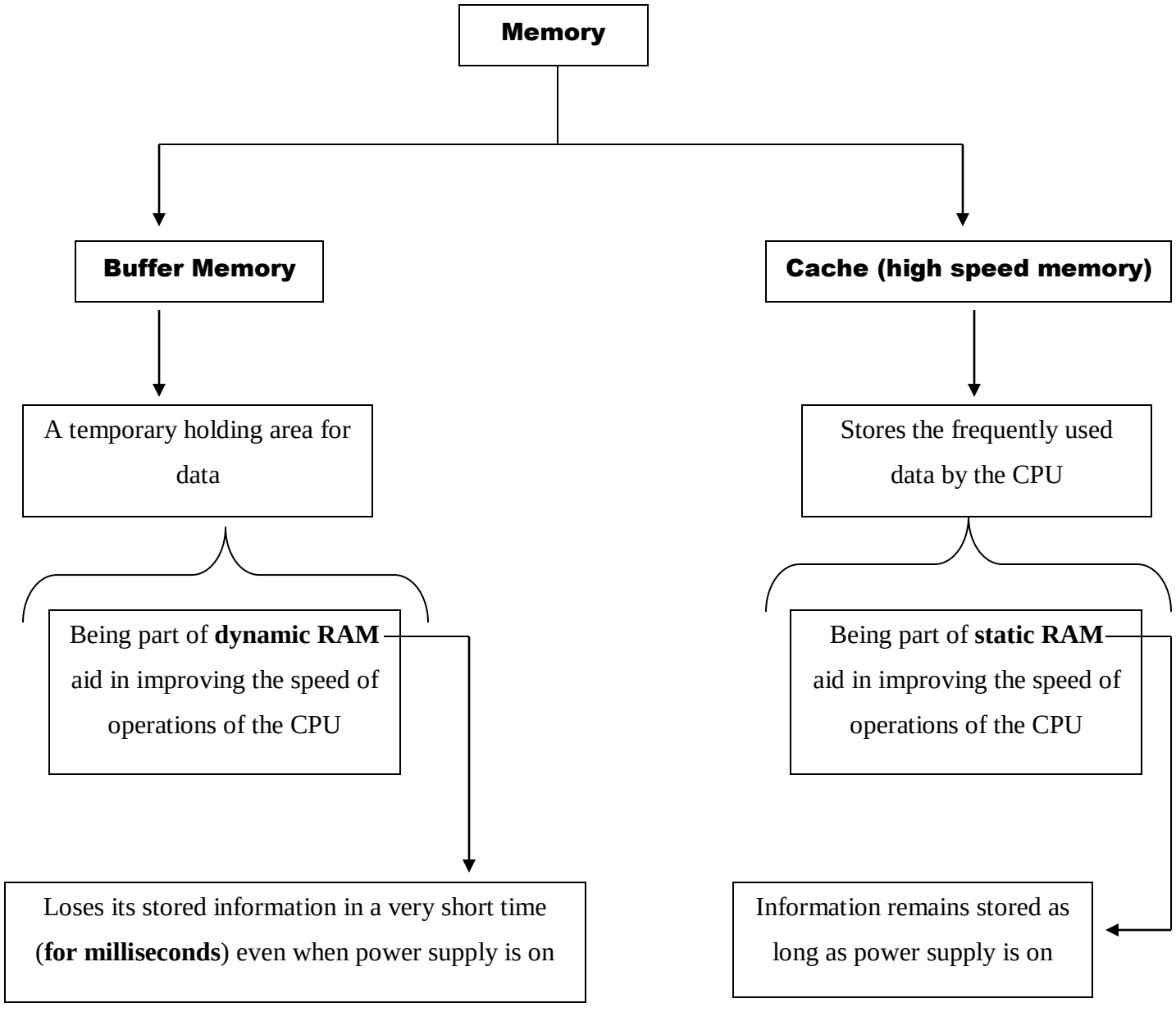
The speech recognition not only can understand the voice but also return answers to questions or other queries that are being asked.

Structure Chart	Flowchart
Represents the basic architecture of the software with the help of modules ↓ Quite complex to understand and design	Represents the graphical structure of an algorithm ↓ Easier to construct and understand

Special purpose registers

(Contain specific information the CPU needs)

- **Instruction Register:** contains the actual instruction which is currently being executed by the CPU
- **The program counter:** contains the address of the next instruction to be executed by the program



DOS (Disk Operating System**) is an operating system that runs from a hard disk drive**

Napier bones

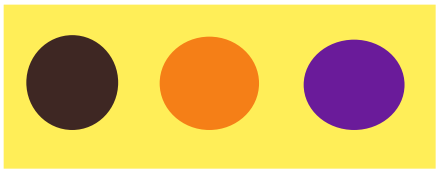
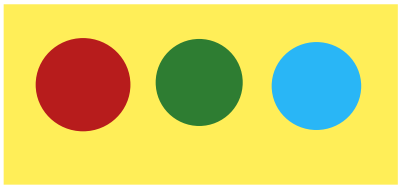
Numbers 0 to 9 were marked on a set of ivory sticks in such a way that the product of any number can be found by placing the sticks side by side.

A programming language is like a natural, human language in that it favors certain metaphors, images, and ways of thinking.

— Mindstorms: Children, Computers, and Powerful Ideas, Seymour Papert (1980)

$$0! = 1$$

$$n! = n \times (n - 1)! \quad \text{if } n > 0$$

Statement (**sentence**)	**Token** (**word**)
	
sequence of tokens (**words**)	sequence of characters (**letters**)

Kernel language	Aid the programmer in reasoning and understanding
Foundational calculus	Mathematical study of programming
Abstract machine approach	Efficient execution on a real machine
Cache mapping	A technique by which the contents of main memory are brought into the cache memory
Record Operations	Making changes to data at the record level

Variables

- local variable
- instance variable
- static variable

Local Variable: A variable declared inside the body of the method

Instance Variable: A variable declared inside the class but outside the body of the method

A variable that is declared as static is called a static variable.

Static binding in Java occurs during compile time while dynamic binding occurs during runtime.

```
// syntax error


#include<stdio.h>
int main()
{
    int x = 23;
    int y = 25;

    printf("%d", (x, y)) // semicolon missed
    return 0;
}
```

```
// run-time error


#include<stdio.h>
int main()
{
    int n = 19, div = 0;

   // wrong logic
   // number is divided by 0,
   // so this program abnormally terminates


    div = n/0;

    printf("result = %d", div);
    return 0;
}
```

```
// linker error
```

Occurs when the executable of the program cannot be generated

```
#include<stdio.h>
int Main() // Here Main() should be main()
{
    int a = 10;
    printf("%d", a);
    return 0;
}
```

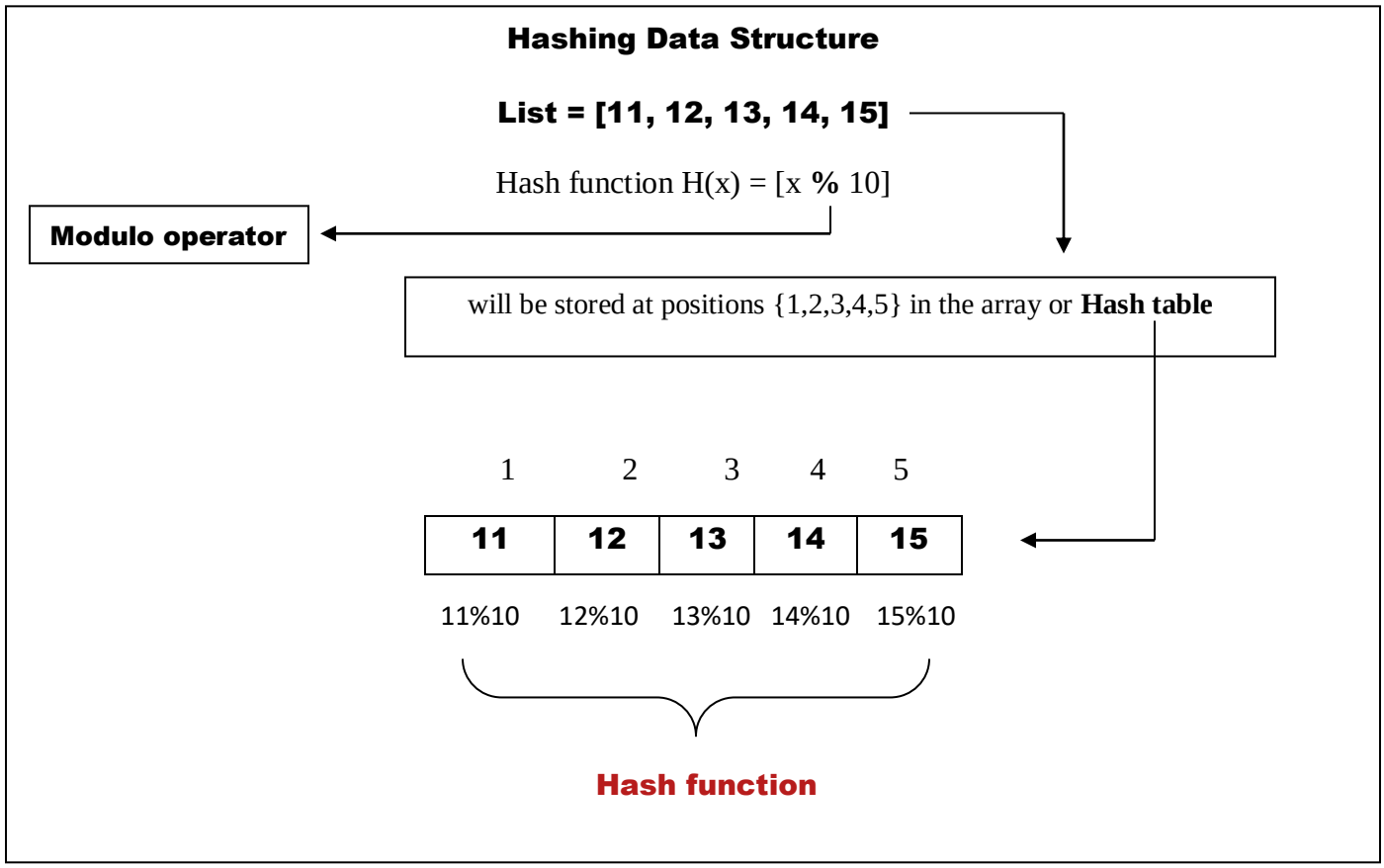
```
// semantic error
```

Occurs when the statements written in the program are not meaningful to the compiler

```
#include<stdio.h>
int Main()
{
  int a, b, c;
  a + b = c; // semantic error
  return 0;
}
```

```
// logical error

#include<stdio.h>
int main()
{
    int i = 0;

    // logical error: a semicolon after loop
```

```
    for(i = 0; i < 3; i++);
    {
        printf("loop ");
        continue;
    }
    getchar();
    return 0;
}
```



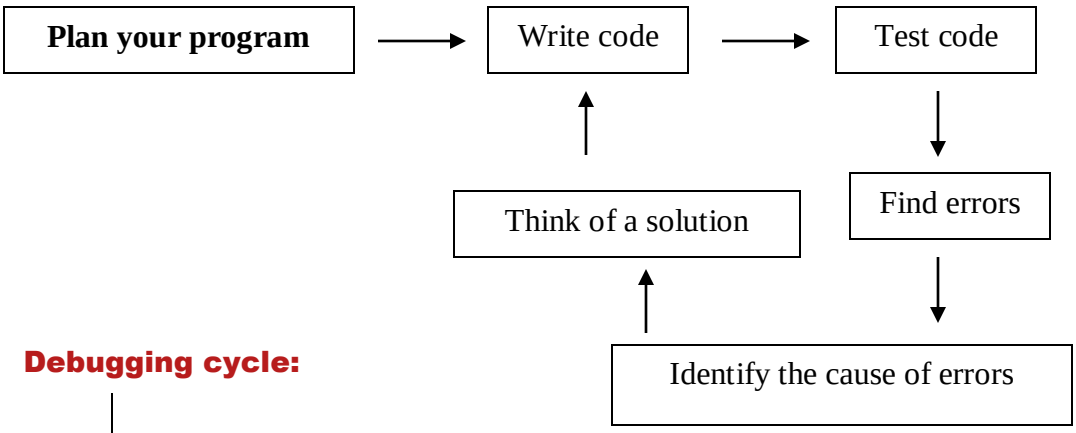
• **h(11)** = 11%10 = 1 • **h(12)** = 12%10 = 2 • **h(13)** = 13%10 = 3	• **h(14)** = 14%10 = 4 • **h(15)** = 15%10 = 5

Class	Object
Blue print from which objects are created	Instance of a class
Declared once	Created many times as per requirement
Does not allocates memory when it is created	Allocates memory when it is created

```
public class MyClass {
  public static void main(String[] args) {
    display(); //there is no object created here because display is a static method
  }
  public static void display(){
    System.out.println("Call from static method");
  }
}
```

- A **static method** can access only static members
- A **non-static method** can access both static and non-static members

```
public class MyClass {
  public static void main(String[] args) {
    HelloWorld tc = new HelloWorld();
    tc.display(); //object is created here because display is a non-static method
  }
  public void display(){
    System.out.println("Call from non-static method");
  }
}
```



Debugging cycle:

Find and remove the number of errors or bugs or defects in a program

Input data testing:

A test phase that concentrates on the accuracy of the data utilized by Machine Learning models during training and prediction

Correctness debugging	Where is the bug?
Performance debugging	Why is it so slow or fast?
Exploratory debugging	What does it do? How does it do it?

Debugging Process:

1. Identify the error
2. Identify the exact location of the error
3. Understand and analyze the type of bug or error
4. Checks all the new errors in the program
5. Fix and validate the bug
6. Retest the program

Computer forensics:

The process of evaluating the damage produced by a security attack and figuring out how it was successful

Android - Application Components

Which are the building blocks of android application?

The main components of the android application are:

- Activities
- Services
- Broadcast Receivers
- Content Providers
- Intent
- View
- Android Virtual Device (AVD)
- Android Emulator

Activities

If you open your phone application, you see number of activities such as received calls, dialed calls, missed calls etc.

If you click on received calls, then another activity (i.e., screen showing the list of received calls) is opened.

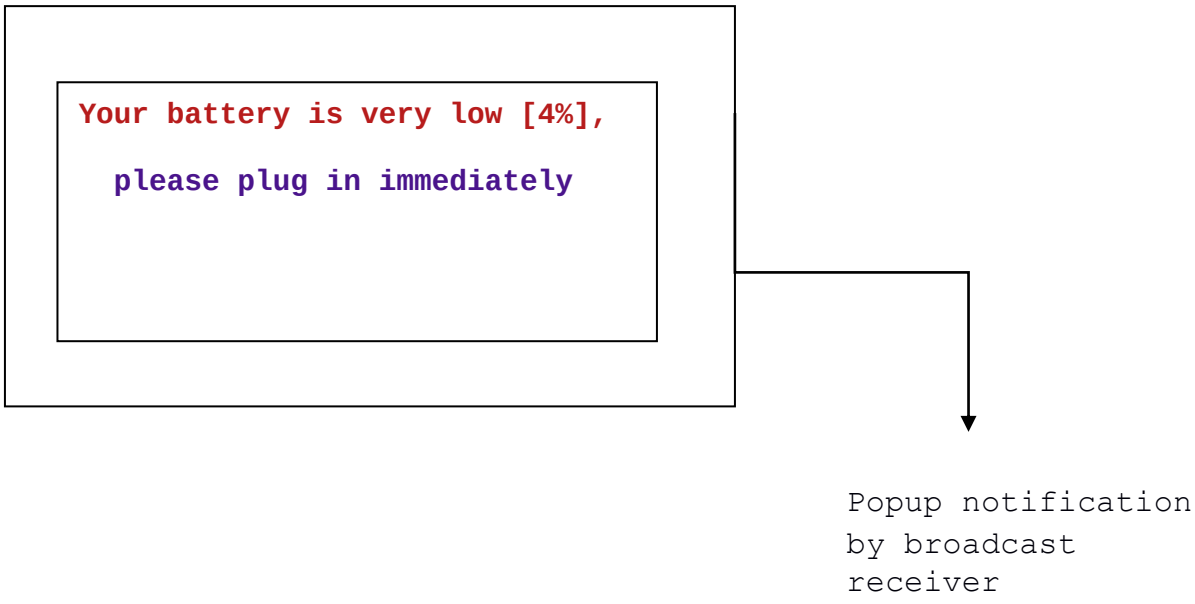
And if you click on one of the received call, then another activity showing the information about the received call (such as the phone number of received call, the time at which it was received etc.) is opened. And if you want to make a call, another activity showing the number keypad is opened.

Services

If you want the music to play in the background or if you want some video to be downloaded while you are browsing over the internet – services provide feasibility for the music to play in the background or video to be downloaded while you are browsing over internet.

Broadcast Receivers

Pop up notifications such as low battery, charging, Power got connected to the mobile device, Power got disconnected from the mobile device, A headset was plugged in, A headset was plugged out.

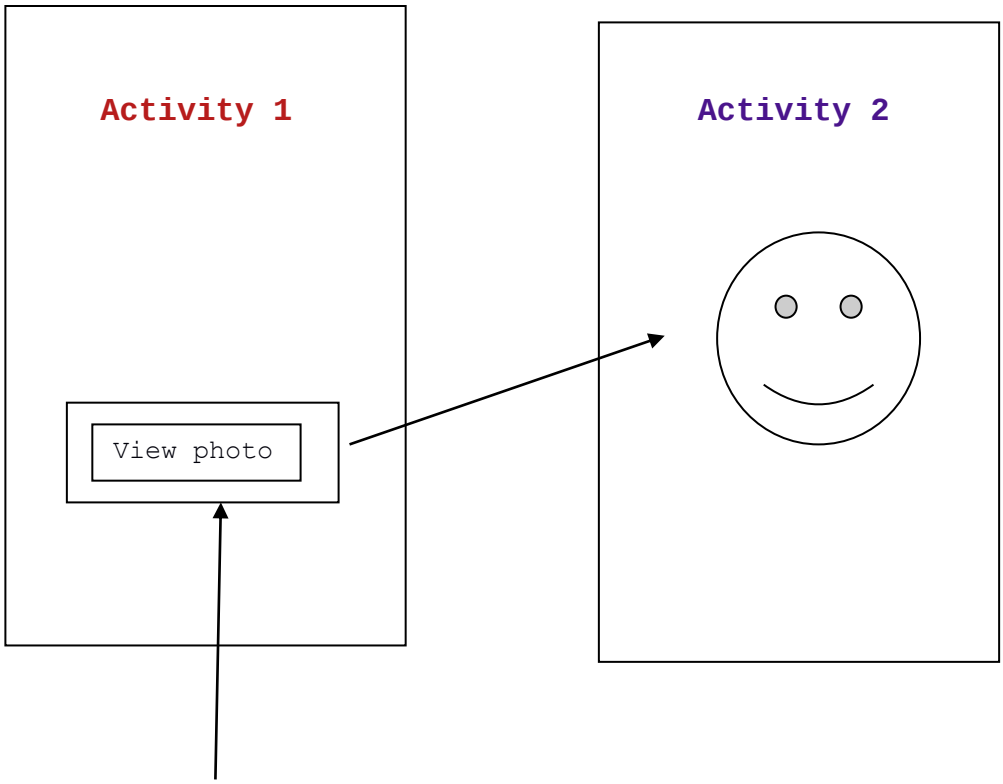


Content Providers

If you type a request for the meaning of a word in the search engine of user dictionary application

User dictionary application sends the request to content resolver and the content resolver sends the request to the content provider and the content provider fetches the information from the database and directs it to the content provider and then from content provider to content resolver and finally from content resolver to user application.

