★ ITERATION STATEMENTS

* Also called as looping statements.
* Looping statements checks for a condition using relational operators when the condition returns true initially the same statement will be executed multiple number of times untill the condition returns false.
* When the condition returns false initially the block of statements called loop body will not be executed.
* three looping statements are.

1. while loop (Entry controlled loop)
2. for loop (Entry controlled loop)
3. do while loop (Exit controlled loop)

* There are 4 elements of a loop.

1. initialisation statements - before entering the loop variable must be initialised to first value.

   Eg :- i = 1, i = 2, j = 1, j = 2

2. test expression :- it is conditions using relational operators that decides whether a block of statements to be executed or not. If the condition returns true the block of statements will be executed or else not.

   Eg :- i < 10, i < n, i <= 10, i <= n, i > 0. etc.

3. update statement :- update statement modifies the loop by changing the value of a variable.

   Eg :- i++, i--, --j, j--, i = i/2 .... etc.

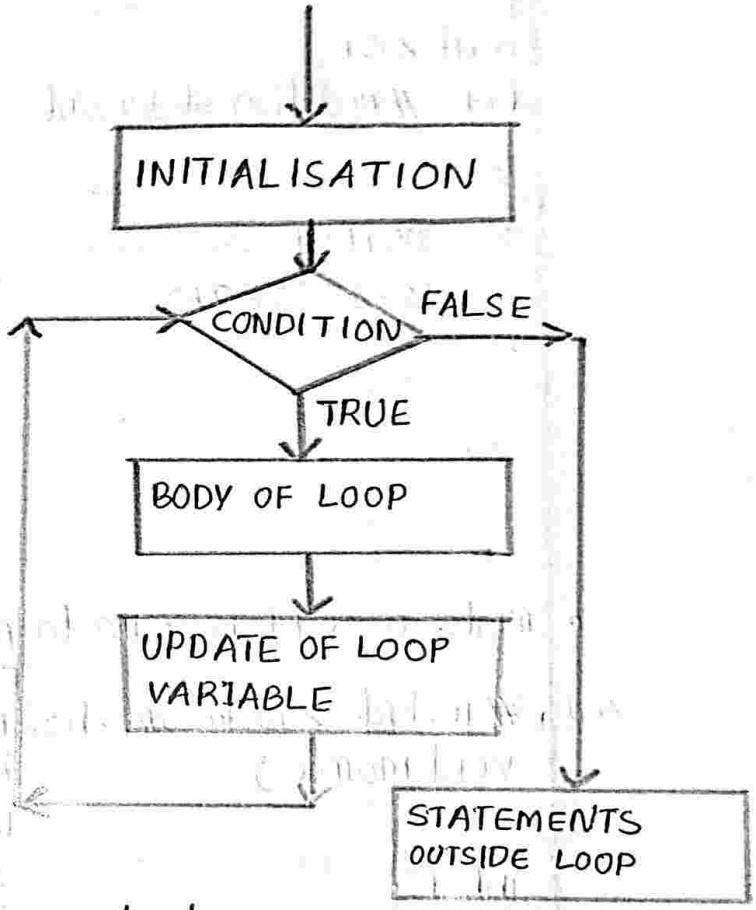
4. body of the loop :- block of statements that is to be executed repeatedly when the test expression returns true untill the condition becomes false is called body of loop. Body of loop can have any type of statements.

★ while statement

* It is an entry controlled loop.
* The condition is checked before entering into the loop body.

SYNTAX

```
initialisation;
while (test expression)
{
body of loop;
updation statement;
}
```



* At first initialisation will be executed, then the test expression will be executed if the condition returns true the body of loop will be executed and updation statement will be executed. Initialisation will be executed only for once.
* Then the test expression is checked when the condition returns true the loop and body and the updation statement will be executed, then the test expression is checked. The process continues until the test expression returns false.

- write a c++ program to print values from 1 to 10.

