Programming Fundamentals

Section 1

Types of Programming Languages

1. **Low-Level Languages**: These include **Machine Language** and **Assembly Language**. They are close to the computer's hardware but difficult for humans to understand and write.

- **Machine Language:**

The lowest-level programming language, consisting only of binary code (0s and 1s) that the computer's processor can execute directly. It is fast but very difficult for humans to read and write.

- **Assembly Language:**

A low-level programming language that uses symbolic codes (mnemonics) instead of binary, making it slightly easier for humans to understand. It requires an **assembler** to convert it into machine language.

2. **High-Level Languages**: Examples include **C, Java, and Python**. These languages are easier to read and write because they use commands that resemble natural language.

C++: A Middle-Level Programming Language

- **C++ is considered a High-Level Programming Language, but it also includes some Low-Level features, making it a Hybrid (Middle-Level) Language.**

- **Why?**
- ✅ **High-Level**: It supports **Object-Oriented Programming (OOP)**, provides built-in libraries, and is closer to natural language.
  ✅ **Low-Level**: It allows **direct memory manipulation**, such as using **pointers** and manual memory management.

cin

- **cin** is defined in <iostream> header file.
- **cin** is used to accept the input from the user
- General form to accept single input :

```
cin >> varName;
```

- General form to accept multiple inputs:

```
cin >> var1 >> var2 >> … >> varN;
```

cout

- **cout** is defined in <iostream> header file.
- **cout** is used to output something to the user.
- General form:

```
cout << varName;
```

or

```
cout << "Some String";
```

- General form to display multiple outputs :

```
cout << var1 << "Some String" << var2 << endl;
```

Comments

- Comments are for the reader, not the compiler
- Two types:
  - Single line

```
// This is a C++ program. It prints the sentence:
// Welcome to C++ Programming.
```

  - Multiple line

```
/*
You can include comments that can
occupy several lines.
*/
```

Read two integer numbers from the user then print their sum and average.

```
#include<iostream>
using namespace std; \\ This lets us use cout and cin without writing std::cout or std::cin
void main()
{
int num1, num2;
cout << "Enter two numbers: ";
cin >> num1 >> num2;
cout << "sum of 2 numbers = " << num1 + num2<<endl;
cout << "average of 2 numbers = " << (num1 + num2)/2 << endl;
system("Pause");
}
```

Read a positive integer from the user and print its square

```
#include<iostream>
#include<math.h>
using namespace std;
void main()
{
int num;
cout << "Enter number : ";
cin >> num;
cout << "Power = " << num *num <<endl;
cout << "Power using Pow function " << pow(num, 2)<<endl;
system("Pause");
}
```

Backslash codes

Code	Meaning
\b	Backspace
\r	Carriage return
\n	New line
\t	Horizontal tab
\a	Alert
\"	Double quotes
\'	Single quote
\\	Backslash
\x	Hexadecimal
\	Octal

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

Function (Structure)

```
#include<iostream>
using namespace std;
int sum(int a, int b)
{
int z = a + b;
cout << "sum of "<<a<<" and "<<b<<"=" << z<<endl;
return 0;
}
int main()
{
sum(3, 4);
sum(5, 6);
int x, y;
cout << "enter 2 numbers: "<<endl;
cin >> x >> y;
sum(x, y);
system("Pause"); // This is a Windows-only command that makes the program
return 0;        wait for the user to press a key before closing.
}
```

END