Intent

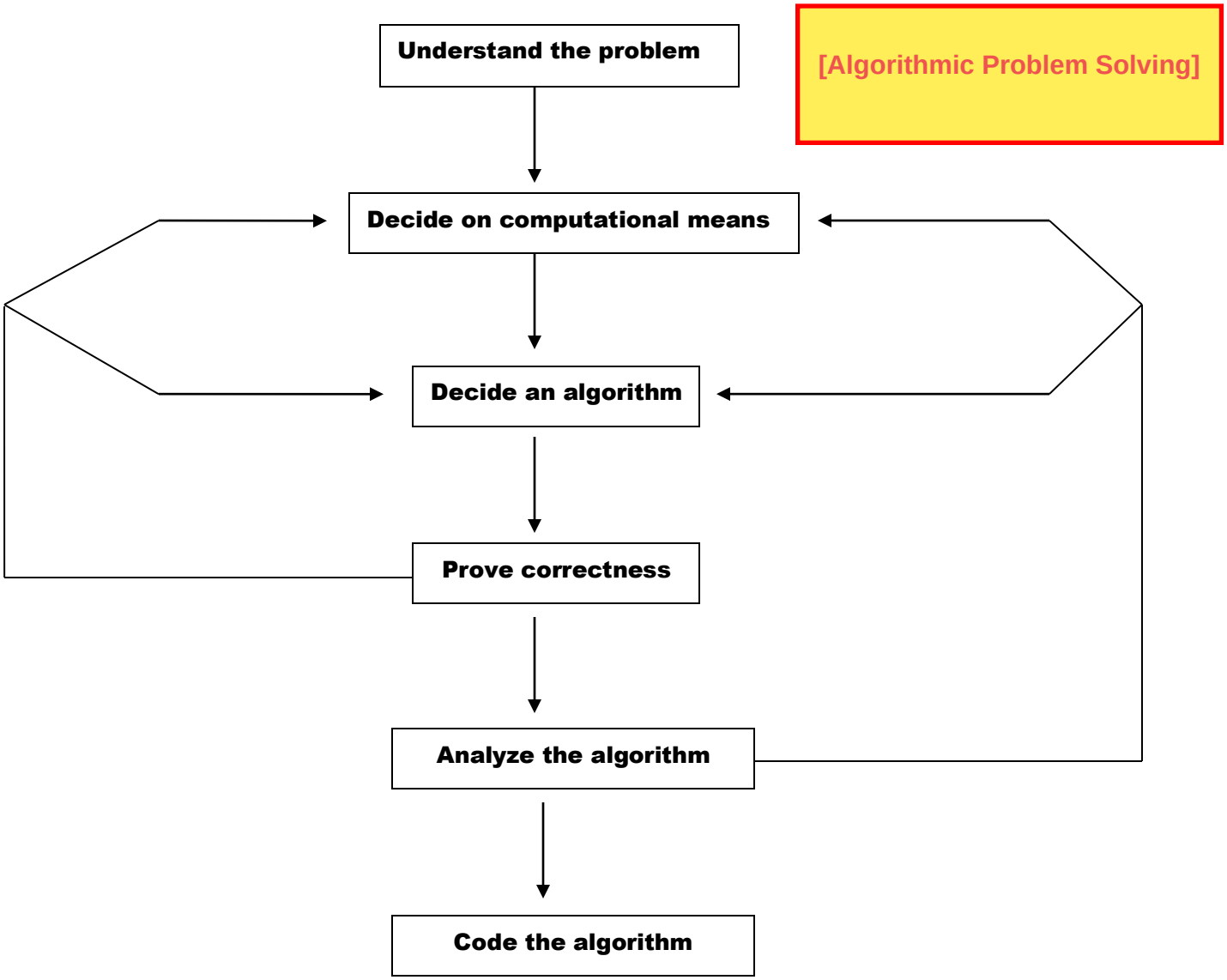


When you press view photo, intent (message) is sent to the android operating system to open another activity (i.e., activity 2) which display the photo

View (apps user interface)

Android Virtual Device (AVD) & Emulator

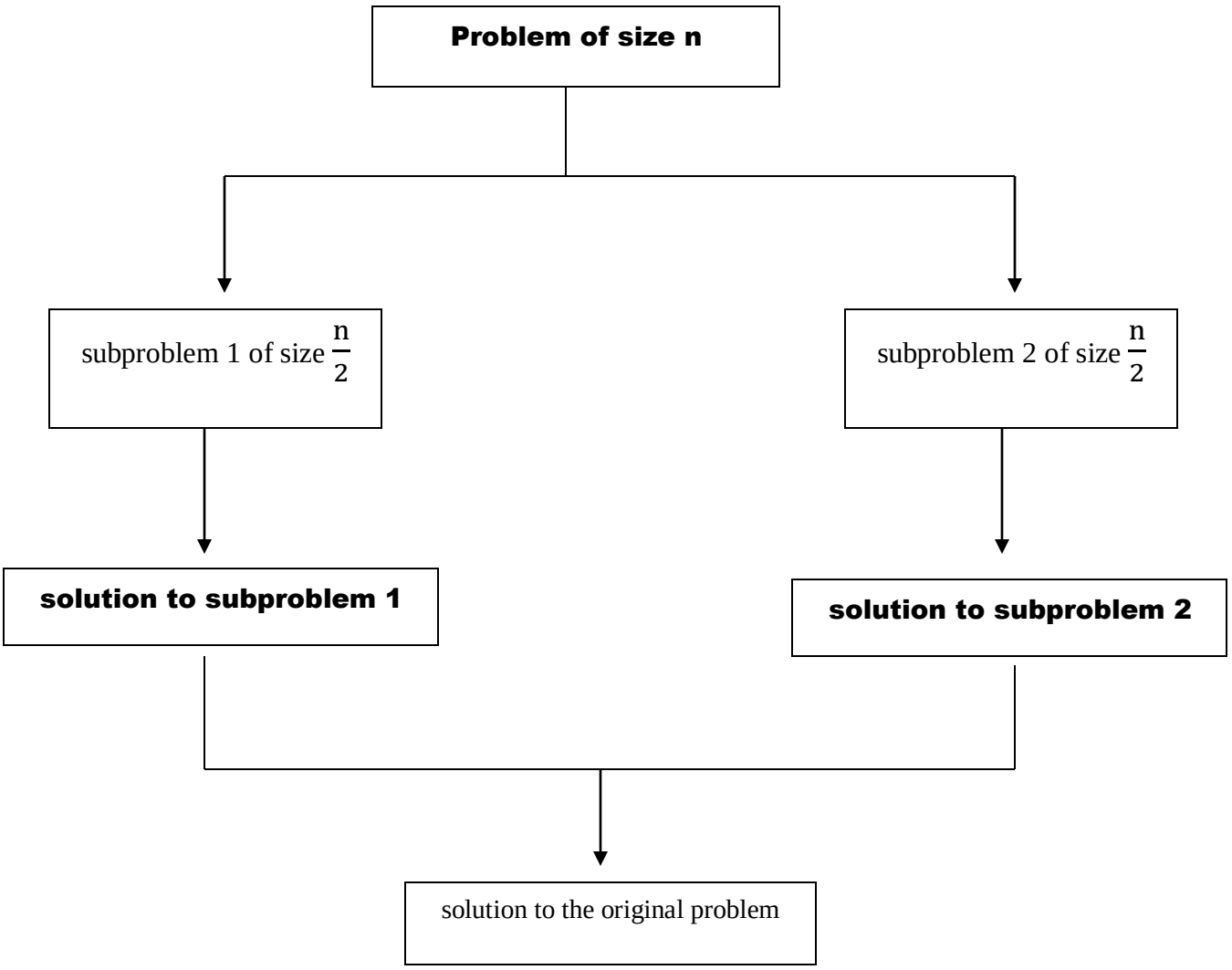
Different android mobile devices possess different configurations. After running and testing your android application on emulator (the component that allows the testing of android application without the necessity to install the application on a physical Android based mobile device) you need Android Virtual Device (AVD) to test whether the application is compatible with a particular android mobile device configuration before installation of the app into that mobile device.



Brute force algorithm finds a solution by trying all possible answers and picking the best one.

If there is a lock of four digit PIN. The digits to be chosen from 0-9 then the brute force will try all possible combinations one by one like 0001, 0002, 0003, 0004, and so on until we get the right PIN. In the worst case, it will take 10,000 tries to find the right combination.

Divide-and-conquer technique

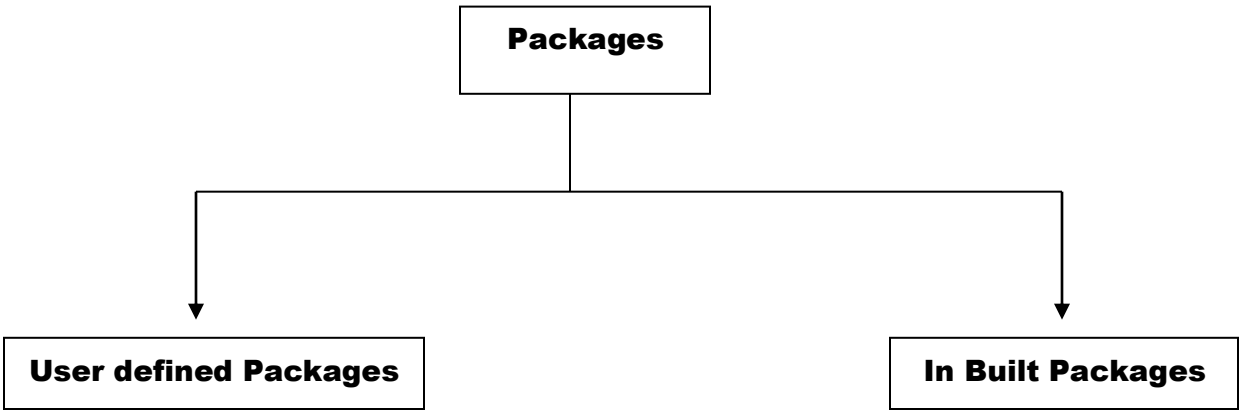


Numerical stability refers to how errors introduced during the execution of an algorithm affect the result

1) Get the current instruction.
2) Determine the meaning of the instruction.
3) Perform the action called for by the instruction.
4) Prepare for getting the next instruction.

Functions of interpreter

Persistent Data Structure	Ephemeral Data Structure
Data structure that always preserves the previous version of itself when it is modified ↓ Modifications are nondestructive	Data structure that do not preserves the previous version of itself when it is modified ↓ A modification destroys the version which we modify



The Reentrant program is a way to handle multitasking.

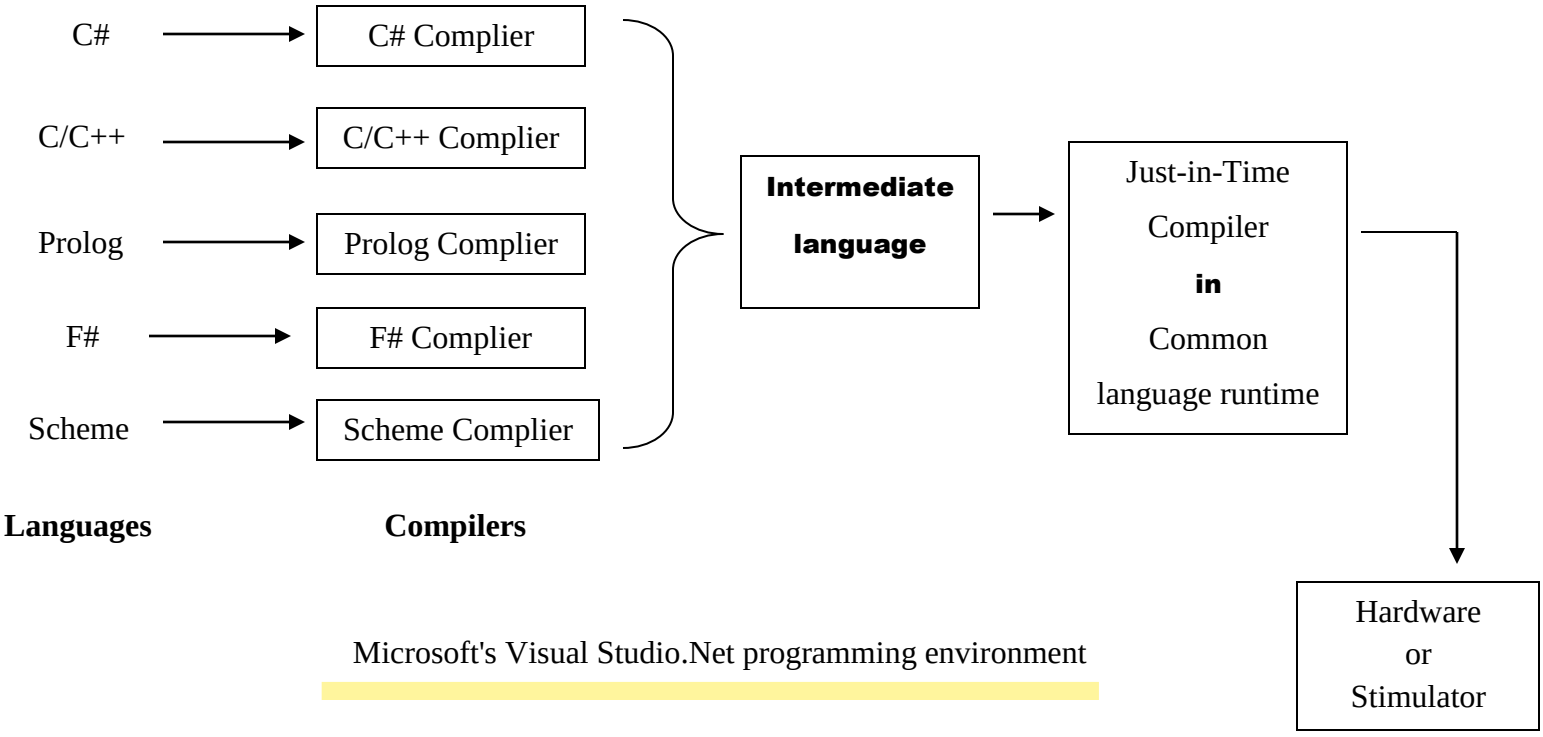
Backtracking

↓

```
If the current solution is not suitable, then eliminate that and backtrack (go back)
and check for other solutions.
```

Time complexity of an program	Space complexity of an program

The amount of time taken by an program to complete its execution as a function of the length of the input	The amount of memory space used by an program to complete its execution as a function of the length of the input
	
	`{`
	`int d = a + b + c;`
	`return(d);`
	`}`
	
	Here, in the program above, a, b, c and d are 4 integers, so each will take 4 bytes. So, the memory required will be 4×4 = 16 bytes. In addition to this memory, while returning value, d will take 4 bytes. Therefore, the total memory requirement will be 16+4 = 20 bytes. This is the **space complexity** of the above-program



Declarative programming → specifies what is to be done rather than how to do it

SQL, HTML, XML and CSS (**model-based programming**)

Does not provide instructions on how to solve the problem

Imperative programming: provides step by step instructions on how to solve the problem

- Ada
- ALGOL
- Assembly language
- BASIC
- Blue
- C
- C#
- C++
- COBOL
- D
- FORTRAN
- Go
- Groovy
- Java
- Julia
- Lua
- MATLAB
- Modula
- MUMPS
- Nim
- Oberon
- OCaml
- Pascal
- Perl
- PHP
- PROSE
- Python
- Ruby
- Rust

- Easy to read and learn
- Solution path is very easy for beginners to understand

- Optimization and extension is more difficult
- Higher risk of errors when editing

Imperative programming (PHP):

```
$participantlist = [1 => 'Peter', 2 => 'Henry', 3 => 'Sarah'];
$firstnames= [];
foreach ($participantlist as $id => $name) {
    $firstnames[] = $name;
}
```

Declarative programming (PHP):

```
$firstnames = array_values($participantlist);
```

Imperative programming

- **Structured Programming:** the code is divided into modules or function and include three basic elements:
  - **Sequence:** Execute a list of statements in order.

```
x = 5
y = 11
z = x + y
WriteLine(z)
```

  - **Selection:** Execute a block of code if a condition is true. The block of code is executed at most once.

```
x = ReadLine()
If x Mod 2 = 0
    WriteLine("The number is even.")
End If
```

- **Repetition:** Repeat a block of code while a condition is true. There is no limit to the number of times that the block can be executed.

```
x = 2

While x < 100

    WriteLine(x)

    x = x * x

End
```

- **Procedural Programming:** the use of code in a step-wise procedure to develop applications
- **Modular Programming:** the process of subdividing a computer program into separate sub-programs

Declarative programming

- **Logic Programming:** presentation of program in the form of symbolic logic
- **Functional Programming:** programs are constructed by applying and composing functions

Imperative programming	algorithms + data (**good for decomposition**)
Functional programming	functions (**good for reasoning**)
Logic programming	facts + rules (**good for searching**)
Object-oriented programming	objects + messages (**good for modeling**)

Structured Programming	**Unstructured Programming**
Code is divided into modules or function	Code is considered as one single block

Easy to read	Hard to read
Easy to do testing and debugging	Hard to do testing and debugging

Prolog program to print `Hello World!`

```
:- initialization(main).
main :- write('Hello World!').
```

```
Hello World!| ?-
```
} **Output on the screen**

For every input array of length $n \geq 1$, the **MergeSort algorithm** performs at most

$6n\log_2 n + 6n$

operations, where $\log_2$ denotes the base-2 logarithm.

