Unit II

Loops

for, while, do-while, Nesting of loops

Dr. Md Tarique Jamal Ansari
Assistant Professor
Department of Computer Application
Integral University, Lucknow
Email: tjansari@iul.ac.in

Loops in C programming language

A **Loop** executes the sequence of statements many times until the stated condition becomes false. A loop consists of two parts, a body of a loop and a control statement. The control statement is a combination of some conditions that direct the body of the loop to execute until the specified condition becomes false. The purpose of the loop is to repeat the same code a number of times.

'C' programming language provides us with three types of loop constructs:

1. The for loop
2. The while loop
3. The do-while loop

1. The *For* Loop

Syntax:

for (*initialization*; *test condition*; *increment or decrement*)
{
Statement(s);
}

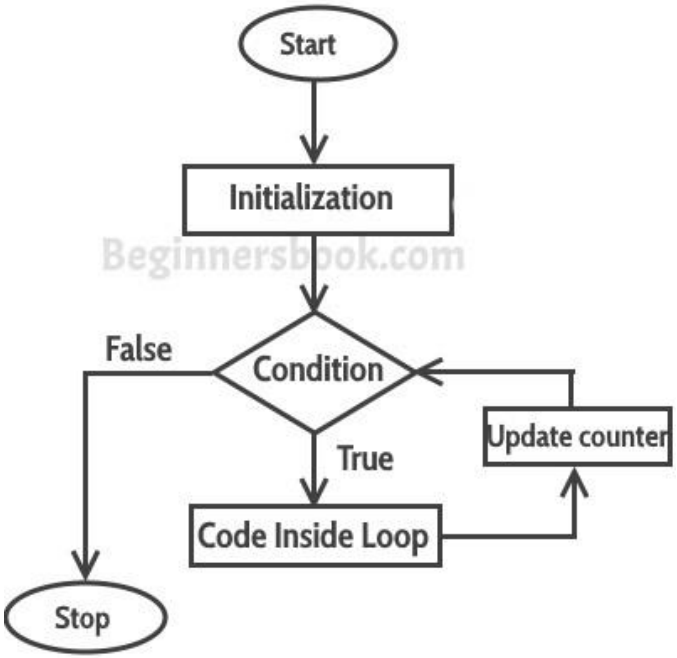

Figure: **The *for* loop statement**

Example: The *For* Loop

```
/* Program to print first n natural numbers */
#include <stdio.h>
int main()
{
int i,n;
printf ("Enter value of n \n");
scanf ("%d",&n);
printf("\nThe first %d natural numbers are :\n", n);
for (i=1;i<=n;++i)
{
printf ("%d",i);
}
return 0;
}
```

OUTPUT

```
Enter value of n
6
The first 6 natural numbers are:
1 2 3 4 5 6
```

2. The *While* Loop

Syntax:

while (test condition*)*
{
body_of_the_loop;
}

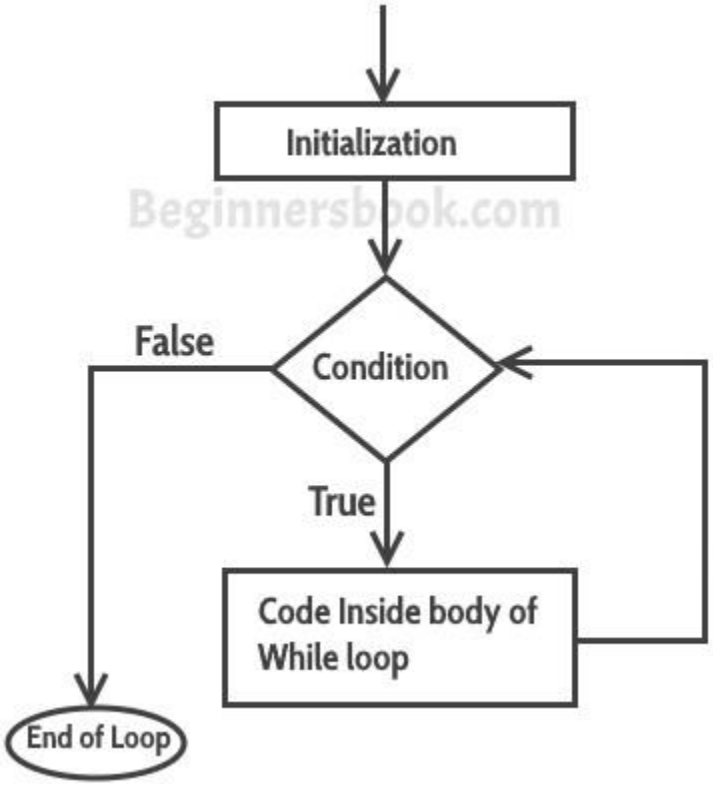

Figure: **The *while* loop statement**

Example: The *While* Loop

```
#include <stdio.h>
int main()
{
int count=1;
while (count <= 4)
        {
        printf("%d ", count);
        count++;
        }
return 0;
}
```

