Programming Basics :- I

Lesson :- 2

⇒ Writing the first program :-

(ide :- Integrated development environment)

⇒ Before going to write our first code, we need the code editor, where we can write the things, in a very well mannered. Such example of the code editors are Vs code, Codeblocks. (or) you can simply write over internet with the help of repl.it. No need of installing any ide.

⇒ Install on the computer And start writting the code.

⇒ .cpp is the extension of c++ programs. So, whenever you need to create a file in Vs code, you have to use (.cpp) Extension.

⇒ As like in Flowchart, we have the start button, same we have in the programming, we have `int main() { }` This is the structure of any code, (or) you can say the starting code (or) main function. In bracket belongs, some code for execution. whatever code written in the Braces belongs to the main function.

⇒ The cout function is used to print any statement in C++ code. (cout <<)

⇒ And for the cout function, we need a header file, that may be standard (or) user-defined.

(directive)

`#include <iostream>`

so, to welcome in c++ program,

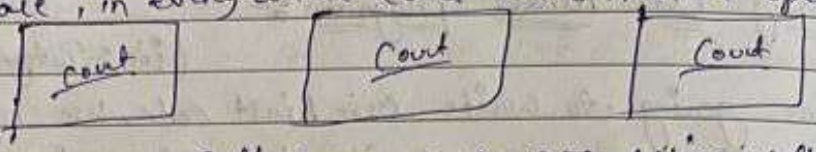
```
using namespace std.
int main()
{
cout << placement
}
```

(endl → for new line)

<< → used with cout for printing

cout << "Welcome to lagegi" << endl;

⇒ What does namespace std mean?
⇒ Let's think, we have 3 type of namespace. And in the namespace, in every where cout has different operation. like



using namespace std mean, we are using the cout, from the std namespace (I mean a namespace whose name is std, through which we can print something

Homework :- ① can i create custom library and own namespace? A:- Yes, using std.

⇒ #include <iostream> → It may ① standard file. (os)
② userdefined file.

⇒ Everything you need to print should be in quotes, for printing.

Question :- Can we create your own namespace?
A:- Yes, by userdefined namespace.
Q:- What all other namespaces? (Such as global namespace) (Contradata)

⇒ We can use for any printing statement in void function because, in void only execution statement happend, nothing Return function.

Data types and variables :-

⇒ For storing any type of data (like decimal, True/false, number) for that storing, we have data types.

```
int a = 5;
```

int → size of data, Type of data } we get two information.

1 byte = 8 bits

bits means 0/1

classmate Date Page 11

⇒ Generally, integer has 4 bytes (or) 8 bits. That Mean we got to know about 2 things. ① integer has 4 bytes (size) ② Here we are storing integer (Data type)

⇒ Somewhere, integer has 2 bytes (Based on compilers). Int can store numbers (Bothe +ve & -ve).

int a = 5 ; (Variable name). when we are saying that int a = 5 ; Means a block created and 5 is get stored. How? int has 4 bytes 00000000 00000000 00000000 00000101 (5 = binary no.)

⇔ bool a = true ; bool b = False ; (Bool has 1 byte). 1 byte has the smallest attribute/entity in cpu.

⇔ float f = 1.2 (float has 4 byte) (Decimal store)

⇔ Double d = 1.31 (Double has 8 byte). when we are saying that Double has 8 byte, int has 4 byte. What does that Mean? That Mean it will take 8 byte in Memory, for storing any data.

⇔ The main difference between double and float is one is size and the next one is precise. You can store More precisely in double rather than float.

⇒ There were some conditions Regarding the creation of variable name, i.e Case sensitive is not matter, you can give underscore but not space and you can not create number in start for variable name. You can also start with underscore (as variable name)

Remember ⇒ In bool Data type "1" is true and "0" Means false. (Int)

Now, we can print all, the data type & variable in our editor.

```
int a = 5;
Cout << " Value of a is " << a << endl;   ⇒ Print 5
float b = 1.25 ;
Cout << " Value of b is " << b << endl;   ⇒ 1.25
```

in bool, bool b = true;
cout << " value of b is " << b << endl; It will print 1.
Because, as we know 1 Mean true and 0 Mean false.
So, Here it will print 1 as true.

⇒ For getting size of any type of data :-
cout << " the size of int is " << sizeof(a) << endl; → (4 bytes)
⇒ Like, this you can find other data type.
Char ch = 'a'; (Character should always in single quote)
And the size of char is 1 (one).

⇒ In Character, only single Letter can store.

How the number stored in Memory ?

int a = 5; → variable = "a".
type → int, size :- 4 byte / 32 bit.

00000000 00000000 00000000 00000101

Here, 101 is the binary representation of 5. This is how (+ve) number get stored.

How -ve number get stored ?

There are Basically 3 steps. ① ignore -ve sign.
② Convert into binary code
③ Take 2's complement.

⇒ What does 2's Complement Mean.
① Take 1's Complement. ② Then add 1.
⇒ What is 1's Complement. Flip → all the bit.
Mean 0 to 1 & 1 to Zero (0).

classmate
Date
Page 13

For example :- 1010 (Find 2's complement).
① Flip 1 to 0, 0 to 1. (0101).
② Add 1. + 1
0110 (Done)

Note :- In addition of 1, Alway Remben if you have to plus 1 & 1, then shift 1 to next number, and put Zero in ans.