Some basic features of Prolog include:

- pattern-matching mechanism
- backtracking strategy that searches for possible solutions
- uniform data structures from which programs are built

Recursion Uses **selection structure**	**Iteration** Uses **repetition structure**
A program is called recursive when a statement in a function calls itself repeatedly	A program is called iterative when a loop repeatedly executes until the controlling condition becomes false
Reduces the size of the code	Makes the code longer
Slow in execution	Fast in execution
Consumes more memory	Consumes less memory

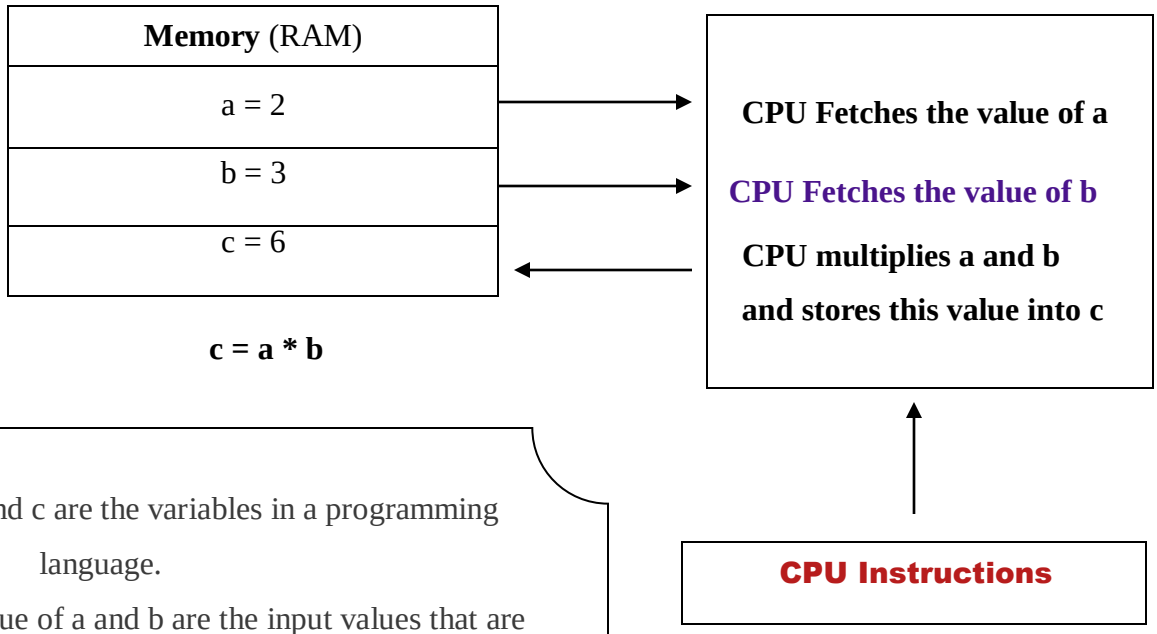
```c
#include<stdio.h>
int fact(int n){
if(n == 0)
return 1;
else
return n * fact(n-1);
}
int main() {
printf("Factorial of 5 is %d", fact(5));
return 0;
}
```

```c
#include<stdio.h>
int main() {
int i, n = 5, fact = 1;
for(i = 1; i <= n; ++i)
fact = fact * i;
printf("Factorial of 5 is %d", fact);
return 0;
}
```

```
Factorial of 5 is 120
```

Stack Memory	Heap Memory
Smaller in size	Larger in size
It is not flexible because we cannot alter the allocated memory	It is flexible because we can alter the allocated memory
Memory allocation is continuous	Memory allocated in random order

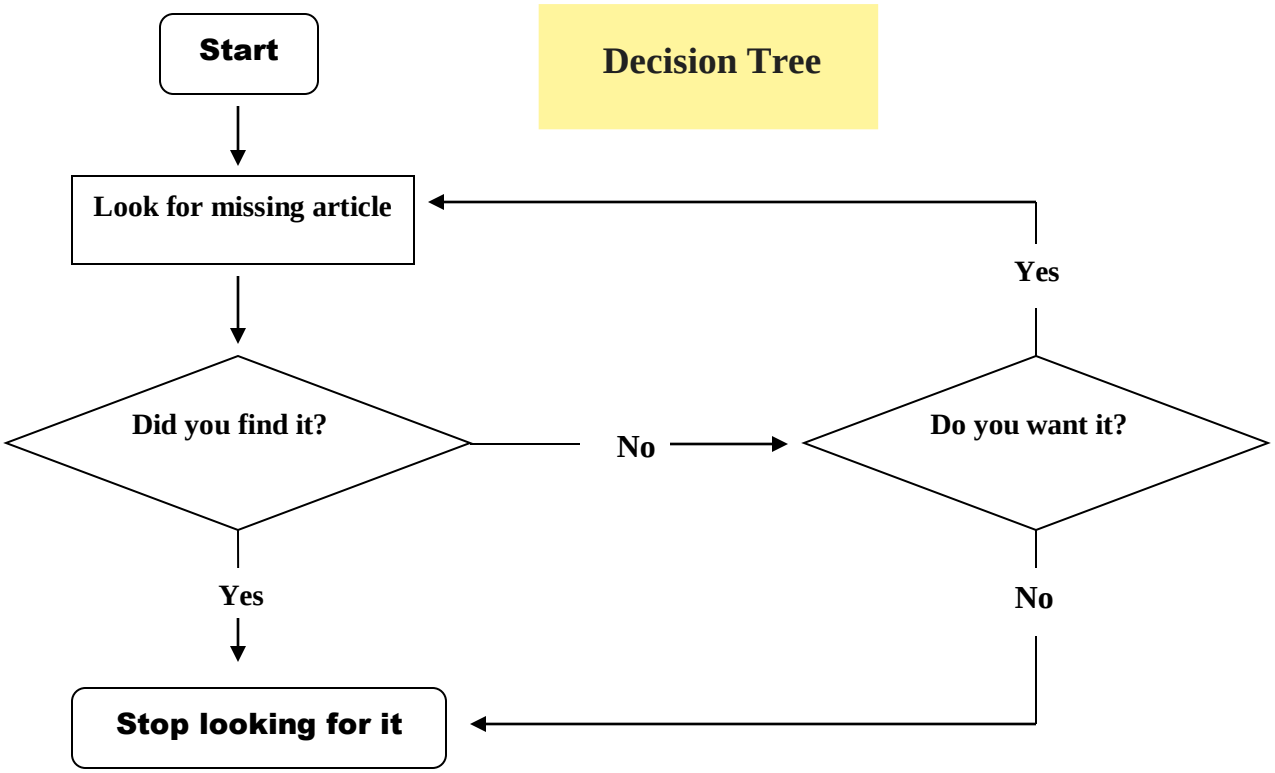
Procedural abstraction refers to using subprograms to hide information about how problems are solved.



c = a * b, where a, b, and c are the variables in a programming language.

If a = 2 and b = 3, the value of a and b are the input values that are called **literals**.

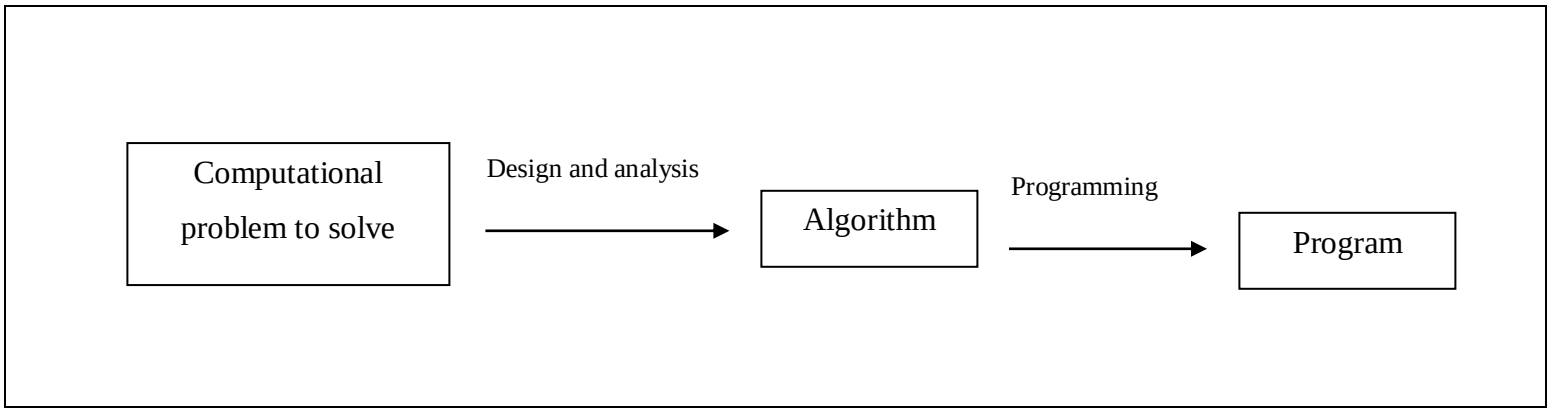
We ask the computer to calculate the value of a*b, which results in c, i.e. the **expected output**.

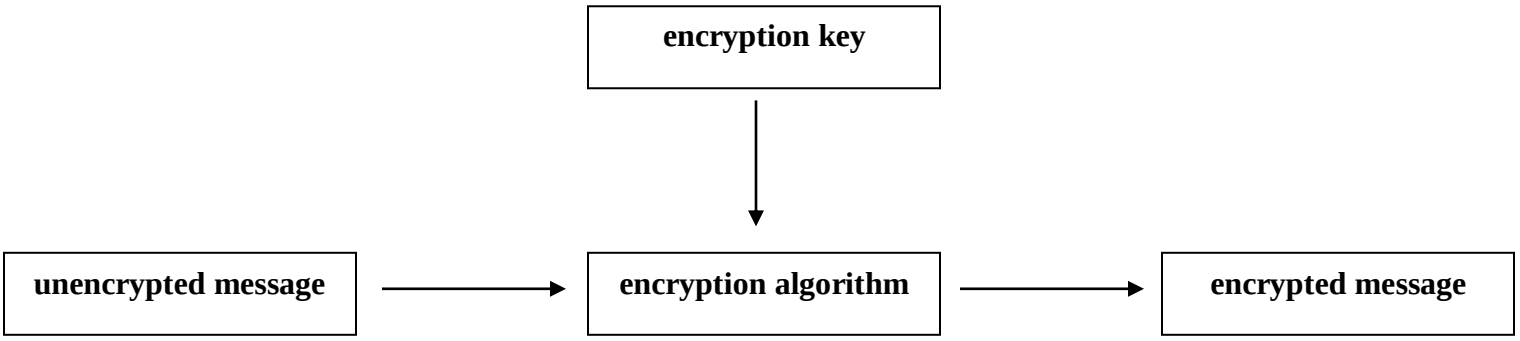


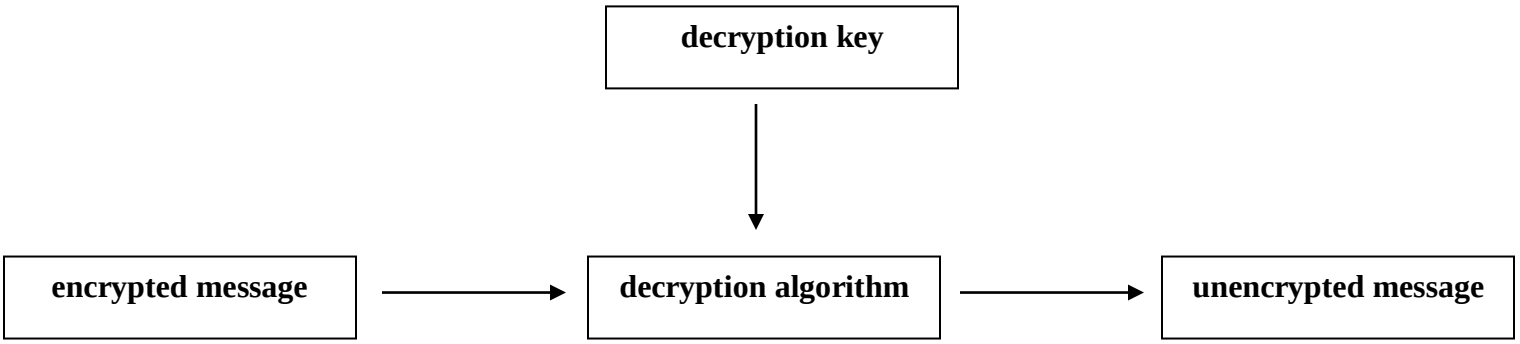
Repeating patterns	Loops
Directions	Algorithms
Order of steps	Sequence
Completed project	Program
Problem solving	Debugging
Paths to completion	Conditionals
Break into steps	Decompose
Amount	Value

Programming Language	Applications
Java	Desktop GUI application (**AWT or Swing api**), Applets, online shopping sites, internet banking, jar files for secured file handling, enterprise applications, mobile applications, gaming software.
C	Operating Systems, Embedded systems, Database management systems, Compiler, gaming and animation.
Python	Machine learning, Artificial Intelligence, Data analysis, face detection and image recognition Software.
C++	Banking and trading enterprise software, virtual machines and compilers.
Visual Basic .NET	Windows services, controls, control libraries, Web applications, Web services.
C#	Desktop applications like a file explorer, Microsoft office applications like Word, Excel, Web browsers, Adobe Photoshop.
JavaScript	Client side and server side validations, DOM handling, developing web elements using jQuery (**JS library**).
PHP	Static and dynamic websites and applications, Server side scripting.
SQL	Querying database, CRUD operations in database programming, creating a stored procedure, triggers, database management.
Objective – C	Apple's OS X, iOS operating system and APIs, Cocoa and Cocoa Touch.

A Prior Analysis	A Posterior Analysis
Theoretical analysis of an algorithm	Empirical analysis of an algorithm
It will give approximate answer and is independent of language of compiler and types of hardware	It will give exact answer and is dependent on language of compiler and type of hardware



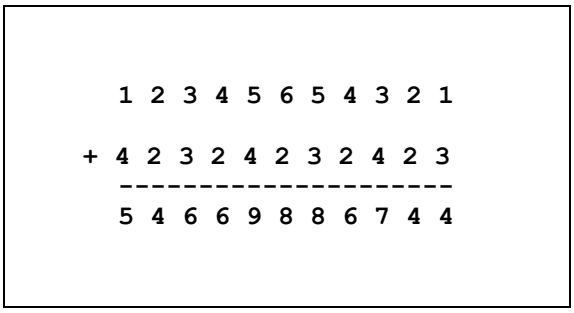




unencrypted message → **1 2 3 4 5 6 5 4 3 2 1**

encryption key → 4232

```
1 2 3 4 5 6 5 4 3 2 1

4 2 3 2 4 2 3 2 4 2 3
```

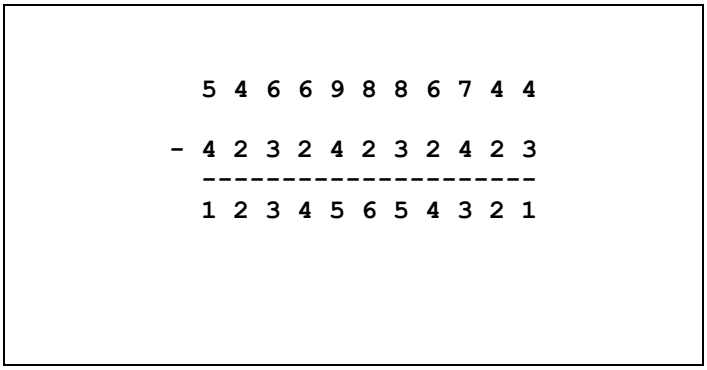


```
  1 2 3 4 5 6 5 4 3 2 1
+ 4 2 3 2 4 2 3 2 4 2 3
  ---------------------
  5 4 6 6 9 8 8 6 7 4 4
```

54669886744 → **encrypted message**

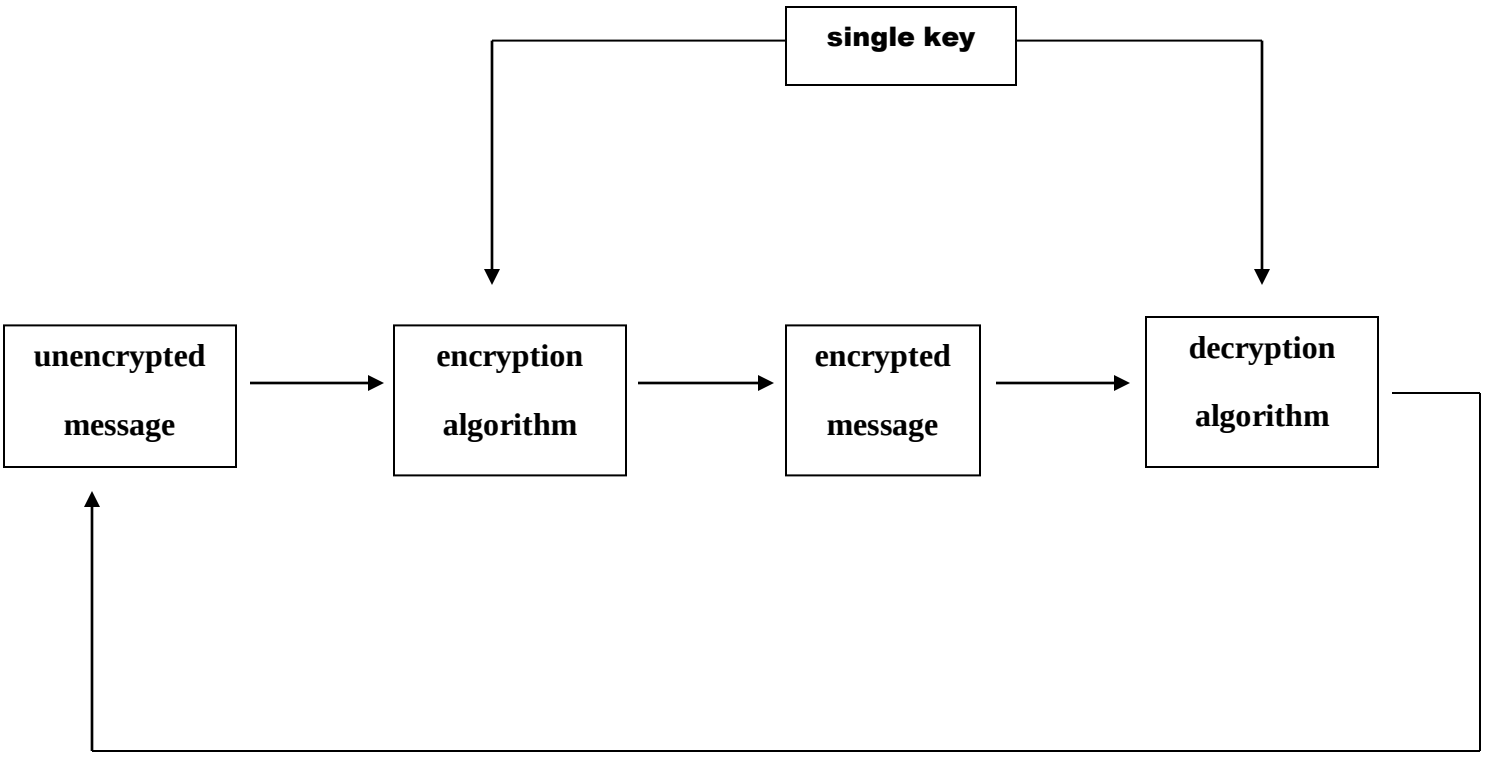
decryption key → 4232

```
5 4 6 6 9 8 8 6 7 4 4

4 2 3 2 4 2 3 2 4 2 3
```



