C Loops – While Loop, Do-While Loop, For Loop

C Loops

C Loops : Looping statements are used to repeat the execution of a list of statements. There are three types of looping statements (iterative statements). They are :

- While loop
- Do-while loop
- For loop

C While Loop

The while loop provides a mechanism to repeat the execution of a list of statements while a particular condition is true.

The syntax of while loop is:

```
while (condition) {
    //while block statement(s)
}
```

Let us write a C program with while loop. In the following program, we print whole numbers from 0 to 5 using C While Loop.

C Program

```
#include<stdio.h>

int main() {
    int i=0;
    while(i<=5) {
        printf("%d",i);
        i++;
    }
    return 0;
}
```

Output

```
0
1
2
3
4
5
```

The while loop will execute as long as the given condition is true. If the given condition is not true, then the statements enclosed in the loop are never executed.

Following is another C program that uses while loop to find the sum of first 100 numbers.

C Program

```c
#include<stdio.h>

int main() {
    int i=0,sum=0;
    while(i<=100) {
        sum+=i;
        i++;
    }
    printf("Sum : %d",sum);
    return 0;
}
```

Output

```
Sum : 5050
```

While loop is very useful, when the number of times the statements in the loop has to be executed is not known in advance.

Read more about C While Loop.

C Do-While Loop

The do-while loop is similar to while loop. The only difference is that in do-while loop, the test condition is evaluated at the end of loop. In do-while loop, the test condition is evaluated at the end. So, the body of the loop gets executed atleast one time even if the condition is false.

The syntax of do-while loop is

```c
do {
    //while block statement(s)
} while (condition);
```

In the following C program, we printing numbers from 0 to 5 using do-while loop.

C Program

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

Output

```
0
1
2
3
4
5
```

In the following C program, we use do-while loop to print the number, its square and its cube.

C Program

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

Output

```
Enter number:3

|        1      |        1      |        1
|        2      |        4      |        8
|        3      |        9      |        27
```

Read more about C Do-While Loop.

C For Loop

C For Loop contains the initialization and update of loop control variables in the syntax itself.

Syntax

```
for(initialization; condition; update) {
    // for block statement(s)
}
```

where,

initialization of the loop allows to initialize the loop control variables.

condition is an expression that evaluates to a boolean value.

update may include incrementing, decrementing, or any other modification to the loop control variables or other variables.

Following is a C Program that uses For Loop to find factorial of a number.

C Program

```
#include<stdio.h>
int main() {
    int fact=1,num,i;
    printf("\n Enter number:");
    scanf("%d",&num);
    if(num==0) {
        fact=1;
    } else {
        for(i=1;i<=num;i++)
        fact=fact*i;
    }

    printf("Factorial of %d is %d",num,fact);
    return 0;
}
```

Output

```
Enter number:5
Factorial of 5 is 120
```

Read more about C For Loop.

Nested Loops

Nested loop means a loop inside another loop.It works with all three loops but mostly it can be used with for loop.Inner loop will execute some of the statements and the outer loop will execute the inner loop depends on given conditions.

Example

```c
#include<stdio.h>

int main() {
    int i,j;
    for(i=0;i<=4;i++) {
        for(j=0;j<=i;j++) {
            printf("* ");
        }
        printf("\n");
    }
    return 0;
}
```

Output

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

Interview Questions on C Loops

1. What will be output of following c code?

```c
#include<stdio.h>
int main() {
    int i,j;
    i=j=2,3;
    while(--i&&j++)
        printf("%d  %d",i,j);
    return 0;
}
```

Answer

```
1 3
```

Step by step explanation

- Initial value of variable
- i = 2
- j = 2
- Consider the while condition : –i && j++
- In first iteration:
- –i && j++
- = 1 && 2 //In c any non-zero number represents true.
- = 1 (True)
- So while loop condition is true. Hence printf function will print value of i = 1 and j = 3 (Due to post increment operator)
- In second iteration:
- –i && j++
- = 0 && 3 //In c zero represents false
- = 0 //False
- So while loop condition is false. Hence program control will come out of the for loop.

2. Find the output of following c program

```
#include<stdio.h>
int main(){
        for(;;) {
             printf("%d ",10);
        }
        return 0;
}
```

Answer

Infinite loop.

Explanation

Refer Note provided in the earlier part of this article.

3. What would be the output of the following c program

```
#include<stdio.h>
int i=40;
extern int i;
int main() {
    do {
        printf("%d",i++);
    }
    while(5,4,3.6.0,0);
    return 0;
}
```

Answer

Step by step explaination

Initial value of variable i is 40First iteration:printf function will print i++ i.e. 40do – while condition is : (5,4,3.6.0,0)Here comma is behaving as operator and it will return 0. So while condition is false hence program control will come out of the for loop.

4. How many times this loop will execute?

```c
#include<stdio.h>
int main(){
    char c=125;
    do
        printf("%d ",c);
    while(c++);
    return 0;
}
```

Answer

Finite number of times. Precisely 131 times.

Explanation

- If we will increment the char variable c it will increment as:
- 126,127,-128,-127,126 . . . . , 3, 2, 1, 0
- When variable c = 0 then loop will terminate.

Conclusion

In this C Tutorial, we have learnt about different looping statements in C programming language. Also, we have gone through different examples to understand the usage of C looping statements.

- C Constants
- C if
- C if else
- C Ternary Operator

C loops

- C While Loop
- C For Loop
- C Structures
- C Unions
- C typedef

C String Operations

- C Reverse String
- C String Length
- C Compare Strings

C FileOperations

- C Write to File
- C Delete File
- C Concatenate Files