⇒ That Mean when you are adding 1, (10 is binary no. of 2)

```
  1
  0101
+    1
------
  0110
```

1+1=2 (10)

Now, let's take example int x = -5. (And we need to store this number)
step :- 1 Convert it into +ve number.
step :- 2 Make it's binary number into 1's complement.

```
  00000000  00000000  00000000  00000101
1's complement
  11111111  11111111  11111111  11111010
                                      +1
  ----------------------------------------
  11111111  11111111  11111111  11111011
```

step :- 3 Adding 1

This is how -ve no. get stored in Memory.

⇒ That Mean Whenever a number is stored, in such way in Test we are checking the 1st bit is 0 (or) 1. If 1 (that) Mean bit is negative. and if "0" bit is positive.
→ for ex: 11111111 11111011 (That Mean negative) 1st bit show the sign of the number. And from the 2nd bit to the last bit shows the numbers binary Representation.

⇒ For integer range, Any number binary Representation
Ex: 0000000 . . . . 101 → Rest int Range
→ 1st bit we will get +ve (or) -ve.
2 power because either 0 (or) 1.

-2^{31} to $2^{31}-1$

=) Char has 1 byte. = 8 bits. [| | | | | | | |]
=) Char value is stored in ASCII value.
So, the Range is [2^8] That Mean 8 block and 2 types you can store 0/1.
=) How are we going to differentiate between int (or) Char in Memory?
=) From the size of Data type we can get Char has 1 byte and int has 4 bytes.

Operator :- Arithmetic operator :- (+, -, %, /, *)

① int a = 5; ① int sum = a+b; ③ int Diff = a-b;
int b = 6; ② int Multiply = a*b;

④ int division = a/b; = 0.8 but, here as it's data type is int. So, it will store 1.
So, int / int ans is int.

=) So, if float / int = 5.0/3 = 1.6 (Ans in float).

Q:- int ans = 5.0/2 = {ans 1}
cout << 5.0/2 ⟶ {1.4}
That Means, whatever the denominator not, so powerful.
=) There are two types of type casting. ① Implicit. ② Explicit. =) Implicit Means by own. =) Explicit Means by ourselves.

=) Char ch = 'a';
int num = (int) ch;
Here as per ascII value of 'a' is 97, it will print 97.
⑤ Modulo operator Means gives Reminder. (5 % 2 = 1)
(Reminder)

Relational Operator :-

⇒ Relational operators are like =, >, <, >=, <=, != (not equal to)
⇒ (==) operator Means we are doing comparison.
⇒ if(a != b) a value is not equal to b it will give True.

⇒ int x = 5 , y = 3 ;
bool a = (x == y) ⟶ false.
int x = 5 , y = 4 ; bool b = (x != y) ⟶ True.

Logical Operator :- (&&, ||, !)

andand operator, OR operator, Not equal to operator.

For example :- ① Anshuman is a student.
② Anshuman is a coder.

⇒ if the both the statement is True in (and and) operator, then statement is True, if one false Means, then condition will false.

⇒ In (or) operator, if any one statement is True, then whole statement will True. (At least one True)

⇒ Not operator Means "0" to "1" to "0" and True to false
False to True

Bitwise operator :- Now, we are working bit level.

I Means Binary Representation.

int a=5 ; ⟶ 101
int b=6 ; ⟶ 110

int ans = a & b

(and operator)

```
  1 0 1
  1 1 0
  -----
  1 0 0   → 4
```

1 (or) a = 5, b = 6 → 1 0 1
(1 is True, 0 is false) (Remember) | 1 1 0
1 1 1 — (7) (Binary Representation)

And

x	y	output
0	0	0
0	1	0
1	0	0
1	1	1

or

x	y	output
0	0	0
0	1	1
1	0	1
1	1	1

NOT operator (!) :- Convert 0 to 1 and 1 to 0.

XOR (Exclusive operator) (symbol) (Most dangerous operator) (a ⊕ b)

x	y	operator
0	0	0
0	1	1
1	0	1
1	1	0

=> if both x and y is same, then ans is (0), And if different ans is (1). (Remember Strogies)

Left shift operator :- (padded 0 in the end)

[5 << 1], shif 5 by 1. , left shift operator Means pad with one zero. 00000101 ←
(5 X 2) 000000101(0) (0 added) → (ans is 10)
[5 << 2] 00000101 ←
(5 X 2 X 2) 00001010[00] (2 0 added) → (ans is 20)
(in the end 0 added)

Left shift operator, Basically Multiplying by 2. This is generally applicable, for small number, but not for any big number.
(#Not alway left shift Means multiply by 2)

Right shift operation :- (padded with 0 is start)

[15 >> 1] (5/2)

000101

0 0 0 0 1 0 → 2 (Binary Represent)

[5 >> 2] $\left(\frac{5}{2\times2}\right)$

000101

0001 → 1 (Binary Represent)

→ Whenever we Right shift, the number is divided by 2 to get the binary Represent.

Note:- ① (Just Read Article, how Right shift & Left shift works for negative)

② In the -ve number padded is based on compiler, but for posetive it is padded in last (Left), padded in 1st (Right).

H.W (DRY Run in Compiler about Right & Left shift operation).