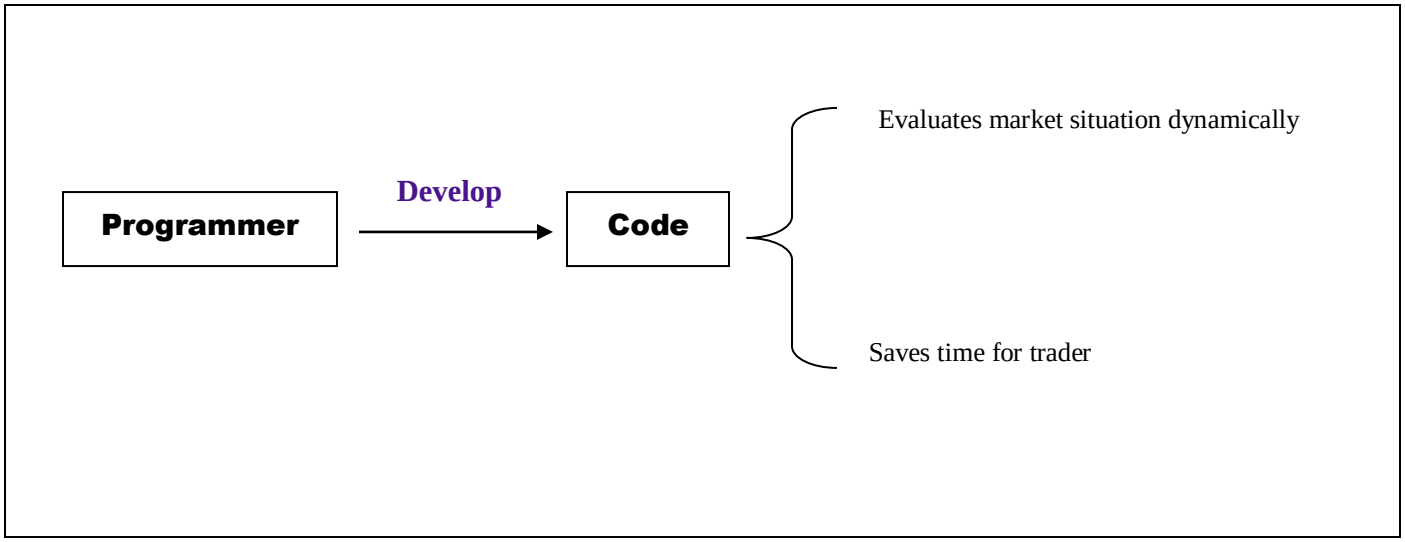
```
  5 4 6 6 9 8 8 6 7 4 4
- 4 2 3 2 4 2 3 2 4 2 3
  ---------------------
  1 2 3 4 5 6 5 4 3 2 1
```

1 2 3 4 5 6 5 4 3 2 1 → unencrypted message



Algorithmic Trading →

The use of algorithms to employ strategies for executing trades

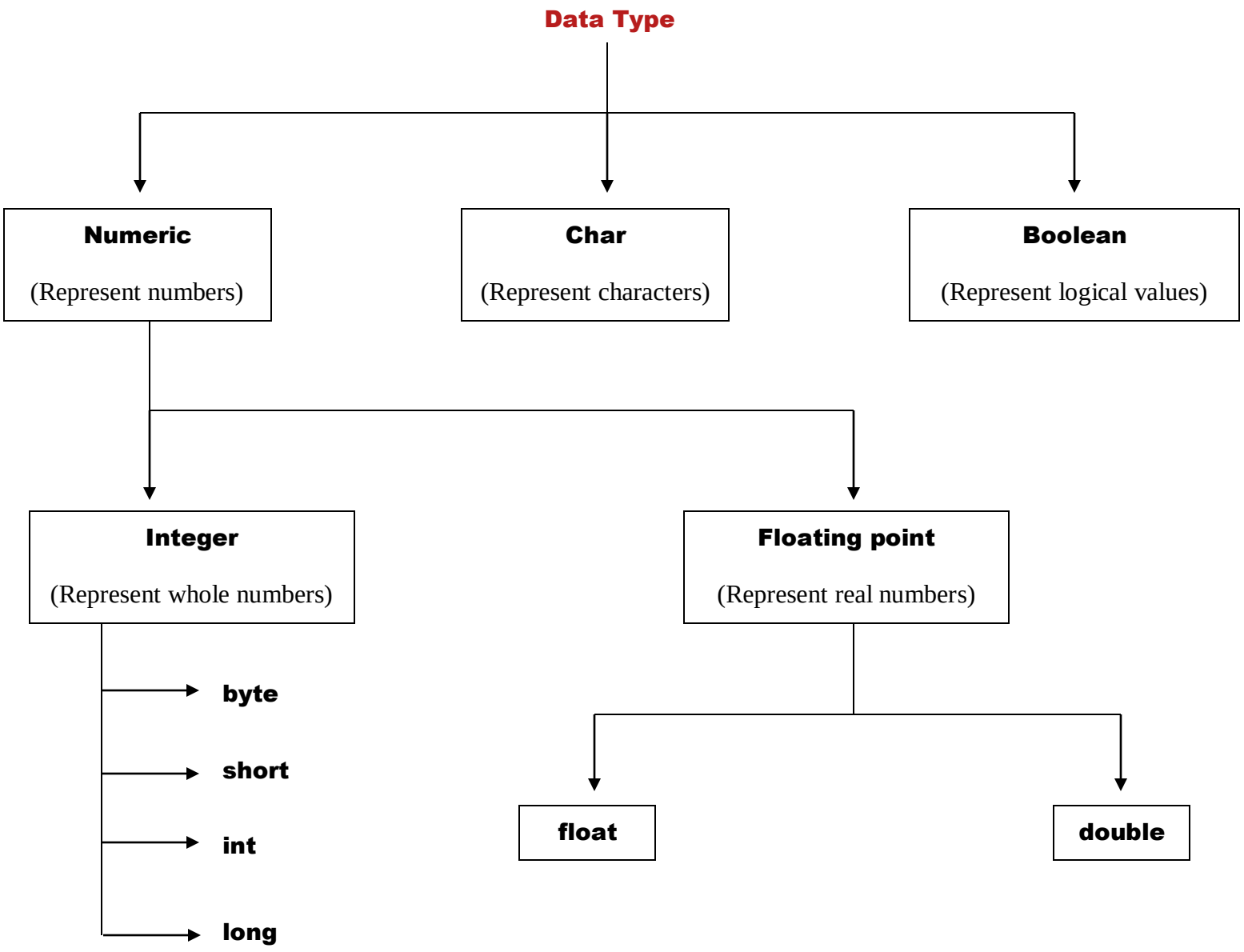


```
#include<iostream>
using namespace std;
int main ( )
{
long long a = 123456789;
long long b = 987654321;
cout << a*b << "\n";
return 0;
}
```

```
#include <iostream>
using namespace std;
typedef long long ll;
int main()
{
ll a = 123456789;
ll b = 987654321;
cout << a*b << "\n";
return 0;
}
```

Output on the screen:

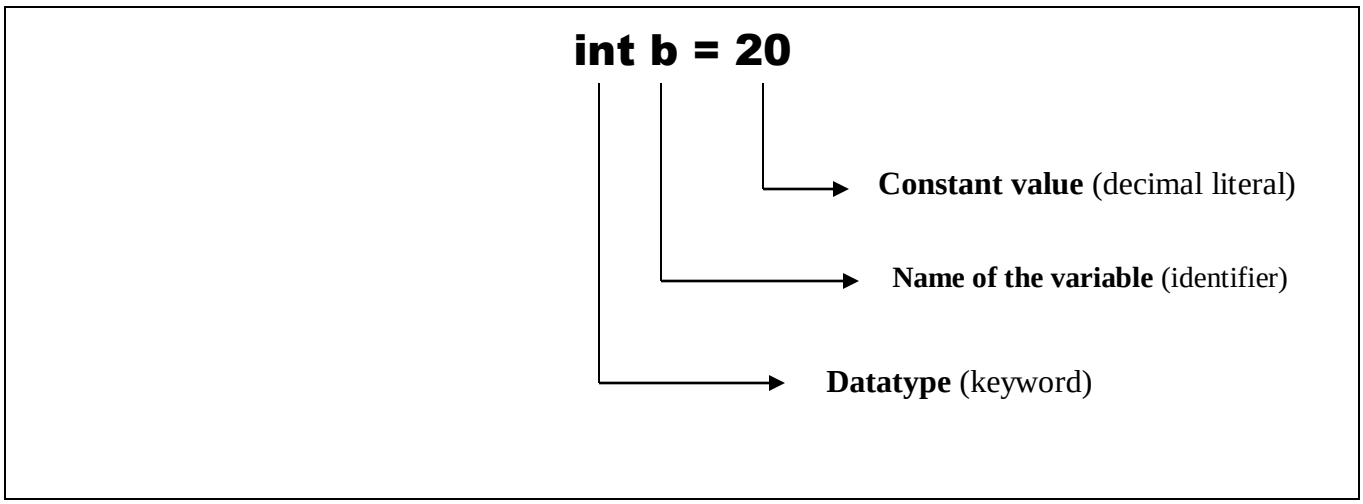
121932631112635269



double	float
14 to 15 decimal places of accuracy	**5 to 6 decimal places of accuracy**
Size: 8 bytes	**Size:** 4 bytes

Data type	Size
byte	**1 byte**
short	**2 bytes**

int	**4 bytes**
long	**8 bytes**
float	**4 bytes**
double	**8 bytes**
boolean	**Not applicable**
char	**2 bytes**




```
public class MyClass {
    static
     {
        for (int i = 0; i < 10; i++)
            System.out.println(i);
            }
   public static void main(String[] args)
   {
      System.out.println("......hi......");
   }
}
```

Output on the screen:

0

1

2

3

4

5

6

7

8

9

.....hi.....

```
public class MyClass {
    static
     {
        for (int i = 0; i < 10; i++)
            System.out.println(i);
            System.exit(0); // it stops execution
            }
   public static void main(String[] args)
   {
      System.out.println("......hi......");
   }
}
```

Output on the screen:

0

1

2

3

4

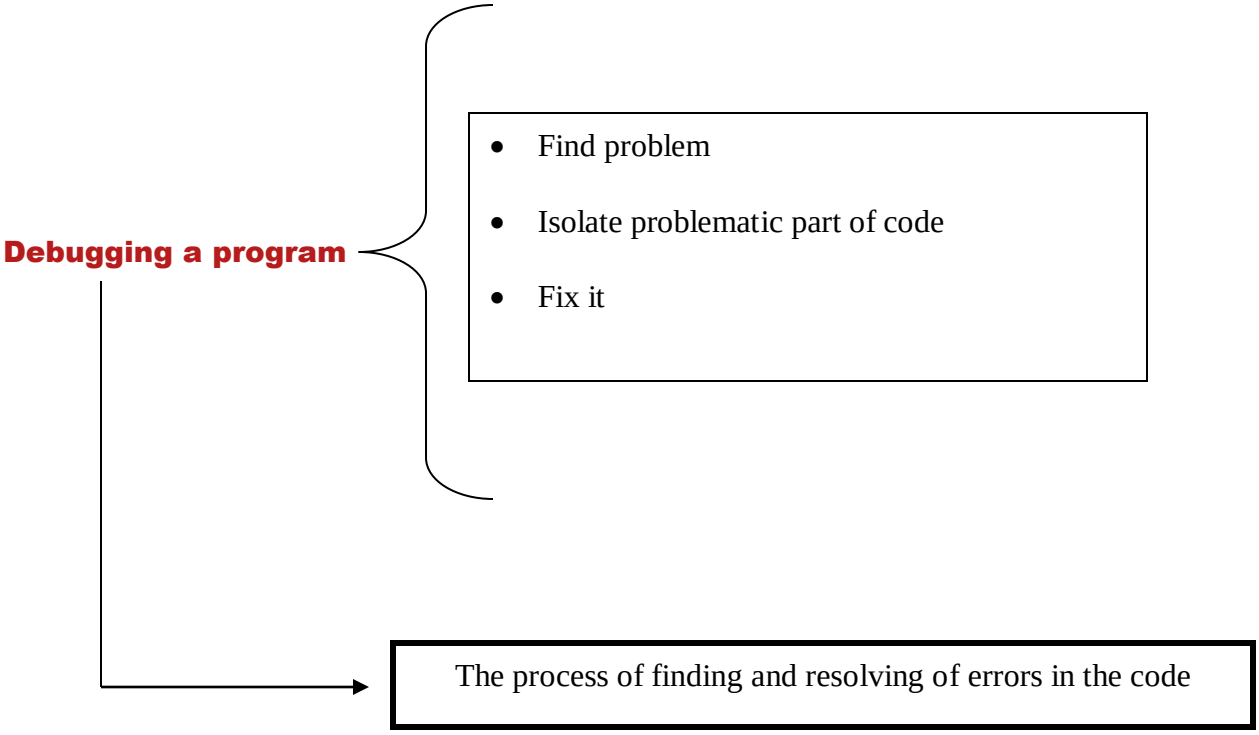
5

6

7

8

9



Mathematical theory

Syntactic rules

(Rules governing the composition of well-formed expressions in a programming language)

Axioms

(Basic theorems which are accepted without proof)

Inference rules

(Mechanisms for deducing new theorems from axioms and previously proved theorems)

Activation of a block → a time interval during which that block is being executed

Logic + Control = Algorithms

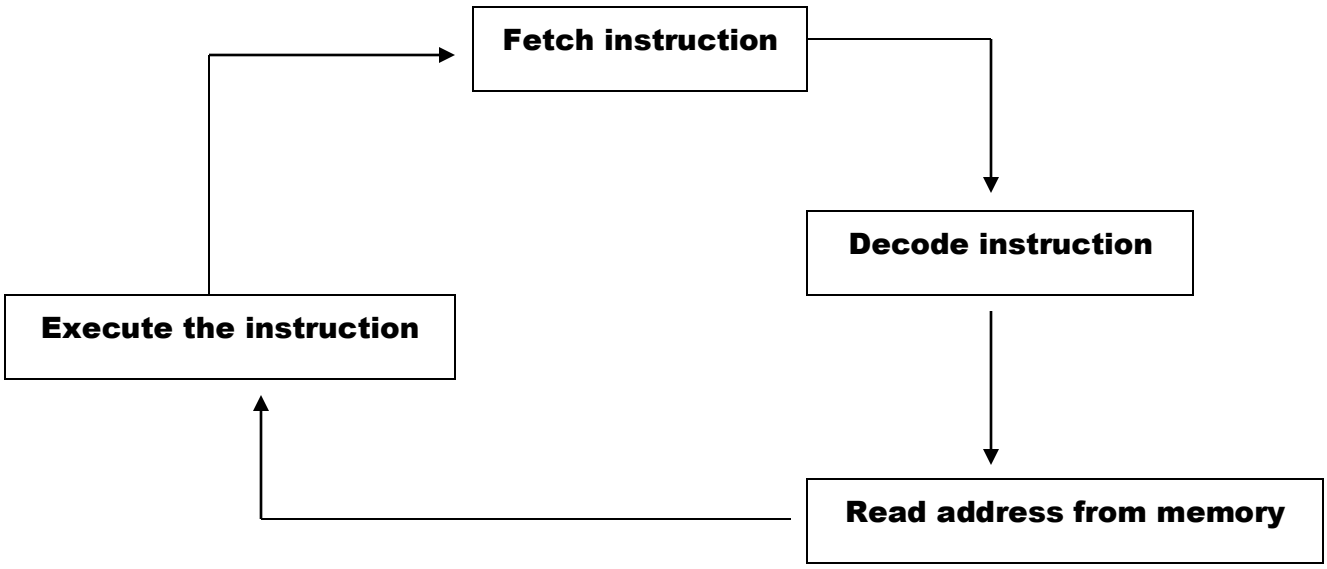
Imperative Programming	Declarative Programming
Specify how it is to be done.	Specify what is to be done.
Describes the control flow of computation.	Expresses the logic of computation.

Domain failure

occurs when the input parameters of an operation do not satisfy the requirements of the operation

Range failure

occurs when an operation is unable to produce a result for values which are in the range



Instruction life cycle

Memory cycle

Memory allocation → Memory usage → Memory release

Python:

Create a list	`a = [2, 3, 4, 5, 6, 7, 8, 9, 10]` `print(a)`

Indexing	`a = [2, 3, 4, 5, 6, 7, 8, 9, 10]`
	
	`# Accessing the item in the position 0`
	`print(a [0])`
	
	`# Accessing the item in the position 5`
	`print(a [5])`

	# Accessing the last item using negative indexing **print(a [-1])**
Slicing	a = [2, 3, 4, 5, 6, 7, 8, 9, 10] **# printing a part of the list** **print(a [0:3])**
Concatenation	a = [2, 3, 4, 5, 6, 7, 8, 9, 10] b =[11,12] **# adding and printing the items of two lists** **print(a + b)**
Repetition	b =[11,12] **# create a multiple copies of the same list** **print(b*3)**
Updating	a = [2, 3, 4, 5, 6, 7, 8, 9, 10] **# updating the list using index value** a[2]=13 **print(a)**
Membership	a = [2, 3, 4, 5, 6, 7, 8, 9, 10] **# returns true if item is present in list. otherwise returns false** **print (5 in a)** **print (13 in a)**
Comparison	a = [2, 3, 4, 5, 6, 7, 8, 9, 10] b =[11,12] **# returns true if all items in both lists are same. otherwise returns false** **print(a==b)**

```
# list
a =[12, "John", "C++", 78.5]
print(a)

# tuple
b =(13, "Joseph", "Java", 88.5)
print(b)

# dictionary
```

```
c ={"Roll no":15, "class":"Python", "percentage":98.50}
print(c)
```

Output on the screen:

```
[12, 'John', 'C++', 78.5]
(13, 'Joseph', 'Java', 88.5)
{'Roll no': 15, 'class': 'Python', 'percentage': 98.5}
```

Command:

```
yum search httpd
```

Description:

Search for a Package (**apache**) {**CentOS**}

Command:

```
yum check-update
```

Description:

Find how many of installed packages on your system have updates available {**CentOS**}

Command:

```
yum grouplist
```

Description:

List all available Group Packages {**CentOS**}

Command:

```
yum repolist
```

Description:

List Enabled Yum Repositories

Command:

```
yum clean all
```

Description:

Clean Yum Cache

Command:

```
yum history
```

Description:

View History of Yum

Command:

```
yum repolist all
```

Description:

List all Enabled and Disabled Yum Repositories

Command:

```
netstat -r
```

Description:

Display Kernel IP routing

Type casting is a way to convert a variable from one data type to another data type.

