Different Types of loop in C

Rupa Nayak
Maste of Computer Application
Kalyani Government Eng. College
Kalyani,Nadia,India
rupa.nayak98@gmail.com

***Abstract*—Besides all other programming languages C also have the special feature to execute block of statements more than one time, which is called 'Loop'. This loops helps C programs more structured and more effective. There exist different types of loops in C. Let's learn about then for writing better C programs.**

I. INTRODUCTION

C is a middle level structured programming language because it have a very special feature such as different types of Loops. Loop controls structures are used when we want to execute a block of statements several times. Loop statement always comes with a conditional statement. Until the condition become 'false' the loop will execute continuously. There are three types of loop control statements in C programming language

- **for loop**
- **while loop**
- **do-while loop**

II. DETAIL EXPLANATIONS ARE GIVEN BELOW-

A. for loop

The 'for' loop statement is very useful to write programs in C language. It has three expressions in 'for' loop separated by semicolons (;). The 'for' loop statements are written as-

```
for(expression1;expression2;expression3)
        statement1;
        statement2;
.....................
}
```

The body of the loop can be single statement or a block of statement. If the body contains only one statement the it can be written as-

```
for(expression1;expression2;expression3)
        statement;
```

Here expression1 is the initialisation expression, expression2 is conditional expression and expression3 is an update expression. Expression1 is first executed when the loop is started and this expression initialised the loop variable. This expression executes only ones. Then expression2 is condition and is tested before each iteration of the loop. This expression generally relational and logical operator. When this expression becomes false the loop terminated. Expression3 is used to update the loop variable after each iteration. This expression generally increment or decrement operator.

First expression1 is executed and initialised the loop variable. Then the compiler check the expression2, either the given condition become false or not. If the condition become false the loop terminated and executing following statements after the 'for' loop closing brush. If the condition become true then body of the loop is executed and the control is shifted to the expression3 statement of the 'for' loop. The expression3 update loop variable and again control transfer to expression2 for checking the condition to be true or false .In this repeated way the loop is executed. Flow chart is given below for better understanding-

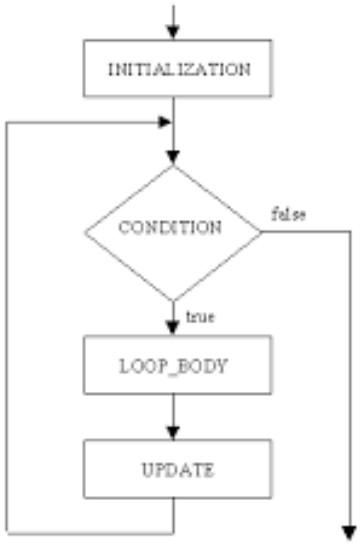

Fig. 1. Flow chart of "for" loop

This loop is generally used when the number of iterations are known in advance. Let's see an example using 'for' loop. Suppose we want to print the sum of first 10 numbers.

```
for(i=1;i¡=10;i++){
        s=s+i;
              }
```

Here in the starting of the loop the loop variable ‘i’ is initialised by 1 and the check whether ‘i’ is equal to 10 or not. ‘i’ is not 0, so the value of ‘i’ is added with and then the value of ‘i’ is incremented by 1. Again the control checks the condition and so on. When the value of ‘i’ becomes 10 then loop will terminated and started execute the rest of the program.

B. while loop

While loop is also a very popular loop. This loop is generally used when the no of iterations do not know in advance. ‘while’ loop is represented as-

```
While(expression)
        Statement1;
        Statement2;
        }
```

If there exist only one statement in the body of the loop then the loop can be written as

```
While(expression)
        Statement;
```

This loop first check whether the loop variable satisfy the condition or not. If loop variable satisfy the condition the body of the loop will be executed and go to next iteration and check the condition again. If the condition becomes false the loop will be terminated. For this featured of ‘while’ loop it is called as ‘entry control’ loop. The flow chart of ‘while’ loop is given below-

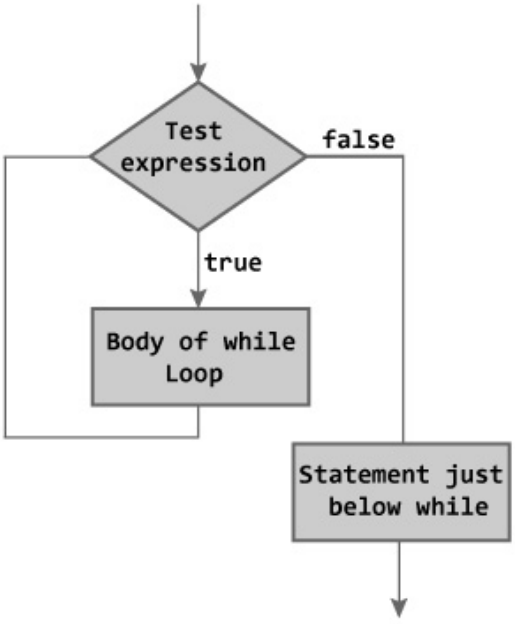

Figure: Flowchart of while Loop

Fig. 2. Flow Chart of while loop

Suppose we want to print 1 to 10 numbers using ‘while’ loop-

```
a                       =                       1;
while                                           (a¡=10)
        printf                                  (“
        a++;
}
```

Here loop variable a is initialised before entering the loop and check for condition and the condition is true so the value will be printed and increment the value of a by a. After the body of the loop is finished as well as next iteration will be started and again check the condition and so on.

C. do-while loop

It is also a loop but do not used frequently. This loop can be represented as –

```
do
        Statemebt1;
        Statement2;
        ..............
}while(expression);
```

The body of the loop can contain one or more sentences. In this loop first body of the loop is executed and at the end of each iteration condition has been checked. For this reason it is called ‘exit control’ loop. It is basically used in menu driven programs. Flow chart of ‘do-while’ loop is as follow- Here firstly

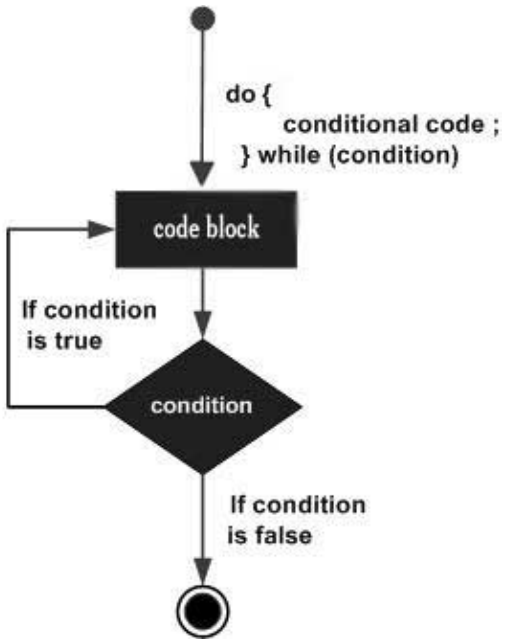

Fig. 3. Flow Chart of do while loop

body of the loop is executed then condition is evaluated. If the condition is true the body of the loop is executed repeatedly until the condition become false. In this

loop the semicolon (;) is placed after the condition. Suppose we want inputs from user until it is 0 and the print it.

```
do{
        printf("Enter a number (0 to stop): ");
                scanf("%d",&n);
        printf("%d",n);
}while(n);
```

First the program asked for a number. Then number has been printed and then check if the number is 0 or not. If it is 0 then the loop terminated otherwise again asked for another number and so on.