Conditionals & Loops :- Lecture - 3

Conditionals - (if-else) statements at what condition particular code will run.

① If condition :-

→ Syntax -

```
if (condition)
{

}
```

These brackets are the scope for the if condition.

→ The code inside the brackets will only execute when the given condition will be true.

eg:- ①

```
if (age >= 18)
{
  cout << "Eligible" ;
}
  cout << "Not eligible" ;
```

* In the given eg. it will print eligible when age >= 18 otherwise it will print not eligible.

②

```
if (balance >= 10)
{
  cout << "Maggi" ;
}
```

(2) if-else condition :-

Syntax :

```
if (condition)
{
}
```

These brackets are the scope for if condition.

```
else
{
}
```

These brackets are the scope for else condition.

* Code inside the else will only execute when the condition for if become false.

Eg :- (1)

```
if (side == 3)
{
  cout << "Triangle" ;
}
else
{
  cout << "Not a triangle" ;
}
```

* If the side will be other than 3 it will go to the else and print the scope of code inside the else.

(3) If - Else if condition :-

Syntax -

```
if (condition)
{
}
```

These brackets are the scope for if condition.

```
else if (condition)
{
}
```

These brackets are the scope for else - if condition.

```
:
:
```

(optional) ← `else`

```
{
}
```

These brackets are the scope for else condition.

* We can use this condition when we have multiple condition to check. The else condition is the optional. It is not necessary to end with else condition.

eg:-

```
if(num > 0)
{
cout << "Positive number";
}
else if(num < 0)
{
cout << "Negative number";
}
else {
cout << "Zero";
}
```

Loops :-

Sequence of instruction that repeats until a condition is met.

- for
- while
- do-while
- for-each

① for- loop :-

Syntax – `for( int i=0 ; i<10 ; i=i+1 )`

- `int i=0` → initialization
- `i<10` → condition
- `i=i+1` → updation

Scope of for-loop:

```
{
// logic
}
```

Eg :-

```
for(int i=1 ; i<=5 ; i=i+1)
{
cout << "Shashank";
}
```

Dry Run

i=1
i<=5 – True
"Shashank"
i=i+1 = 1+1 = 2

2<=5 – True
"Shashank"
i= i+1 = 2+1 = 3

3<=5 – True
"Shashank"
i=i+1 = 3+1 = 4

4<=5 – True
"Shashank"
i=i+1 = 4+1 = 5

5<=5 – True
"Shashank"
i=i+1 = 5+1 = 6

6<=5 – False

Eg:-

```
for(int i = 10; i<=12; i=i+1)
{
  cout << "Mahalakshmi" :
}
```

Dry Run

i = 10

10 <= 12 — True "Mahalakshmi"

i = i+1 = 10+1 = 11

11 <= 12 — True "Mahalakshmi"

i = i+1 = 11+1 = 12

12 <= 12 — True

"Mahalakshmi"

i = i+1 = 12+1 = 13

13 <= 12 — False

Q. Print table for 2:

Ans:

```
for(int i=2; i<=10; i=i*2)
{
  cout << i << endl;
}
```

Output

2
4
8

```
for(int i=2; i<=10; i++)
{
  cout << 2*i << endl;
}
```

4	12	18
6	14	20
8	16	
10		

```
for(int i=100; i>=0; i=i/2)
{
  cout << i << endl;
}
```

100	6
50	3
25	1
12	

Combination of multiple loop :

Eg:-

```
for(int i=0; i<3; i=i+1)
{
  cout << "Outer loop";
  for(int j=0; j<3; j=j+1)
  {
    cout << "Inner loop";
  }
}
```

(i) → i=i+1
=1+1=2
2<3 — True
"Outer loop 2"
j=0
0<3 — True
"Inner loop 0"
j=j+1
=0+1=1
1<3 — True
"Inner loop 1"
j=j+1
=1+1=2
2<3 — True
"Inner loop 2"
j=j+1
=2+1=3
2<3 — False

(ii) → i=i+1
=0+1=1
1<3 — True
"Outer loop 1"
j=0
0<3 — True
"Inner loop 0"
j=j+1
=0+1=1
1<3 — True
"Inner loop 1"
j=j+1
=1+1=2
2<3 — True
"Inner loop 2"
j=j+1
=2+1=3
3<3 — False

(iii) → i=i+1
=2+1=3
3<3 — False

Dry Run

(i) → i=0
0<3 — True
"Outer loop 0"
j=0
0<3 — True
"Inner loop 0"
j=j+1=0+1=1
1<3 — True
"Inner loop 1"
j=j+1=1+1=2
2<3 — True
"Inner loop 2"
j=j+1=2+1=3
3<3 — False