Implicit Conversion

(do not require any operator)

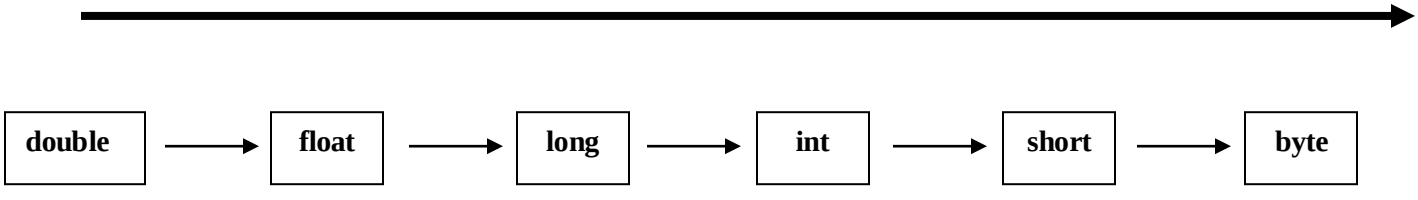
Explicit Conversion

(require operator)

- **Widening:** Converting a lower data type into higher data type
- **Narrowing:** Converting a higher data type into lower data type
- **Automatic type conversion:** Type conversion done by the compiler wherever required

Static Typing	Dynamic Typing
Variables have types	Variables have no types
Values have types	Values have types
Variables cannot change type	Variables change type dynamically

Narrowing Type Casting



Widening Type Casting

- **Mutable objects** → The objects in which you can change the fields and states after the object is created.
- **Immutable objects** → The objects in which you cannot change the fields and states after the object is created.

Threads created by Java Virtual Machine → **Daemon threads**

Threads created by an application itself → **User threads**

Steps to create Java Database Connectivity Application:

- Import the packages
- Register the Java Database Connectivity driver
- Open a connection
- Execute a query
- Extract data from the result set
- Clean up the environment

Training data → Learn Algorithm → Build Model → Perform → Feedback

Machine Learning

Tableau → data visualization tool

- Combines data from multiple sources
- Applies logic and mathematics to data to provide insights for making better quick decisions
- Analyze multiple independent datasets to gain the combined insight
- Visualize and create interactive, shareable dashboards.

6 steps involved in Software Testing Life Cycle:

- Requirement Analysis
- Testing strategy is defined
- Testing team prepares the test data for testing
- Set up the test environment
- Test analysts test the data and compare the expected and actual results.
- Note the summary of the bugs removed and errors found during the software development life cycle.

A **blockchain** is a backlinked, decentralized and distributed-database of encrypted records, called blocks, which are linked and secured using cryptography.

The most common Shells used in Linux are:

- bash (**Bourne Again Shell**)
- ksh (**Korn Shell**)
- csh (**C Shell**)
- zsh (**Z Shell**)
- fish (**Friendly Interactive Shell**)

- `free -m`
- `vmstat`
- `top`
- `htop`

Commands for Checking Memory Usage in Linux

3 kinds of permission in Linux:

- **r – Reading permission:** Allows a user to open and read the file
- **w – Writing permission:** Allows a user to open and modify the file
- **x – Execution permission:** Allows a user to run the file.

The process states in Linux:

- The process is created
- The process is ready to run
- The process is running
- The process is waiting for input from the user
- The process has been stopped
- The process is dead but its information still exists in the process table

Defect density:

The number of flaws that were identified during software development divided by the size of the software

Concept	Description
Abstraction	Hides the implementation details and only provides the functionality to the user
Encapsulation	Wraps up the data and code together as a single unit
Inheritance	Where one class acquires the properties of another
Polymorphism	The ability of a variable, function or object to take multiple forms

2 types of polymorphism:

- Compile time polymorphism
- Run time polymorphism.

Used for

- Method Overriding
- Code Reusability

Constructor is basically a method that is automatically called to create an object.

2 types of Constructor:

- **Default Constructor** → can be called with no arguments.
- **Parameterized Constructor** → can be called with specific number of arguments.

Problem Occurs → **Create Exception** → Throw Exception → **Handle Exception**

Exception handling:

The process of having a software program react to unwelcome or unexpected events. Without this process, these events would interfere with a program's normal execution and cause it to crash

Differences between C, C++ and Java:

	C	C++	Java
Programming Paradigm	Procedural language	Object-Oriented Programming (OOP)	Pure Object Oriented Oriented
Origin	Based on assembly language	Based on C language	Based on C and C++
Developer	Dennis Ritchie in 1972	Bjarne Stroustrup in 1979	James Gosling in 1991
Translator	Compiler only	Compiler only	Interpreted language (Compiler + interpreter)
Platform Dependency	Platform Dependent	Platform Dependent	Platform Independent
Code execution	Direct	Direct	Executed by JVM (Java Virtual Machine)
Approach	Top-down approach	Bottom-up approach	Bottom-up approach
File generation	.exe files	.exe files	.class files
Pre-processor directives	Support header files (#include, #define)	Supported (#header, #define)	Use Packages (import)
keywords	Support 32 keywords	Supports 63 keywords	50 defined keywords
Datatypes (union, structure)	Supported	Supported	Not supported
Inheritance	No inheritance	Supported	Supported except Multiple inheritance
Overloading	No overloading	Support Function overloading (Polymorphism)	Operator overloading is not supported
Pointers	Supported	Supported	Not supported
Allocation	Use malloc, calloc	Use new, delete	Garbage collector
Exception Handling	Not supported	Supported	Supported
Templates	Not supported	Supported	Not supported
Destructors	No constructor neither destructor	Supported	Not supported
Multithreading/ Interfaces	Not supported	Not supported	Supported
Database connectivity	Not supported	Not supported	Supported
Storage Classes	Supported (auto, extern)	Supported (auto, extern)	Not supported

Write a python program to get all links from a webpage

```
from bs4 import BeautifulSoup
from urllib.request import Request, urlopen

soup = BeautifulSoup(urlopen(Request("http://www.scribd.com/")), "lxml")

urls = []

for url in soup.findAll('a'):
    urls.append(url.get('href'))

print(urls)
```

Write a python program to Print all Integers that Aren't Divisible by Either 2 or 3

```
# the maximum number to which you want to print
x = 20

# assign a value to a variable 'i'
i = 1

# run until it reaches maximum number 'x'
print("Numbers not divisible by 2 and 3:")
while i <= x:

    # check if 'i' is divisible by 2 and 3
    if i % 2 != 0 and i % 3 != 0:
        print(i)

    # incrementing 'i'
    i = i+1
```

Write a Python program which iterates the integers from 1 to 20. For multiples of 3 print "Albert" instead of the number and for the multiples of 5 print "Elsa". **For numbers which are multiples of both 3 and 5 print "Elsa Albert".**

```
for i in range(21):
    if i % 3 == 0 and i % 5 == 0:
        print("Elsa Albert")
        continue
    elif i % 3 == 0:
        print("Albert")
        continue
    elif i % 5 == 0:
        print("Elsa")
        continue
    print(i)
```

Write a Python program to convert seconds into hours, minutes and seconds

```
# Python Program to convert seconds into hours, minutes and seconds

def convert(seconds):
    seconds = seconds % (24 * 3600)
    hour = seconds // 3600
    seconds %= 3600
    minutes = seconds // 60
    seconds %= 60

    return "%d:%02d:%02d" % (hour, minutes, seconds)


x = 98765
print(convert(x))
```

Data Science

↓

Statistics + Data Analysis + Machine Learning

(Evaluate the data to derive knowledge and insights and make future predictions)

Aids in →

- **Better choices** (What should we pick, X or Y?)
- **Analysis of the future** (What will take place next?)
- **The discovery of patterns** (Search for patterns or any concealed information in the data)

Conventional Programming:

Computing Data + Computer Algorithm → Conclusion

Machine Learning:

Computing Data + Conclusion → Computer Algorithm

- **Population:** A collection of people or things from which we want to gather data
- **Census:** A gathering of information on each person within a population
- **Sample:** Information on a particular segment of the population in order to represent all

"Machine Learning is a field of study that gives computers the ability to learn without being programmed."

Arthur Samuel (1959)

Azure Machine Learning (Azure ML):

A platform for building and managing **machine learning** applications in the cloud. It is intended to assist data scientists and machine learning engineers in utilizing their current model development and data processing skills and frameworks

- **Bandits:** A class of algorithms where users repeatedly select from or make judgments in response to various options
- **Box plot:** A diagram that illustrates how the data vary in terms of pattern
- **Causal modeling:** Build a model to explain the connections between ideas connected to a particular phenomenon
- **Data Partitioning:** The process of dividing data over several tables, drives, or locations in order to speed up query processing or make databases easier to maintain
- **Data Product:** Any tool or software that processes data and produces output
- **Dichotomous:** A result with only two possible values. **For example**: a coin tossing experiment's results are dichotomous ("head" or "tail")
- **Discriminant Analysis:** A statistical method that divides data into distinct groups according to how well they perform on one or more quantitative predictor factors

Central Limit Theorem:

Irrespective of the probability distribution of the population from which the sample is obtained, the sampling distribution of the mean reaches normality as the sample size increases

- **Endogenous Variable:** A variable in a statistical model whose effect or significance is affected by how it interacts with other variables in the model
- **Exact Test:** A test where all presumptions made during the distributional derivation of the test statistic are fulfilled if the null hypothesis is true

- **Face validity:** The degree to which a test is seen as covering the idea that it is meant to measure
- **Factor Analysis:** A method for breaking down a large number of variables into a smaller number of factors
- **Family-wise Type I Error:** The likelihood that we will incorrectly assume that at least one pair of populations differs, even if all samples come from the same population
- **Frequency Interpretation of Probability:** When an experiment is repeated a very large number of times, the probability of an event is the proportion of times the event takes place

Heteroscedasticity → unequal variation of data

Homoscedasticity → equal variation of data

- **Likelihood Ratio Test:** A statistical test that compares the likelihood ratios of two models to assess the goodness of fit of the models
- **Meta-analysis:** An epidemiologic study design that is rigorous, systematic, and utilized to methodically evaluate the findings of earlier research in order to draw conclusions about that body of research
- **Normality:** A property of a random variable that follows the normal distribution
- **Predictive Validity:** An assessment of the degree of agreement between the outcomes produced by the instrument under evaluation and those produced by more direct and objective measurements
- **Repeatability:** The difference in results from experiments conducted under the same circumstances
- **Reproducibility:** The variation of results of an experiment performed under conditions that vary within a typical range, such as when measurement is performed by the same equipment by various operators in various laboratories
- **Sequential Analysis:** A statistical analysis in which decisions are taken concerning the size of the sample and the type of data to be gathered and updated as the investigation develops, taking into account knowledge gained at prior iterations

- **Step-wise Regression:** The iterative process of building a regression model step-by-step, including choosing independent variables to be included in the final model
- **Stratified Sampling:** A kind of sampling procedure in which the entire population is split up into smaller groups to finish the sampling process
- **Systematic Error:** The expected value of the overall error
- **Test-Retest Reliability:** A reliability score that is achieved by administering the same test to a set of individuals twice over time
- **Two-Tailed Test:** A kind of hypothesis testing where the null hypothesis is disproved if the observed sample statistic deviates significantly from the critical value in either direction (higher than the positive critical value or lower than the negative critical value)
- **MATLAB:** A platform for programming and numerical computing used by millions of engineers and academics to interpret data, construct algorithms, and make predictions
- **Logic programming:** A programming methodology in which calculation is performed using a knowledge base of facts and regulations. **LISP** and **Prolog** are two logic programming languages employed in the development of Artificial Intelligence
- **H2O.ai:** A powerful AI cloud platform designed to build, operate and innovate with **AI** in any environment easier and faster

Tidyverse

A comprehensive set of **R packages** for exploratory data analysis, data visualization and data manipulation that corresponds to a single design philosophy

- **Complex structured data:** Data that cannot be easily processed by structured query languages and tools because they are made up of two or more intricate, interrelated parts
- **Key-value Database:** A kind of non-relational database that stores data via the simple key-value approach
- **Topological Data Analysis:** A method that focuses on the structure of complex data to find clusters and any statistical significance that may be present

- **Data Catalogue:** A technique for organizing, categorizing, and documenting collections of data in a data repository that simplifies searching and, frequently, also includes features that enable users to review or remark on data-sets, benefiting future users' queries
- **Data Democratization:** A procedure for increasing Data accessibility within an organisation, especially for non-specialist staff
- **Operating Model:** An organization's operating model that describes how an organization operates, including the tasks it does, how they are carried out, and the individuals, procedures, and technology involved
- **Data Owner:** The individual responsible for an organization's classification, protection, utilization, and quality of one or more data sets
- **Data Processor:** An individual, corporation, or other entity that handles personal data on behalf of the data controller
- **Data Education:** Inform individuals about the utilization of data and its significance to the enterprise organisation
- **Embedded BI:** The incorporation of data visualizations, dashboards, and reports within an application
- **Management Information (MI):** Any information obtained to support business decisions
- **Single Version of the Truth:** The process of providing decision-makers with precise data in the form of responses to extremely strategic questions
- **Single Source of Truth:** A technique for organizing information models and related conceptual frameworks so that each piece of data is recorded exactly once
- **Analysis Facility:** A bundled-up collection of the tools and data required to perform a certain set of analysis
- **Chart (Graph):** A method for organizing and displaying data with the ultimate goal of making it simpler to interpret and, in particular, identify patterns
- **Clickstream Analytics:** The monitoring and evaluation of website traffic in terms of page views, unique visitors, and repeat visitors
- **Data Asset:** A program output file, document, report, database, or webpage that business organizations utilize to make money

- **Data Controller:** An organization responsible for the gathering, storing, using, and erasing of personal data
- **Data Issue Management (DIM):** A method of eliminating or lessening the effects of barriers that limit the efficient use of data
- **Data Maturity:** A way to evaluate how effective an enterprise organization's data analysis is
- **Data Protection Officer (DPO):** A senior manager in charge of making sure that his/her company complies with the relevant data protection laws when handling the personal information of its employees, clients, vendors, or any other individuals (also known as **data subjects**)
- **Data Roadmap:** A broad vision or strategy for implementing a **Data Strategy**
- **Data Sourcing:** The procedure of obtaining Data from multiple Data Sources, including Data Repositories, Databases supporting Transactional Systems, website logs or other flat-file outputs, APIs, or external sources of data
- **End User Computing (EUC):** Platforms and computer systems that assist non-programmers in developing applications
- **Flat-file:** A kind of database that saves information in plain text
- **Operational Repository:** A Data Repository that is primarily concerned with facilitating the generation of Information and is an important component of a modern Data Architecture
- **Reference Data:** Data that is used to describe and categorize other data
- **Gamification:** The design principles employed in the game to boost consumer interest in non-game enterprises
- **Apache Sqoop:** A tool developed to move data between relational database servers and Hadoop
- **Weather Data:** The data patterns and trends used to monitor the atmosphere
- **Explanation-based learning:** A machine learning method that allows intelligent systems to learn from examples

- **Intelligent matching:** A method of managing data in which a database's contents are identified, indexed, and retrieved using a variety of artificial intelligence-based algorithms for data matching and sorting.
- **Expressiveness:** The ability of a knowledge representation framework to communicate the key characteristics of a given problem
- **Parser:** A program that determines one or more possible ways a sentence in a language could have been constructed using the grammar rules
- **Problem description:** A summary of a problem that needs to be solved
- **Problem reduction search:** A method of problem-solving that involves iteratively solving simpler subproblems first, then tackling more complex problems by breaking them down into simpler ones
- **Symbolic manipulation:** Processing symbolic data to create new symbolic structures or derived conclusions
- **Theorem proving:** The method of systematically deriving new formula using rules of reasoning in order to prove a mathematical theorem or solve a predicate calculus challenge
- **Trigram:** A sequence of three words
- **Uncertain reasoning:** Reasoning under uncertainty
- **Weak method:** A method of problem-solving that can be used to solve a range of issues but is not ensured to be effective or yield a solution
- **Decision Aid:** A software that facilitates decision-making by individuals, especially for complicated issues where a high level of knowledge is required
- **Human-Centered Computing:** Computers and other devices should be constructed to effectively meet user needs and expectations
- **Plan Recognition:** Prediction of an interactive system's top-level strategies from its observed actions
- **Programming by Demonstration:** A set of end-user programming methods that create code from examples that the user has provided
- **Toy System:** Small-scale use of an idea or methodology that works well for testing a few key features but is inappropriate for solving difficult or real-world issues

Black Box Algorithms:

An algorithm that has a black box structure prevents the user from seeing how it functions within. The output of this algorithm is determined by correlating specific data attributes across millions of input data points.

Cybernetic Organism:

A hybrid of human and machine

Friendly Artificial Intelligence:

A Superintelligence that would benefit people, at the very least supports their interests, or otherwise work to advance the humankind

Unfriendly Artificial Intelligence:

Superintelligence with an objective that makes it useful for the AI to damage mankind and it could represent an existential risk to humanity

Weight: The degree to which two neurons from two different layers of a neural network are connected

"First, solve the problem. Then, write the code. "

– John Johnson

Big data

A collection of technologies and strategies needed to handle, evaluate, and preserve the large amount of data. Big data operates under the assumption that the more one know about a topic, the more accurately one can learn new things and make predictions about what's to come. Today, it is applied in a variety of fields, including agriculture, environmental conservation, medicine, and sports.

Space analysis:

Analyzing, evaluating, and modeling geographical data features, such as locations, traits, and their relationships, which expose the geometrical or geographical properties of data, in order to understand entities.

Administrative data:

Data generally gathered for administrative, not research, objectives.

Null Hypothesis	Research Hypothesis
There is no statistically significant difference between the sample means of two groups	There is statistically significant difference between the sample means of two groups

A hypothesis or idea supported by insufficient data. For confirmation, more research and testing are needed. A hypothesis may be found to be true or untrue through more testing and experiments. It means that it might turn out to be false or true.

Analogue data

↓

Any information that is physically captured, whether it be through electrical or optical means, and stored

Anomaly:

A practice, procedure, or observation that deviates from what is common or usual

At-risk data: Information that could be lost.