Ans)

```
#include<iostream.h>
void main( )
{
int i;
i=1; // initialisation statement
while(i<=10) // test expression
{
cout<<i;
i++; // updation statement
}
}
```

OUTPUT

12345678910

EXECUTION

1.
i=1
1<=10 → t
2.
i=2
2<=10 → t
3.
i=3
3<=10 → t
4.
i=4
4<=10 → 10
5.
i=5
5<=10 → t
6.
i=6
6<=10 → t
7.
i=7
7<=10 → t
8.
i=8
8<=10 → t
9.
i=9
9<=10 → t
10.
i=10
10<=10 → t
11.
i=11
11<=10 → f

- write a c++ program to print values from 10 to 1.

Ans)

```
#include<iostream.h>
void main( )
{
int i;
i=10;
while(i>=1)
{
cout<<i;
i--;
}
}
```

Execution

1.
i=10
10>=1 → t
2.
i=9
9>=1 → t
3.
i=8
8>=1 → t
4.
i=7
7>=1 → t
5.
i=6
6>=1 → t
6.
i=5
5>=1 → t
7.
i=4
4>=1 → t
8.
i=3
3>=1 → t
9.
i=2
2>=1 → t
10.
i=1
1>=1 → t
11.
i=0
0>=1 → f

- write a c++ program to print n number of values.

Ans)

```
#include<iostream.h>
void main()
{
int i, n;
i =1;
cout<<"enter the value for n";
cin>>n;
while(i<=n)
{
cout<<i;
i++;
} }
```

Execution.

```
1.
i =1
n=5
1<=5 →t
f=2
n=5
2<=5 →t
i=3
n=5
3<=5 →t
i=4
n=5
4<=5 →t
i=5
n=5
5<=5 →t
i=6
n=5
6<=5 →f
```

OUTPUT

12345

- write a c++ program to print sum of first 10 natural numbers.

Ans)

```
#include<iostream.h>
void main()
{
int i, sum = 0;
i =1;
while (i<=10)
{
sum = sum + i;
i++;
}
cout<< sum;
}
```

- write a c++ program to print the sum of n natural numbers.

Ans)

```
#include<iostream.h>
void main()
{
int i,n,sum;
i=1;
```

```
cout<<"enter the value for n";
cin>>n;
while (i<=n)
{
sum = sum+i;
}
cout<<sum;
}
```

- Write a C++ program to print the even numbers up to 10 -

Ans)

```
#include<iostream.h>
void main()
{
int i;
i=2;
while(i<=10)
{
cout<<i;
i+=2;
}
}
```

- Write a C++ program to find the sum of first 10 even numbers.

Ans)

```
#include<iostream.h>
void main()
{
int i, sum;
i = 2;
while(i<=10)
{
sum = sum+i;
i+=2;
}
}
```

• Write a c++ program to find factorial of a number.

Ans)
```
#include<iostream.h>
void main()
{
int n, i, f=1;
i=1;
cout<<" enter the value for n";
cin>>n;
while(i<=n)
{
f=f*i;
}
cout<<f;
}
```

2. for LOOP

* Entry controlled loop.
* contains initialisation statement, test expression and updation statement

Initialisation statement :- a variable to control the loop is assigned to some value.

Eg :- i++, ++i, j++, ++j, j+=2

Test expression :- condition that will checked before each execution of loop body.

Eg :- i<10, i<=10, i>10

Updation :- variable that controls the loop execution will be done by updation statements.

Eg :- i++, ++i, j++, ++j

SYNTAX

```
for (inisialisation ; test_expression ; updation)
{
loop body;
}
```

* Initialisation statement will be executed only once, updation and test expression will be executed multiple number of times.
* Looping execution starts with initialisation statement, then the test expression will be executed when the condition returns true the control of the program enters the loop body executing block of statements.
* When the execution of loop body completes, the program control reaches the updation statement changing the value of the variable that controls the loop. Then the test expression is again executed when the condition returns true again the loop body will get executed. The same process continues until the condition returns false.
* When the condition or test_expression returns false value the loop will be terminated.