<u>The FOR Loop</u> :-

Syntax :

```
for (initialization-expression; condition-expression; update-expression)
{
        body of the loop;
}
```

<u>Example :</u>

```
for (int i=1; i<=10; i++)
        cout<<"\t"<<i;
```

This loop will print the natural numbers from 1 to 10.

The three elements : *initialization-expression*, *condition-expression*, and *update-expression* govern the working of the loop and are hence called the loop control statements.

In a for loop, there can be multiple initialization and update expressions. These multiple expressions should be separated by commas. But, there can be only one condition-expression. Also, these three loop control elements are optional in a for loop but the two semicolons must appear even if any one (or all) of these expressions are missing.

<u>Example :</u>

```
for ( ; ; )
{

}
```

This is a perfectly valid code in C++. This will start an infinite loop.

The WHILE Loop :-

Syntax :

```
initialization-expression;
while(condition-expression)
{
    body of the loop;
    update-expression;
}
```

In a while loop, a loop control variable should be initialized before the loop begins. Also, the loop variable should be updated inside the body of the loop.

Example :

```
int i, num=5, fact;
while(num>0)
{
     fact = fact*i;
     i--;
}
```

This code snippet calculates the factorial of the number 5.

Note :

The For and While are *entry-controlled* Loops, i.e., the condition-expression is checked before entering the loop, if it evaluates to true, only then the loop body is executed.

Practice Problems :

1. Write a code to print the following shapes :-

(i)

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

Hint : Use two nested for loops

(ii)

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

Hint : Use two nested for loops, but the initialization-expressions will be the reverse of the first part.

(iii)

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

Hint : Try to combine both the above parts to get this output

Now, try out the same shapes using a while loop.