Black box: Any device whose operations and functionality the user cannot access or understand.

Data Catalogue → A curated set of metadata regarding datasets and their data components.

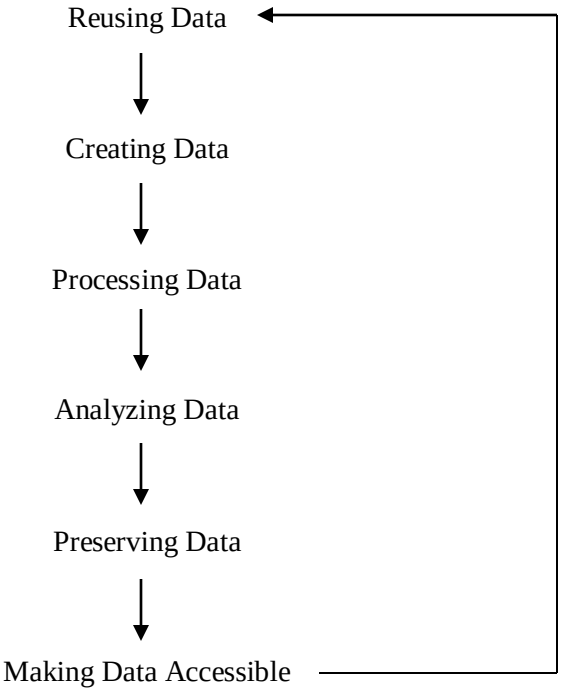
Data dredging

↓

Using a data collection to learn more information than it actually contains

- **Derived research data:** Research data that are produced by merging or analyzing 'raw' data, frequently reproducible but intensive.
- **E-Infrastructure:** A system that brings together communications, computation, and digital technologies to promote teamwork and research.
- **Experimental research data:** Research findings from experiments that is frequently reproducible but intensive.
- **Informatics:** The discipline of gathering, classifying, storing, retrieving, and sharing information.

Research Data Lifecycle



Simulation research data:

Research data produced by test models, where the model and metadata may be more significant than the model's output data.

Stream processing

The process of performing calculations on individual data elements as they pass through a system

Secondary Data:

Existing data that is being used for a different objective from the one for which it was originally acquired

Data quality checklist

A list of potential problems with a dataset

Data management plan:

A clear description outlining the management and documentation of research data throughout a project as well as the conditions for the data's future deposit with a data repository for long-term management and preservation

FAIR Data Principles

A group of researchers and research organizations came up with the **FAIR Data Principles** (Findable, Accessible, Interoperable, and Reusable), which were published in Scientific Data in 2016 and serve as a set of guiding principles for the reuse of digital assets.

Digitisation	Digitalisation
Information that is being transferred from a physical format to a digital one	The method of utilizing digitization to enhance business procedures

AI Literacy:

Knowledge of what artificial intelligence is and what it needs to function. With the aid of this knowledge, it is feasible to assess AI technologies critically, interact with AI systems, work with them, and use them as useful tools for one's own work.

Customer analytics:

The analysis of a certain consumer base based on data. This makes it possible to identify valuable client relationships and consumer interests in order to develop appropriate, unique offers

Datafication:

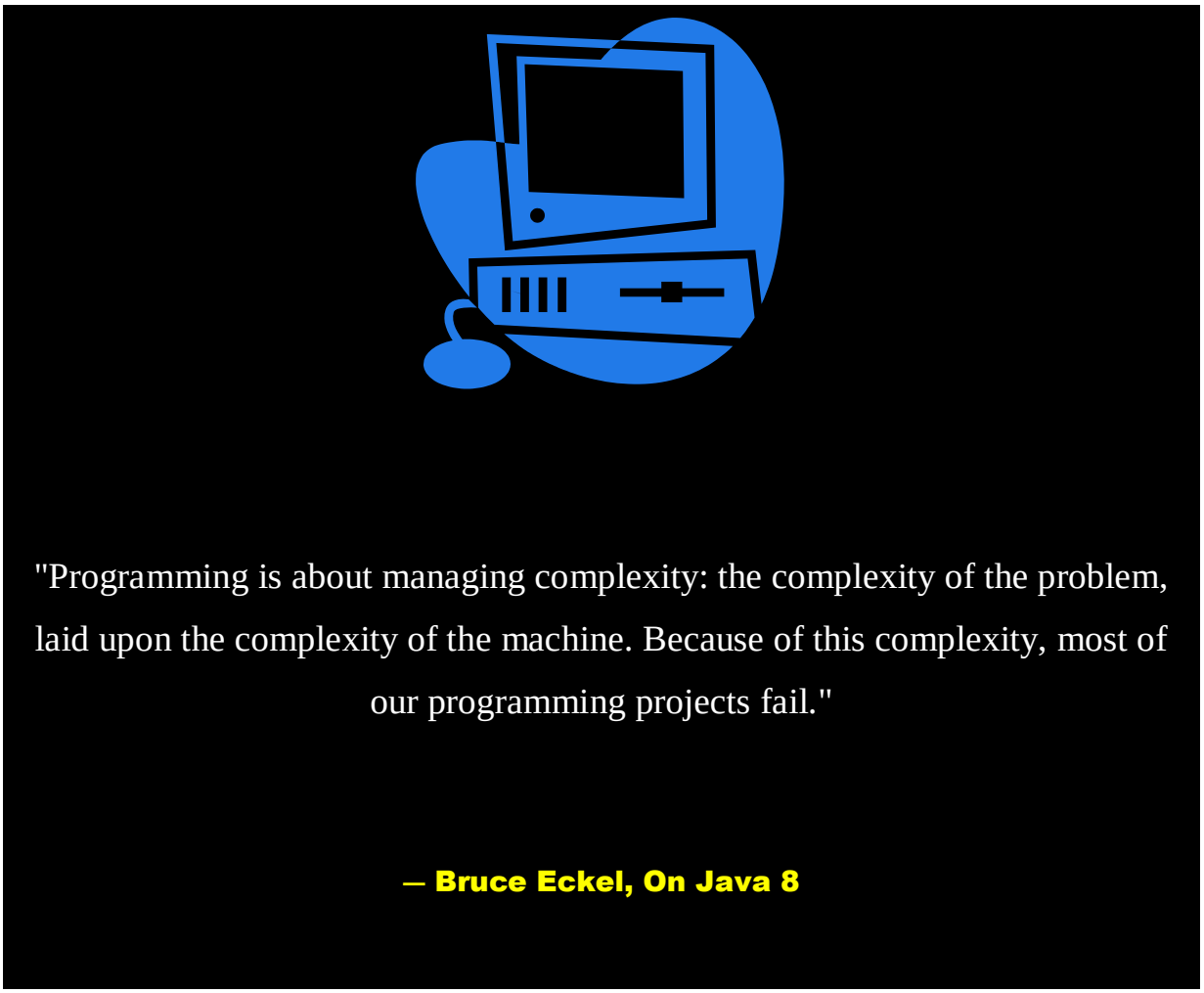
Explains the creation of data that results from the digitization of everyday activities and aspects of existence

Data Thinking

To successfully create data products and data strategies, data thinking unites data science and design thinking methodologies. Data Thinking takes into account and addresses issues from the user's perspective.

- **Keras**: An open source library. Rapid neural network implementation is accomplished using Keras. It is frequently a crucial building piece in deep learning applications.
- **Matplotlib:** A library for the Python programming language. It is very suitable for data visualization.
- **Pandas:** A Python programming language application package that enables the data to be managed, cleansed, and analyzed. Pandas' major feature, Data Frames, represents data as a table with columns and rows.
- **Predictive Maintenance:** A maintenance procedure used to reduce the "downtime" of manufacturing machinery. By assessing the likelihood of system breakdowns, it seeks to optimize maintenance cycles.
- **PySpark:** The Python API for Apache Spark, a free and open-source framework and collection of modules for processing massive amounts of data quickly and efficiently.
- **Trend analysis:** A method used in technical analysis that makes predictions about future stock price movements using trend data that has just been observed. To predict the long-

term direction of economic outlook, trend analysis makes use of previous data, such as price fluctuations and transaction volume.



"Programming is about managing complexity: the complexity of the problem, laid upon the complexity of the machine. Because of this complexity, most of our programming projects fail."

— Bruce Eckel, On Java 8

Best Linux Books that Every Superuser Should Read:

- **How Linux Works: What Every Superuser Should Know**

  Book by Brian Ward

- **The Linux Programming Interface**

  Book by Michael Kerrisk

- **Linux pocket guide**

  Book by Daniel J. Barrett

- **Linux for Beginners**

  Book by Jason Cannon

- **How Linux Works: What Every Superuser Should Know**

  Book by Brian Ward

- **Linux Kernel Development**

  Book by Robert Love

- **Linux: The Complete Reference**

  Book by Richard Petersen

- **Linux in a Nutshell**

  Book by Ellen Siever and Robert Love

- **Linux Basics for Hackers: Getting Started with Networking, Scripting, and Security in Kali**

  Book by OccupyTheWeb

- **Linux Command Line and Shell Scripting Bible**

  Book by Christine Bresnahan and Richard BLUM

- **Linux Administration: The Linux Operating System and Command Line Guide for Linux Administrators**

  Book by Jason Cannon

- **The Art of Unix Programming**

  Book by Eric S. Raymond

- **The Linux Command Line, 2nd Edition: A Complete Introduction**

  Book by William Shotts

- **Linux Bible**

  Book by Christopher Negus

- **Linux System Programming: Talking Directly to the Kernel and C Library**

  Book by Robert Love

- **A Practical Guide to Linux Commands, Editors, and Shell Programming**

  Book by Mark G. Sobell

- **Linux for Beginners and Command Line Kung Fu**

  Book by Jason Cannon

- **Linux Device Drivers**

  Book by Alessandro Rubini, Greg Kroah-Hartman, and Jonathan Corbet

- **Advanced Linux programming**

  Book by Alex Samuel, Jeffrey Oldham, and Mark Mitchell

- **Understanding the Linux Kernel**

Book by Daniel Pierre Bovet and Marco Cesati

- **Learn Linux Quickly: A Beginner-friendly Guide to Getting Up and Running with the World's Most Powerful Operating System**

  Book by Ahmed Alkabary

- **Linux administration**

  Book by Wale Soyinka

- **Linux For Dummies**

  Book by Richard Blum

- **Linux Essentials**

  Book by Christine Bresnahan and Richard BLUM

- **The Linux Command Line Beginner's Guide**

  Book by Jonathan Moeller

- **Linux All-in-One for Dummies**

  Book by Emmett Dulaney

- **Learning the bash Shell**

  Book by Cameron Newham

- **Linux for Developers: Jumpstart Your Linux Programming Skills**

  Book by William "Bo" Rothwell

- **Lfm: Linux Field Manual**

  Book by Tim Bryant

- **CompTIA Linux+ Study Guide: Exam XK0-005**

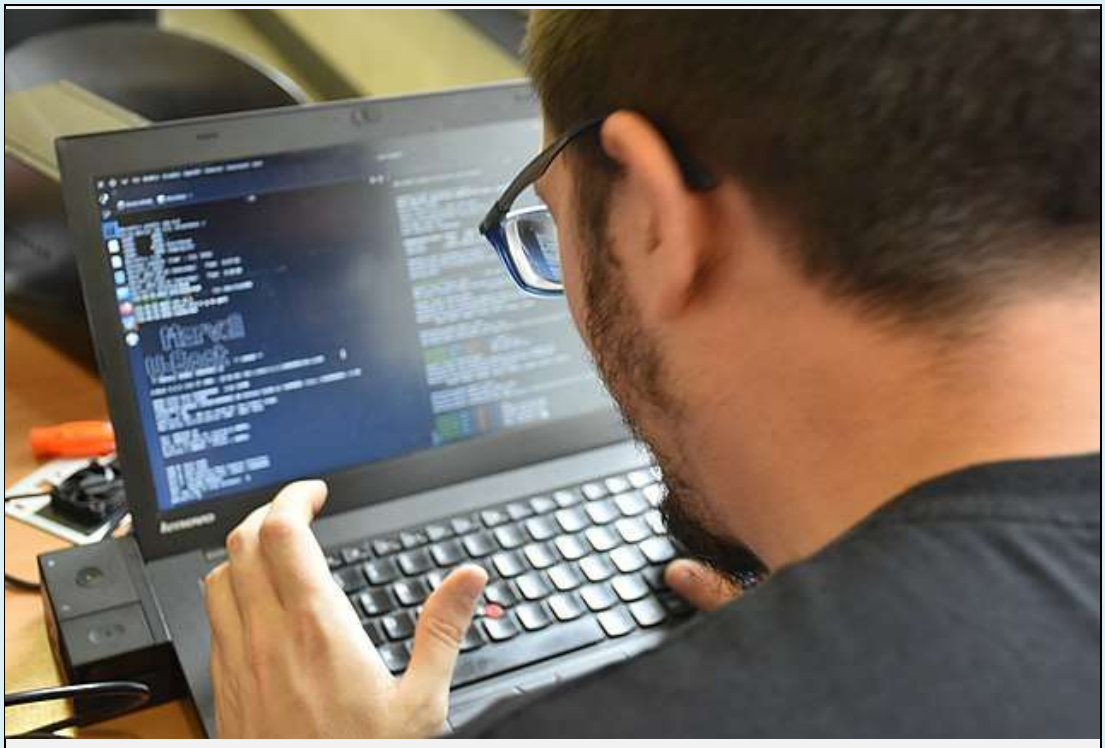
  Book by Christine Bresnahan and Richard BLUM

- **sed & awk**

  Book by Arnold Robbins and Dale Dougherty

- **Linux From Scratch**

  Book by Gerard Beekmans



Linux is a complex example of the wisdom of crowds. It's a good example in the sense that it shows you can set people to work in a decentralized way - that is, without anyone really directing their efforts in a particular direction - and still trust that they're going to come up with good answers.

– James Surowiecki

Best Programming Books that Every Programmer Should Read:

C:

- **The C Programming Language**

  Book by Brian Kernighan and Dennis Ritchie

- **C Programming Absolute Beginner's Guide**

  Book by Dean Miller and Greg Perry

- **Head First C**

  Book by David Griffiths and Dawn Griffiths

- **Expert C Programming**

  Book by Peter van der Linden

- **C Programming: A Modern Approach**

  Book by K. N King

- **C: The complete reference**

  Book by Herbert Schildt

- **Learn C the Hard Way: Practical Exercises on the Computational Subjects You Keep Avoiding (Like C)**

  Book by Zed Shaw

- **C in a Nutshell: The Definitive Reference**

  Book by Peter Prinz and Tony Crawford

- **C Programming In Easy Steps**

  Book by Mike McGrath

- **Computer Fundamentals and Programming in C**

  Book by Reema Thareja

- **Hands-On Network Programming with C: Learn Socket Programming in C and Write Secure and Optimized Network Code**

  Book by Lewis Van Winkle

- **Let Us C**

  Book by Yashavant Kanetkar

- **Low-Level Programming: C, Assembly, and Program Execution on Intel® 64 Architecture**

  Book by Igor Zhirkov

- **Effective C: An Introduction to Professional C Programming**

  Book by Robert C. Seacord

- **Data Structures Using C**

  Book by Reema Thareja

- **A Book on C: Programming in C**

  Book by Al Kelley and Ira Pohl

- **C Traps and Pitfalls**

  Book by Andrew Koenig

- **C Interfaces and Implementations: Techniques for Creating Reusable Software**

  Book by David Hanson and David R. Hanson

- **Mastering algorithms with C**

  Book by Kyle Loudon

- **Extreme C: Taking You to the Limit in Concurrency, OOP, and the Most Advanced Capabilities of C**

Book by Kamran Amini

- **Let Us C Solutions**

  Book by Yashavant Kanetkar

- **Bare Metal C: Embedded Programming for the Real World**

  Book by Stephen Oualline

- **Introduction to C Programming**

  Book by Reema Thareja

- **C - In Depth**

  Book by Deepali Srivastava

- **C How to Program**

  Book by Harvey Deitel and Paul Deitel

- **The C answer book**

  Book by Clovis L. Tondo

- **C Programming For Dummies**

  Book by Dan Gookin

- **Understanding Pointers In C & C++: Fully Working Examples and Applications of Pointers**

  Book by Yashavant Kanetkar

C++:

- **The C++ Programming Language**

  Book by Bjarne Stroustrup

- **Effective Modern C++**

Book by Scott Meyers

- **C++ Primer**

  Book by Barbara E. Moo, Josée Lajoie, and Stanley B. Lippman

- **A Tour of C++**

  Book by Bjarne Stroustrup

- **Programming: Principles and Practice Using C++**

  Book by Bjarne Stroustrup

- **C++ Concurrency in Action**

  Book by Anthony Williams

- **Modern C++ Design**

  Book by Andrei Alexandrescu

- **More Effective C++: 35 New Ways To Improve Your Programs And Designs**

  Book by Scott Meyers

- **Accelerated C++: Practical Programming by Example**

  Book by Andrew Koenig and Barbara E. Moo

- **C++ Templates: The Complete Guide**

  Book by David Vandevoorde, Douglas Gregor, and Nicolai M. Josuttis

- **More Exceptional C++: 40 New Engineering Puzzles, Programming Problems, and Solutions**

  Book by Herb Sutter

- **C++ coding standards**

  Book by Andrei Alexandrescu and Herb Sutter

- **C++ Template Metaprogramming: Concepts, Tools, and Techniques from Boost and Beyond**

  Book by Aleksey Gurtovoy and David Abrahams

- **Beginning C++ Game Programming**

  Book by Michael Dawson

- **Beginning C++ Programming**

  Book by Richard Grimes

- **C++ Programming**

  Book by D. S. Malik

- **Sams Teach Yourself C++ in One Hour a Day**

  Book by Rao Siddhartha

- **Professional C++**

  Book by Nicholas A. Solter and Scott J. Kleper

- **Starting Out with C++**

  Book by Tony Gaddis

- **C++: The Complete Reference**

  Book by Herbert Schildt

- **Exceptional C++: 47 Engineering Puzzles, Programming Problems, and Solutions**

  Book by Herb Sutter

- **C++ High Performance: Master the Art of Optimizing the Functioning of Your C++ Code**

  Book by Björn Andrist and Viktor Sehr

- **The C++ Standard Library: A Tutorial and Reference**

  Book by Nicolai M. Josuttis

- **Effective C++ Digital Collection: 140 Ways to Improve Your Programming**

  Book by Scott Meyers

- **C++17 - The Complete Guide**

  Book by Nicolai M. Josuttis

- **C++ Primer Plus**

  Book by Stephen Prata

- **Data Parallel C++: Mastering DPC++ for Programming of Heterogeneous Systems using C++ and SYCL**

