C Loops

The looping can be defined as repeating the same process multiple times until a specific condition satisfies. There are three types of loops used in the C language. In this part of the tutorial, we are going to learn all the aspects of C loops.

Why use loops in C language?

The looping simplifies the complex problems into the easy ones. It enables us to alter the flow of the program so that instead of writing the same code again and again, we can repeat the same code for a finite number of times. For example, if we need to print the first 10 natural numbers then, instead of using the printf statement 10 times, we can print inside a loop which runs up to 10 iterations.

Advantage of loops in C

1) It provides code reusability.

2) Using loops, we do not need to write the same code again and again.

3) Using loops, we can traverse over the elements of data structures (array or linked lists).

Types of C Loops

There are three types of loops in C language that is given below:

1. do while
2. while
3. for

do-while loop in C

The do-while loop continues until a given condition satisfies. It is also called post tested loop. It is used when it is necessary to execute the loop at least once (mostly menu driven programs).

The syntax of do-while loop in c language is given below:

```
1. do{
2. //code to be executed
3. }while(condition);
```

Flowchart and Example of do-while loop

while loop in C

The while loop in c is to be used in the scenario where we don't know the number of iterations in advance. The block of statements is executed in the while loop until the condition specified in the while loop is satisfied. It is also called a pre-tested loop.

The syntax of while loop in c language is given below:

```
1. while(condition){
2. //code to be executed
3. }
```

Flowchart and Example of while loop

for loop in C

The for loop is used in the case where we need to execute some part of the code until the given condition is satisfied. The for loop is also called as a per-tested loop. It is better to use for loop if the number of iteration is known in advance.

The syntax of for loop in c language is given below:

```
1. for(initialization;condition;incr/decr){
2. //code to be executed
3. }
```

do while loop in C

The do while loop is a post tested loop. Using the do-while loop, we can repeat the execution of several parts of the statements. The do-while loop is mainly used in the case where we need to execute the loop at least once. The do-while loop is mostly used in menu-driven programs where the termination condition depends upon the end user.

do while loop syntax

The syntax of the C language do-while loop is given below:

```
do{
//code to be executed
}while(condition);
```

Example 1

```
#include<stdio.h>
#include<stdlib.h>
void main ()
{
    char c;
    int choice,dummy;
    do{
    printf("\n1. Print Hello\n2. Print Javatpoint\n3. Exit\n");
    scanf("%d",&choice);
    switch(choice)
    {
        case 1 :
        printf("Hello");
        break;
        case 2:
        printf("Javatpoint");
        break;
        case 3:
        exit(0);
        break;
```

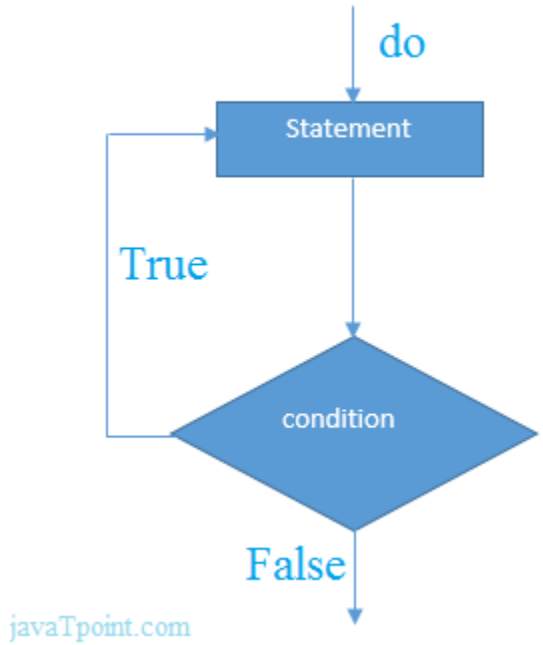
```
21.         default:
22.         printf("please enter valid choice");
23.     }
24.     printf("do you want to enter more?");
25.     scanf("%d",&dummy);
26.     scanf("%c",&c);
27.     }while(c=='y');
28. }
```

Output

```
1. Print Hello
2. Print Javatpoint
3. Exit
1
Hello
do you want to enter more?
y

1. Print Hello
2. Print Javatpoint
3. Exit
2
Javatpoint
do you want to enter more?
n
```

Flowchart of do while loop



do while example

There is given the simple program of c language do while loop where we are printing the table of 1.

```
#include<stdio.h>
int main(){
int i=1;
do{
printf("%d \n",i);
i++;
}while(i<=10);
return 0;
}
```

Output

```
1
2
3
4
5
6
7
8
9
10
```

Program to print table for the given number using do while loop

```
#include<stdio.h>
int main(){
int i=1,number=0;
printf("Enter a number: ");
scanf("%d",&number);
do{
printf("%d \n",(number*i));
i++;
}while(i<=10);
return 0;
}
```

Output

```
Enter a number: 5
5
10
15
20
25
30
35
40
45
50
Enter a number: 10
10
20
30
40
50
60
70
80
90
100
```

Infinitive do while loop

The do-while loop will run infinite times if we pass any non-zero value as the conditional expression.

```
1. do{
2. //statement
3. }while(1);
```