Output:

```
1 2 3 4
```

3. The do...while Loop

Syntax:

do

{

statement(s)*;*

} ***while****(test condition);*

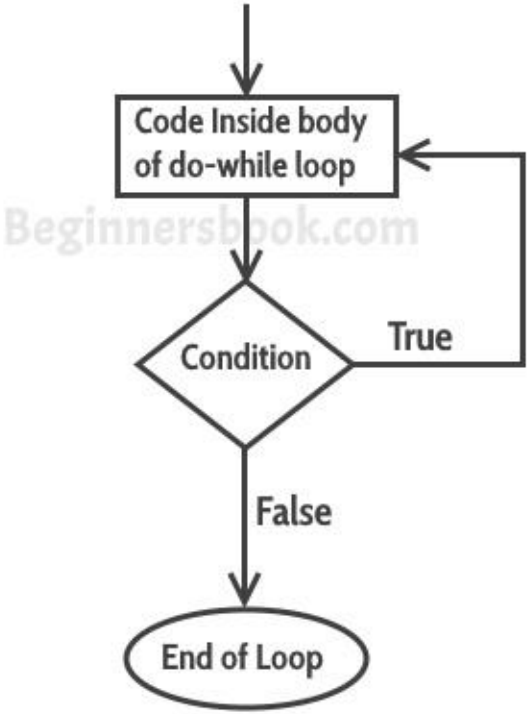

Figure: **The *do…while* statement**

Example: do...while Loop

```
#include <stdio.h>
int main()
{
int j=0;
        do
        {
        printf("Value of variable j is: %d\n", j);
        j++;
        }
        while (j<=3);
        return 0;
}
```

Output:

```
Value of variable j is: 0
Value of variable j is: 1
Value of variable j is: 2
Value of variable j is: 3
```

Nesting of loops

Syntax:

```
Outer_loop
{
        Inner_loop
        {
        // inner loop statements.
        }
        // outer loop statements.
}
```

Here **Outer_loop** and **Inner_loop** are the valid loops that can be a 'for' loop, 'while' loop or 'do-while' loop.

1. Nested for loop

Syntax:

```
for (initialization; condition; update)
{
   for(initialization; condition; update)
   {
         // inner loop statements.
   }
   // outer loop statements.
}
```

Example: Nested for loop

```
#include <stdio.h>
int main()
{
int n;   //variable declaration
printf("Enter the value of n :");
scanf("%d", &n);
// Displaying the n tables.
for(int i=1;i<=n;i++)//outer loop
{
        for(int j=1;j<=10;j++)//inner loop
        {
        printf("%d\t",(i*j));//printing the value
        }
printf("\n");
}
return 0;
}
```

Output

input

```
Enter the value of n : 3
1       2       3       4       5       6       7       8       9       10
2       4       6       8       10      12      14      16      18      20
3       6       9       12      15      18      21      24      27      30


...Program finished with exit code 0
Press ENTER to exit console.
```

2. Nested while loop

Syntax:

```
while(condition)
{
    while(condition)
    {
        // inner loop statements.
    }
// outer loop statements.
}
```

Example: Nested while loop

```
#include <stdio.h>
int main()
{
int a = 1,
while(a <= 5)
{
Int b = 1;
while(b <= 5)
{
printf("%d ", b);
b++;
}
printf("\n");
a++;
}
return 0; }
```

```
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
```

3. Nested do..while loop

Syntax:

```
do
{
  do
 {
     // inner loop statements.
  }while(condition);
// outer loop statements.
}while(condition);
```

Example

```
#include <stdio.h>
int main()
{
 int i=1;
do          // outer loop
{
   int j=1;
   do      // inner loop
  {
    printf("*");
    j++;
  }while(j<=8);
   printf("\n");
   i++;
    }while(i<=4);
}
```

Problem Statement

```
/*printing the pattern
    ********
    ********
    ********
    ******** */
```