  Book by James Brodman, Michael Kinsner, Xinmin Tian, Ben Ashbaugh, John Pennycook, James Reinders

- **Data Structures Using C++**

  Book by Varsha H. Patil

- **The Design and Evolution of C++**

  Book by Bjarne Stroustrup

- **Modern CMake for C++: Discover a Better Approach to Building, Testing, and Packaging Your Software**

  Book by Rafał Świdziński

- **Expert C++: Become a Proficient Programmer by Learning Coding Best Practices with C++17 and C++20's Latest Features**

  Book by Shunguang Wu and Vardan Grigoryan

- **Learn C++ Quickly: A Complete Beginner's Guide to Learning C++, Even If You're New to Programming**

  Book by Code Quickly

- **Memory Management Algorithms and Implementation in C/C++**

  Book by Bill Blunden

- **Thinking in C++**

  Book by Bruce Eckel

- **Modern C++ Programming Cookbook**

  Book by Marius Bancila

- **Hands-On Design Patterns with C++: Solve Common C++ Problems with Modern Design Patterns and Build Robust Applications**

  Book by Fedor G. Pikus

- **A complete guide to programming in C++**

  Book by Ulla Kirch-Prinz

- **C++ Crash Course: A Fast-Paced Introduction**

  Book by Josh Lospinoso

- **C++ For Dummies**

  Book by Stephen Randy Davis

JAVA:

- **Head First Java**

  Book by Bert Bates and Kathy Sierra

- **Effective Java**

  Book by Joshua Bloch

- **Thinking in Java**

Book by Bruce Eckel

- **Java Concurrency in Practice**

  Book by Brian Goetz

- **Core Java**

  Book by Cay S. Horstmann and Gary Cornell

- **Java: A Beginner's Guide**

  Book by Herbert Schildt

- **Java: The Complete Reference**

  Book by Herbert Schildt

- **Java 8 in Action: Lambdas, streams, and functional-style programming**

  Book by Alan Mycroft and Mario Fusco

- **Beginning Programming With Java for Dummies**

  Book by Barry A. Burd

- **Learn Java in One Day and Learn It Well**

  Book by Jamie Chan

- **Modern Java in Action: Lambdas, Streams, Functional and Reactive Programming**

  Book by Alan Mycroft, Mario Fusco, and Raoul-Gabriel Urma

- **Test Driven: Practical TDD and Acceptance TDD for Java Developers**

  Book by Lasse Koskela

- **Java Puzzlers: Traps, Pitfalls, and Corner Cases**

  Book by Joshua Bloch and Neal Gafter

- **Spring in Action**

  Book by Craig Walls and Ryan Breidenbach

- **Java generics and collections**

  Book by Maurice Naftalin

- **Core Java Volume I−Fundamentals**

  Book by Cay S. Horstmann

- **Think Java: How to think like a computer scientist**

  Book by Allen B. Downey

- **Java SE 8 for the Really Impatient**

  Book by Cay S. Horstmann

- **Head First Object-Oriented Analysis and Design: A Brain Friendly Guide to OOA&D**

  Book by Brett McLaughlin

- **Learning Java: An Introduction to Real-World Programming with Java**

  Book by Daniel Leuck, Marc Loy, and Patrick Niemeyer

- **Core Java for the Impatient**

  Book by Cay S. Horstmann

- **Java By Comparison: Become a Java Craftsman in 70 Examples**

  Book by Jorg Lenhard, Linus Dietz, and Simon Harrer

- **High-Performance Java Persistence**

  Book by Vlad Mihalcea

- **Java in a Nutshell: A Desktop Quick Reference**

  Book by Benjamin Evans and David Flanagan

- **Spring Microservices in Action**

  Book by Edward John Carnell

- **Optimizing Java: Practical Techniques for Improving JVM Application Performance**

  Book by Benjamin Evans, Chris Newland, and James Gough

- **Java Performance: The Definitive Guide: Getting the Most Out of Your Code**

  Book by Scott Oaks

- **Microservices Patterns: With Examples in Java**

  Book by Chris Richardson

- **Functional Programming in Java: Harnessing the Power of Java 8 Lambda Expressions**

  Book by Venkat Subramaniam

- **Mastering Java Machine Learning**

  Book by Krishna Choppella and Uday Kamath

- **OCA: Oracle Certified Associate Java SE 8 Programmer I Study Guide: Exam 1Z0-808**

  Book by Jeanne Boyarsky and Scott Selikoff

- **Let Us Java: Strong Foundation For Java Programming**

  Book by Yashavant Kanetkar

- **Learn Java: A Crash Course Guide to Learn Java in 1 Week**

  Book by Timothy C. Needham

- **Java Performance**

  Book by Binu John and Charlie Hunt

- **The Java Language Specification**

  Book by Oracle Corporation

- **Elements of Programming Interviews in Java: The Insider's Guide**

  Book by Adnan Aziz, Amit Prakash, and Tsung-Hsien Lee

PYTHON:

- **Fluent Python**

  Book by Luciano Ramalho

- **Learn Python the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code**

  Book by Zed Shaw

- **Python Cookbook: Recipes for Mastering Python 3**

  Book by Brian K. Jones and David M. Beazley

- **Automate the Boring Stuff with Python: Practical Programming for Total Beginners**

  Book by Al Sweigart

- **Python Crash Course: A Hands-On, Project-Based Introduction to Programming**

  Book by Eric Matthes

- **Head First Python**

  Book by Paul Barry

- **Programming Python**

  Book by Mark Lutz

- **Think Python: An Introduction to Software Design**

  Book by Allen B. Downey

- **Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython**

  Book by Wes McKinney

- **Python Tricks: A Buffet of Awesome Python Features**

  Book by Dan Bader

- **Learning Python**

  Book by Mark Lutz

- **Introduction to Machine Learning with Python: A Guide for Data Scientists**

  Book by Andreas C. Müller and Sarah Guido

- **Learning Python: Powerful Object-Oriented Programming**

  Book by Mark Lutz

- **Python for Everybody: Exploring Data Using Python 3**

  Book by Charles Severance

- **Python for Kids: A Playful Introduction To Programming**

  Book by Jason R. Briggs

- **Python Programming: An Introduction to Computer Science**

  Book by John M. Zelle

- **Learn Python in One Day and Learn It Well: Python for Beginners with Hands-on Project**

  Book by Jamie Chan

- **Invent Your Own Computer Games with Python**

  Book by Al Sweigart

- **Python Data Science Handbook: Essential Tools for Working with Data**

  Book by Jake VanderPlas

- **Effective Python: 90 Specific Ways to Write Better Python**

  Book by Brett Slatkin

- **Python Pocket Reference: Python In Your Pocket**

  Book by Mark Lutz

- **Python 3 Object Oriented Programming**

  Book by Dusty Phillips

- **How to think like a computer scientist: Learning with Python**

  Book by Allen B. Downey

- **Python in a nutshell**

  Book by Alex Martelli

- **Natural Language Processing with Python**

  Book by Edward Loper, Ewan Klein, and Steven Bird

- **The Big Book of Small Python Projects: 81 Easy Practice Programs**

  Book by Al Sweigart

- **Python Programming for the Absolute Beginner**

  Book by Michael Dawson

- **Python Essential Reference**

  Book by David M. Beazley

- **Deep Learning with Python**

  Book by François Chollet

- **Dive into Python**

  Book by Mark Pilgrim

- **Impractical Python: Projects Playful Programming Activities to Make You Smarter**

  Book by Lee Vaughan

- **Object-Oriented Python: Master OOP by Building Games and GUIs**

  Book by Irv Kalb

- **The Python Bible 7 in 1: Volumes One To Seven (Beginner, Intermediate, Data Science, Machine Learning, Finance, Neural Networks, Computer Vision)**

  Book by Florian Dedov

- **Coding for Kids: Python: Learn to Code with 50 Awesome Games and Activities**

  Book by Adrienne Tacke

- **Learn More Python 3 the Hard Way: The Next Step for New Python Programmers**

  Book by Zed Shaw

- **Artificial Intelligence with Python**

  Book by Prateek Joshi

- **Python Programming: Using Problem Solving Approach**

  Book by Reema Thareja

- **Django for Beginners: Build websites with Python and Django**

  Book by William S. Vincent

- **Python for Dummies**

  Book by Aahz Maruch and Stef Maruch

- **Beginning Programming with Python For Dummies**

  Book by John Paul Mueller

HTML:

- **Learning Web Design**

  Book by Jennifer Niederst Robbins

- **Head First HTML with CSS and XHTML**

  Book by Elisabeth Robson and Eric Freeman

- **HTML5: The Missing Manual**

  Book by Matthew MacDonald

- **Learn HTML for Beginners: The Illustrated Guide to Coding**

  Book by Jo Foster

- **HTML5 for Web Designers**

  Book by Jeremy Keith

- **HTML 4 for the World Wide Web**

  Book by Elizabeth Castro

- **Introducing HTML5**

  Book by Bruce Lawson and Remy Sharp

- **Core HTML5 Canvas: Graphics, Animation, and Game Development**

  Book by David M. Geary

- **The Definitive Guide to HTML5**

  Book by Adam Freeman

- **Html: Basic Fundamental Guide for Beginners**

  Textbook by M. G. Martin

- **HTML5 in Easy Steps**

  Book by Mike McGrath

CSS:

- **CSS Pocket Reference: Visual Presentation for the Web**

  Book by Eric A. Meyer

- **CSS Secrets: Better Solutions to Everyday Web Design Problems**

  Book by Lea Verou

- **CSS: The Missing Manual**

  Book by David McFarland

- **CSS in Depth**

  Book by Keith Grant

- **CSS mastery**

  Book by Andy Budd

- **CSS: The Definitive Guide: Visual Presentation for the Web**

  Book by Eric A. Meyer and Estelle Weyl

- **CSS Visual Dictionary**

Book by Greg Sidelnikov

- **Cascading Style Sheets: The Definitive Guide**

  Book by Eric A. Meyer

- **The Book of CSS3: A Developer's Guide to the Future of Web Design**

  Book by Peter Gasston

- **CSS3: The Missing Manual**

  Book by David Sawyer McFarland

- **Learn CSS in One Day and Learn It Well: CSS for Beginners With Hands-On Project**

  Book by Jamie Chan

- **CSS Master**

  Book by Tiffany A. Brown

- **Basics of Web Design: HTML5 and CSS**

  Textbook by Terry A. Felke-Morris

- **CSS for Babies**

  Book by John Vanden-Heuvel

- **CSS in Easy Steps**

  Book by Mike McGrath

JAVASCRIPT:

- **JavaScript: The Good Parts**

  Book by Douglas Crockford

- **Eloquent JavaScript: A Modern Introduction to Programming**

  Book by Marijn Haverbeke

- **JavaScript and JQuery: Interactive Front-End Web Development**

  Book by Jon Duckett

- **You Don't Know JS: Scope and Closures**

  Book by Kyle Simpson

- **A Smarter Way to Learn JavaScript**

  Book by Mark Myers

- **Effective JavaScript : 68 specific ways to harness the power of JavaScript**

  Book by David Herman

- **Head First JavaScript Programming: A Brain-Friendly Guide**

  Book by Elisabeth Robson and Eric Freeman

- **JavaScript: The Definitive Guide: Master the World's Most-Used Programming Language**

  Book by David Flanagan

- **Learn JavaScript Visually: With Interactive Exercises**

  Book by Ivelin Demirov

- **The Principles of Object-Oriented JavaScript**

  Book by Nicholas C. Zakas

- **Professional JavaScript for Web Developers**

  Book by Nicholas C. Zakas

- **Speaking JavaScript: An In-Depth Guide for Programmers**

  Book by Axel Rauschmayer

- **Programming JavaScript Applications: Robust Web Architecture with Node, HTML5, and Modern JS Libraries**

  Book by Eric Elliott

- **JavaScript Enlightenment**

  Book by Cody Lindley

- **Learning JavaScript Design Patterns**

  Book by Addy Osmani

- **Secrets of the JavaScript Ninja**

  Book by Bear Bibeault, John Resig and Josip Maras

- **Beginning JavaScript**

  Book by Jeremy McPeak and Paul Wilton

- **JavaScript Patterns**

  Book by Stoyan Stefanov

- **Understanding ECMAScript 6: The Definitive Guide for JavaScript Developers**

  Book by Nicholas C. Zakas

- **JavaScript for Kids: A Playful Introduction to Programming**

  Book by Nick Morgan

- **High Performance JavaScript**

  Book by Nicholas C. Zakas

- **Object-Oriented JavaScript**

  Book by Kumar Chetan Sharma and Stoyan Stefanov

- **Maintainable JavaScript**

Book by Nicholas C. Zakas

- **Secrets of the JavaScript Ninja**

  Book by Bear Bibeault and John Resig

- **Coding with JavaScript For Dummies**

  Book by Chris Minnick and Eva Holland

- **DOM Enlightenment**

  Book by Cody Lindley

- **JavaScript from Beginner to Professional: Learn JavaScript Quickly by Building Fun, Interactive, and Dynamic Web Apps, Games, and Pages**

  Book by Laurence Lars Svekis and Rob Percival

- **Testable JavaScript**

  Book by Mark Ethan Trostler

- **You Don't Know JS: Up and Going**

  Book by Kyle Simpson

- **JavaScript Allongé: A strong cup of functions, objects, combinators, and decorators**

  Book by Reginald Braithwaite

- **You Don't Know JS Yet: Get Started**

  Book by Kyle Simpson

- **JavaScript and JQuery: The Missing Manual**

  Book by David Sawyer McFarland

- **The Recursive Book of Recursion: Ace the Coding Interview with Python and JavaScript**

  Book by Al Sweigart

- **The Road to Learn React: Your Journey to Master Plain Yet Pragmatic React. Js**

  Book by Robin Wieruch

- **Javascript In Easy Steps**

  Book by Mike McGrath

- **Building JavaScript Games: For Phones, Tablets, and Desktop**

  Book by Arjan Egges

- **High Performance Browser Networking**

  Book by Ilya Grigorik

- **Learning TypeScript**

  Book by Josh Goldberg

- **Pro JavaScript Techniques**

  Book by John Resig

- **The Little Book on CoffeeScript**

  Book by Alex MacCaw

- **Learning React: Functional Web Development with React and Redux**

  Book by Alex Banks and Eve Porcello

- **Learn JavaScript Quickly: A Complete Beginner's Guide to Learning JavaScript, Even If You're New to Programming**

Book by Code Quickly

- **Test-Driven JavaScript Development**

  Book by Christian Johansen

- **Structure and Interpretation of Computer Programs, JavaScript Edition**

  Textbook by Gerald Jay Sussman and Hal Abelson

- **Functional Programming in JavaScript: How to Improve Your JavaScript Programs Using Functional Techniques**

  Book by Luis Atencio

- **Pro JavaScript Design Patterns**

  Book by Dustin Diaz and Ross Harmes

- **Java Script: The Complete Reference 2/E**

  Book by Thomas Powell

PHP:

- **The Joy of PHP: A Beginner's Guide to Programming Interactive Web Applications with PHP and MySQL**

  Book by Alan Forbes

- **Learning PHP, MySQL, JavaScript, CSS and HTML5: A Step-by-Step Guide to Creating Dynamic Websites**

  Book by Robin Nixon

- **Programming PHP**

Book by Rasmus Lerdorf

- **PHP and MySQL Web Development**

  Book by Laura Thomson and Luke Welling

- **Murach's PHP and MySQL**

  Book by Joel Murach and Ray Harris

- **Head First PHP and MySQL**

  Book by Lynn Beighley and Michael Morrison

- **PHP and MySQL: Novice to Ninja: Get Up to Speed With PHP the Easy Way**

  Book by Tom Butler and Kevin Yank

- **PHP and MySQL: Server-side Web Development**

  Book by Jon Duckett

- **Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5**

  Book by Robin Nixon

- **PHP and MySQL in easy steps, 2nd Edition: Updated to cover MySQL 8.0**

  Book by Mike McGrath

- **PHP: A Beginner's Guide**

  Book by Vikram Vaswani

- **Modern PHP: New Features and Good Practices**

  Book by Josh Lockhart

- **Laravel: Up & Running: A Framework for Building Modern PHP Apps**

Book by Matt Stauffer

- **Mastering PHP 7**

  Book by Branko Ajzele

- **PHP and MySQL**

  Book by Jon Duckett

- **PHP and MySQL: The Missing Manual**

  Book by Brett McLaughlin

- **PHP 8 Programming Tips, Tricks and Best Practices: A Practical Guide to PHP 8 Features, Usage Changes, and Advanced Programming Techniques**

  Book by Cal Evans and Doug Bierer

- **PHP for the Web: Visual QuickStart Guide**

  Book by Larry Ullman

- **PHP and MySQL for Dynamic Web Sites**

  Book by Larry Ullman

- **PHP: Learn PHP in One Day and Learn It Well. PHP for Beginners with Hands-on Project**

  Book by Jamie Chan

- **PHP 7: Real World Application Development**

  Book by Altaf Hussain, Branko Ajzele and Doug Bierer

- **PHP Cookbook**

  Book by Adam Trachtenberg and David Sklar

- **Full Stack Web Development For Beginners: Learn Ecommerce Web Development Using HTML5, CSS3, Bootstrap, JavaScript, MySQL, and PHP**

  Book by Riaz Ahmed

- **Drupal 9 Module Development: Get Up and Running with Building Powerful Drupal Modules and Applications**

  Book by Daniel Sipos

- **Building Web Apps with WordPress: WordPress as an Application Framework**

  Book by Brian Messenlehner and Jason Coleman

- **PHP and MySQL in easy steps**

  Book by Mike McGrath

- **WordPress Complete**

  Book by Karol Król

- **PHP, MySQL, and JavaScript All-in-One For Dummies**

  Book by Richard BLUM

- **PHP in Action: Objects, Design, Agility**

  Book by Chris Shiflett, Dagfinn Reiersol, and Marcus Baker

- **PHP Advanced and Object-Oriented Programming: Visual QuickPro Guide**

  Book by Larry Ullman

- **Learning PHP 7**

  Book by Antonio Lopez

- **PHP 7 Programming Cookbook**

  Book by Doug Bierer

- **PHP: The Complete Reference**

  Book by Steven Holzner

- **PHP and MySQL for Dummies**

  Book by Janet Valade

- **PHP Beginner's Practical Guide**

  Book by Pratiyush Guleria

- **Building RESTful Web Services with PHP 7**

  Book by Haafiz Waheed-ud-din Ahmad

- **Mastering PHP Design Patterns**

  Book by Junade Ali

- **PHP and MongoDB Web Development Beginner's Guide**

  Book by Rubayeet Islam

- **PHP 7 Data Structures and Algorithms**

  Book by Mizanur Rahman

- **Mastering The Faster Web with PHP, MySQL, and JavaScript: Develop State-of-the-art Web Applications Using the Latest Web Technologies**

  Textbook by Andrew Caya

- **Learning PHP: A Gentle Introduction to the Web's Most Popular Language**

Book by David Sklar

- **Learn PHP 8: Using MySQL, JavaScript, CSS3, and HTML5**

  Book by Steve Prettyman

- **PHP Microservices**

  Book by Carlos Perez Sanchez and Pablo Solar Vilarino

- **Professional CodeIgniter**

  Book by Thomas Myer

- **Symfony 5: The Fast Track**

  Book by Fabien Potencier

- **Practical PHP and MySQL Website Databases: A Simplified Approach**

  Book by Adrian W. West

- **PHP Programming for Beginners: Key Programming Concepts. How to use PHP with MySQL and Oracle databases (MySqli, PDO)**

  Book by Sergey Skudaev

ALGORITHMS:

- **Introduction to Algorithms**

  Book by T Cormen, C Leiserson, R Rivest, C Stein

- **The Algorithm Design Manual**

  Book by Steven Skiena

- **Grokking Algorithms: An Illustrated Guide for Programmers and Other Curious People**

Book by Aditya Bhargava

- **Algorithms**

  Book by Robert Sedgewick

- **Algorithm Design**

  Book by Jon Kleinberg and Éva Tardos

- **Algorithms Illuminated: Algorithms for NP-hard problems**

  Book by Tim Roughgarden

- **Algorithms in a nutshell**

  Book by George T. Heineman

- **Data structures and algorithms made easy in Java: data structure and algorithmic puzzles**

  Book by Narasimha karumanchi

- **A Common-Sense Guide to Data Structures and Algorithms: Level Up Your Core Programming Skills**

  Book by Jay Wengrow

- **Cracking the Coding Interview**

  Book by Gayle Laakmann McDowell

- **Guide to Competitive Programming: Learning and Improving Algorithms Through Contests**

  Book by Antti Laaksonen

- **Algorithms Unlocked**

  Book by Thomas H. Cormen

- **Problem Solving with Algorithms and Data Structures using Python**

  Book by Bradley N Miller and David L. Ranum

- **Computer Science Distilled: Learn the Art of Solving Computational Problems**

  Book by Wladston Ferreira Filho

- **Python Algorithms: Mastering Basic Algorithms in the Python Language**

  Book by Magnus Lie Hetland

- **Advanced Algorithms and Data Structures**

  Book by Marcello La Rocca

- **The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World**

  Book by Pedro Domingos

- **Dive Into Algorithms: A Pythonic Adventure for the Intrepid Beginner**

  Book by Bradford Tuckfield

- **Advanced Data Structures**

  Book by Peter Brass

- **Algorithmic Thinking: A Problem-Based Introduction**

  Book by Daniel Zingaro

- **Algorithms + Data Structures = Programs**

  Book by Niklaus Wirth

- **Algorithms Illuminated (Part 2): Graph Algorithms and Data Structures**

  Book by Tim Roughgarden

- **Introduction to the Design and Analysis of Algorithms**

  Book by Anany Levitin

- **Automate This: How Algorithms Came to Rule Our World**

  Book by Christopher Steiner

- **Think Like a Programmer: An Introduction to Creative Problem Solving**

  Book by V. Anton Spraul

- **Spies, Lies, and Algorithms: The History and Future of American Intelligence**

  Book by Amy Zegart

- **Algorithms Illuminated (Part 3): Greedy Algorithms and Dynamic Programming**

  Book by Tim Roughgarden

- **Algorithms for Decision Making**

  Book by Kyle H. Wray, Mykel J. Kochenderfer and Tim A. Wheeler

- **Fundamentals of Computer Algorithms**

  Book by Ellis Horowitz and Sartaj Sahni

- **Information Theory, Inference and Learning Algorithms**

  Book by David J. C. MacKay

- **9 Algorithms That Changed the Future**

  Book by John MacCormick

- **The Art of Computer Programming**

  Book by Donald Knuth

- **Data Structure and Algorithmic Thinking with Python**

  Book by Narasimha Karumanchi

- **A Common-Sense Guide to Data Structures and Algorithms**

  Book by Jay Wengrow

- **Hello World: How to be Human in the Age of the Machine**

  Book by Hannah Fry

- **The Design and Analysis of Computer Algorithms**

  Book by Alfred Aho, Jeffrey Ullman, and John Hopcroft

- **Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services**

  Book by Brendan Burns

- **A Programmer's Guide to Computer Science: A virtual degree for the self-taught developer**

  Book by William M Springer

- **Concrete Mathematics**

Textbook by Donald Knuth, Oren Patashnik and Ronald Graham

- **Understanding Cryptography: A Textbook for Students and Practitioners**

  Textbook by Christof Paar and Jan Pelzl

- **Algorithms For Dummies**

  Book by John Mueller and Luca Massaron

DATA STRUCTURES:

- **Data Structures and Algorithm Analysis in C++**

  Book by Clifford A Shaffer

- **Purely functional data structures**

  Book by Chris Okasaki

- **Open Data Structures: An Introduction**

  Book by Pat Morin

- **Think Data Structures: Algorithms and Information Retrieval in Java**

  Book by Allen B. Downey

- **Data Structures Using C**

  Book by Reema Thareja

- **Data Structures Through C in Depth**

  Book by Deepali Srivastava and Suresh Kumar Srivastava

- **C++ Plus Data Structures**

  Book by Nell B. Dale

- **JavaScript Data Structures and Algorithms: An Introduction to Understanding and Implementing Core Data Structure and Algorithm Fundamentals**

Book by Sammie Bae

- **Data structures using C and C++**

  Book by Yedidyah Langsam

- **Algorithms and Data Structures: The Basic Toolbox**

  Book by Kurt Mehlhorn and Peter Sanders

- **Data Structures and Algorithm Analysis in Java**

  Book by Clifford A Shaffer

- **Handbook of Data Structures and Applications**

  Book by Dinesh P. Mehta and Sartaj Sahni

- **Data structures with Java**

  Book by J. R Hubbard

- **Java Structures**

  Book by Duane A. Bailey

LINUX:

- **The Linux Command Line: A Complete Introduction**

  Book by William E. Shotts Jr. and William E. Shotts, Jr.

- **The Linux Programming Interface**

  Book by Michael Kerrisk

- **Linux bible**

  Book by Christopher Negus

- **Linux pocket guide**

Book by Daniel J. Barrett

- **Linux for Beginners**

  Book by Jason Cannon

- **How Linux Works: What Every Superuser Should Know**

  Book by Brian Ward

- **Linux Basics for Hackers: Getting Started with Networking, Scripting, and Security in Kali**

  Book by OccupyTheWeb

- **Linux Kernel Development**

  Book by Robert Love

- **Linux Administration: The Linux Operating System and Command Line Guide for Linux Administrators**

  Book by Jason Cannon

- **Linux: The Complete Reference**

  Book by Richard Petersen

- **Linux Command Line and Shell Scripting Bible**

  Book by Christine Bresnahan and Richard BLUM

- **Linux in a Nutshell**

  Book by Ellen Siever and Robert Love

- **The Art of Unix Programming**

  Book by Eric S. Raymond

- **Linux System Programming: Talking Directly to the Kernel and C Library**

  Book by Robert Love

- **A Practical Guide to Linux Commands, Editors, and Shell Programming**

  Book by Mark G. Sobell

- **Learn Linux Quickly: A Beginner-friendly Guide to Getting Up and Running with the World's Most Powerful Operating System**

  Book by Ahmed Alkabary

- **Linux for Beginners and Command Line Kung Fu**

  Book by Jason Cannon

- **The Ultimate Kali Linux Book: Perform Advanced Penetration Testing Using Nmap, Metasploit, Aircrack-Ng, and Empire**

  Book by Glen D. Singh

- **Understanding the Linux Kernel**

  Book by Daniel Pierre Bovet and Marco Cesati

- **Linux for Developers: Jumpstart Your Linux Programming Skills**

  Book by William "Bo" Rothwell

- **Learning the bash Shell**

  Book by Cameron Newham

- **Linux Device Drivers**

  Book by Alessandro Rubini, Greg Kroah-Hartman, and Jonathan Corbet

- **Mastering Linux Shell Scripting: A Practical Guide to Linux Command-line, Bash Scripting, and Shell Programming**

  Book by Andrew Mallett and Mokhtar Ebrahim

- **Efficient Linux at the Command Line**

  Book by Daniel J. Barrett

- **Shell Scripting: How to Automate Command Line Tasks Using Bash Scripting and Shell Programming**

  Book by Jaosn Cannon
- **Advanced Programming in the Unix Environment**

  Book by W. Richard Stevens
- **Lfm: Linux Field Manual**

  Book by Tim Bryant
- **CompTIA Linux+ Certification All-in-One Exam Guide**

  Book by Ted Jordan and Sandor Strohmayer
- **Advanced Linux programming**

  Book by Alex Samuel, Jeffrey Oldham and Mark Mitchell
- **Linux Kernel in a Nutshell**

  Book by Greg Kroah-Hartman
- **Linux Kernel Debugging: Leverage Proven Tools and Advanced Techniques to Effectively Debug Linux Kernels and Kernel Modules**

  Book by Kaiwan N Billimoria
- **Bash cookbook**

  Book by Carl Albing
- **Linux for Beginners: A Practical and Comprehensive Guide to Learn Linux Operating System and Master Linux Command Line.**

  Book by Ethem Mining
- **Beginning Linux Programming**

  Book by Mathew Neil and Richard Stones
- **The Linux Command Line Beginner's Guide**

  Book by Jonathan Moeller

- **Linux System Programming**

  Book by Robert Love

- **A Practical Guide to Fedora and Red Hat Enterprise Linux**

  Book by Mark G. Sobell

- **Kali Linux Hacking: A Complete Step by Step Guide to Learn the Fundamentals of Cyber Security, Hacking, and Penetration Testing**

  Book by Ethem Mining

- **Kali Linux Revealed: Mastering the Penetration Testing Distribution**

  Book by Jim O'Gorman, Mati Aharoni and Raphael Hertzog

- **Linux in Easy Steps**

  Book by Mike McGrath

- **Bash Pocket Reference: Help for Power Users and Sys Admins**

  Book by Arnold Robbins

- **Learn PowerShell in a Month of Lunches: Covers Windows, Linux, and macOS**

  Book by Travis Plunk, James Petty and Leon Leonhardt

- **Linux From Scratch**

  Book by Gerard Beekmans

- **Linux For Dummies**

  Book by Richard Blum

DATABASE:

- **Database Systems: The Complete Book**

  Book by Héctor García-Molina, Jeffrey Ullman and Jennifer Widom

- **Fundamentals of Database Systems**

Book by Ramez Elmasri

- **Database Design for Mere Mortals**

  Book by Michael J. Hernandez

- **An Introduction to Database Systems**

  Book by Christopher J. Date

- **SQL Antipatterns: Avoiding the Pitfalls of Database Programming**

  Book by Bill Karwin

- **Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems**

  Book by Martin Kleppmann

- **Database System Concepts**

  Book by Avi Silberschatz, Henry F. Korth and S. Sudarshan

- **Database Internals: A Deep Dive Into How Distributed Data Systems Work**

  Book by Alex Petrov

- **Beginning Database Design Solutions**

  Book by Rod Stephens

- **SQL Performance Explained**

  Book by Markus Winand

- **Database Management Systems**

  Book by Johannes Gehrke and Raghu Ramakrishnan

- **Oracle PL/SQL programming**

  Book by Scott Urman

- **Database in Depth: Relational Theory for Practitioners**

  Book by Christopher J. Date

- **Database Design and Relational Theory: Normal Forms and All That Jazz**

Book by Christopher J. Date

- **Database Systems: A Practical Approach to Design, Implementation, and Management**

  Book by Carolyn Begg and Thomas M. Connolly

- **Introductory Relational Database Design for Business, with Microsoft Access**

  Book by Bonnie R. Schultz and Jonathan Eckstein

- **Head First SQL: Your Brain on SQL − A Learner's Guide**

  Book by Lynn Beighley

- **Readings in Database Systems**

  Book by Joseph M Hellerstein

- **The art of SQL**

  Book by Stéphane Faroult

- **The theory of relational databases**

  Book by David Maier

- **The Data Warehouse Toolkit**

  Book by Ralph Kimball

- **MongoDB: The Definitive Guide**

  Book by Kristina Chodorow and Michael Dirolf

- **Learning SQL**

  Book by Alan Beaulieu

- **Joe Celko's SQL puzzles and answers**

  Book by Joe Celko

- **Foundations of databases**

  Book by Serge Abiteboul

- **Pro SQL Server Relational Database Design and Implementation: Best Practices for Scalability and Performance**

Book by Louis Davidson

- **Six-Step Relational Database Design: A Step by Step Approach to Relational Database Design and Development**

  Book by Fidel A. Captain

- **Databases, Types and the Relational Model: The Third Manifesto**

  Book by Christopher J. Date

- **Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement**

  Book by Eric Redmond and Jim R. Wilson

- **Practical SQL: A Beginner's Guide to Storytelling with Data**

  Book by Anthony DeBarros

- **Refactoring databases**

  Book by Scott Ambler

- **SQL QuickStart Guide: The Simplified Beginner's Guide to Managing, Analyzing, and Manipulating Data With SQL**

  Book by Walter Shields

- **PHP and MySQL: Server-side Web Development**

  Book by Jon Duckett

- **SQL Cookbook**

  Book by Anthony Molinaro

- **Learn MongoDB 4.x: A Guide to Understanding MongoDB Development and Administration for NoSQL Developers**

  Book by Doug Bierer

- **Professional Azure SQL Managed Database Administration: Efficiently Manage and Modernize Data in the Cloud Using Azure SQL**

  Book by Ahmad Osama and Shashikant Shakya

- **PostgreSQL: Up and Running**

  Book by Leo S. Hsu and Regina O. Obe

- **SQL All-in-One for Dummies**

  Book by Allen G. Taylor

- **An Introduction to Relational Database Theory**

  Book by Hugh Darwen

- **Transaction Processing: Concepts and Techniques**

  Book by Jim Gray

- **Making Sense of NoSQL: A Guide for Managers and the Rest of Us**

  Book by Ann Kelly, Ann Marie Kelly and Dan McCreary

- **A Practical Guide to Database Design**

  Book by Rex Hogan

- **Database Development for Dummies**

  Book by Allen G. Taylor

- **SQL Queries for Mere Mortals: A Hands-on Guide to Data Manipulation in SQL**

  Book by John Viescas and Michael J. Hernandez

- **Pro SQL Server 2012 Relational Database Design and Implementation**

  Book by Jessica Moss and Louis Davidson

- **SQL Performance Explained: Everything Developers Need to Know about SQL Performance**

  Book by Markus Winand

- **Refactoring Databases: Evolutionary Database Design**

  Book by Scott W Ambler and Pramod J Sadalage

- **Designing Data-Intensive Applications**

  Book by Martin Kleppmann

Programming today is a race between software engineers striving to build bigger and better idiot-proof programs, and the Universe trying to produce bigger and better idiots. So far, the Universe is winning.

— Rick Cook, The Wizardry Compiled

"In some ways, programming is like painting. You start with a blank canvas and certain basic raw materials. You use a combination of science, art, and craft to determine what to do with them."

– Andrew Hunt

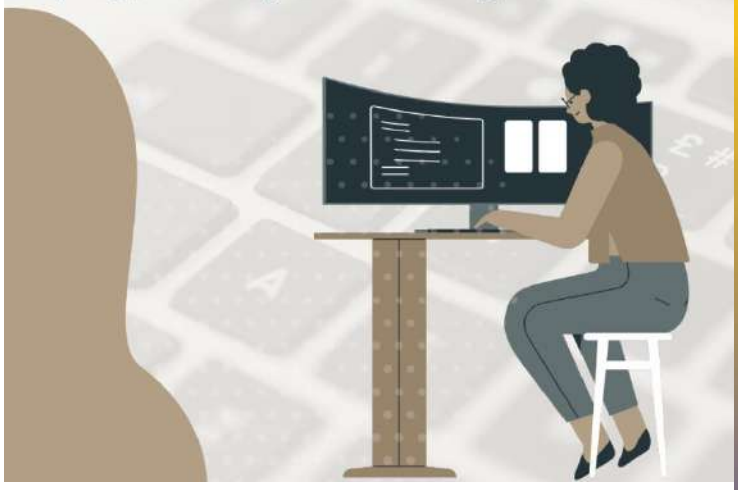
One final thought:

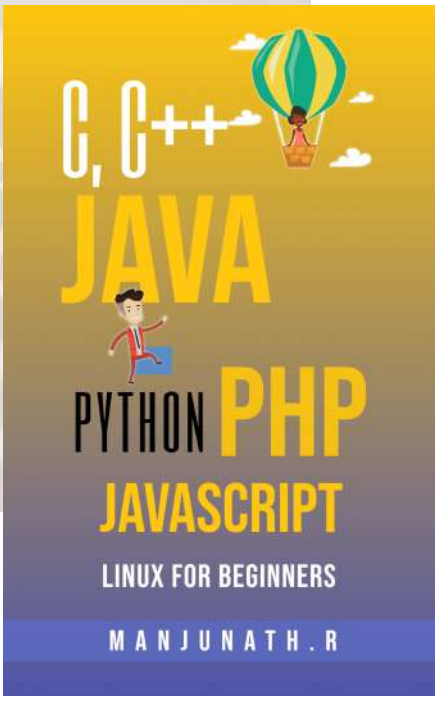
If you feel that this information has been useful to you, please take a moment to share it with your friends on LinkedIn, Facebook and Twitter. Consider writing a brief review on **Google Play Books** if you feel that this book has helped you in your programming career and you have learned something worthwhile.

Coding is both a science and creative art to me. It is both incredibly fun and fascinating. I want to spread my passion to as many individuals as I can. I also hope that this is not the end of your learning.

Thank you!

An ideal addition to your personal elibrary. With the aid of this indispensable reference book, you may quickly gain a grasp of Python, Java, JavaScript, C, C++, CSS, Data Science, HTML, LINUX and PHP. It can be challenging to understand the programming language's distinctive advantages and charms. Many programmers who are familiar with a variety of languages frequently approach them from a constrained perspective rather than enjoying their full expressivity. Some programmers incorrectly use Programmatic features, which can later result in serious issues. The programmatic method of writing programs—the ideal approach to use programming languages—is explained in this book. This book is for all programmers, whether you are a novice or an experienced pro. Its numerous examples and well paced discussions will be especially beneficial for beginners. Those who are already familiar with programming will probably gain more from this book, of course. I want you to be prepared to use programming to make a big difference.





Measuring programming progress by lines of code is like measuring aircraft building progress by weight.

– Bill Gates