

Programming Fundamental

Lab#06

Table of Contents

Control Structure/Loop:.....	2
For Loop:	2
While Loop:	5
Do-while Loop:	7
Nested Loop:	9

Control Structure/Loop:

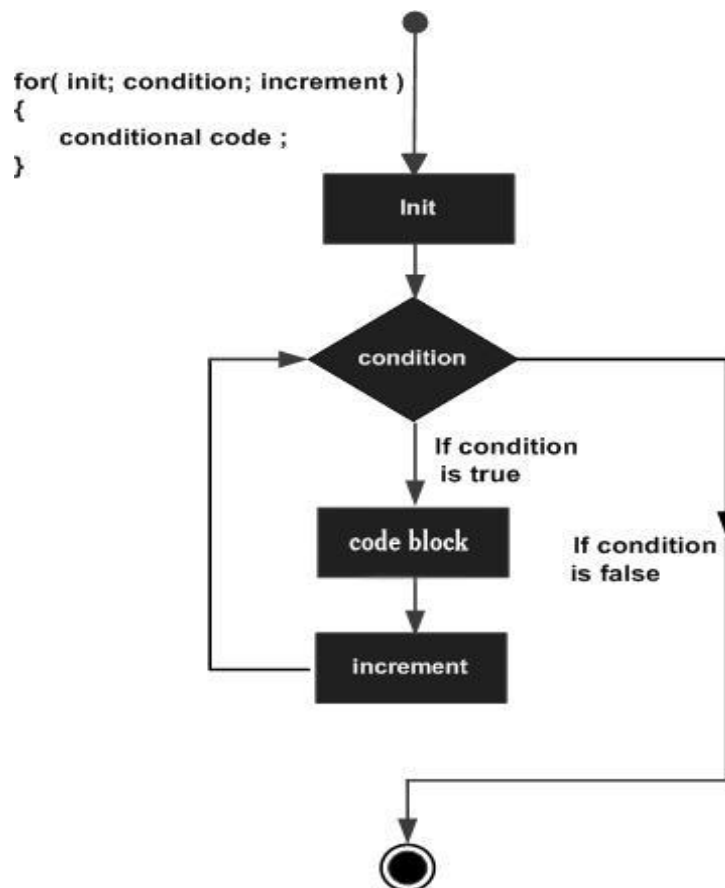
Loops are used in programming to repeat a specific block until some end condition is met.

There are four type of loops in C++ programming:

1. For loop
2. While loop
3. Do while loop
4. Nested loop

For Loop:

A **for** loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.



Syntax

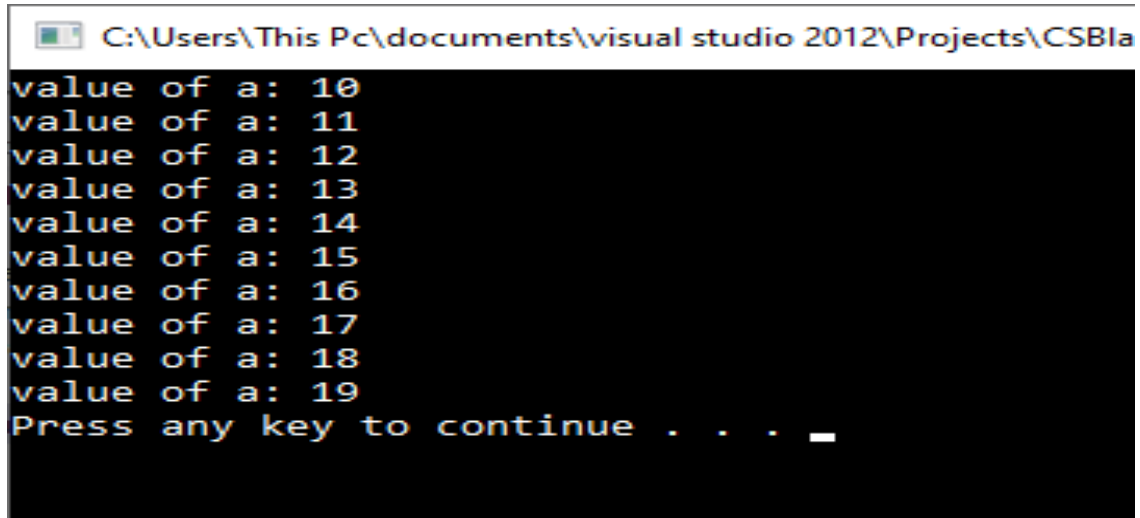
```
for ( init; condition; increment )
{
    statement(s);
}
```

Here is the flow of control in a for loop –

- The **init** step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You are not required to put a statement here, as long as a semicolon appears.
- Next, the **condition** is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and flow of control jumps to the next statement just after the for loop.
- After the body of the for loop executes, the flow of control jumps back up to the **increment** statement. This statement can be left blank, as long as a semicolon appears after the condition.
- The condition is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then increment step, and then again condition). After the condition becomes false, the for loop terminates.

```
#include <iostream>
using namespace std;

int main ()
{
    // for loop execution
    for( int a = 10; a < 20; a++ )
    {
        cout << "value of a: " << a << endl
            ;
    }
    system("pause");
    return 0;
}
```



```
C:\Users\This Pc\documents\visual studio 2012\Projects\CSBla
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15
value of a: 16
value of a: 17
value of a: 18
value of a: 19
Press any key to continue . . . _
```

```
#include <iostream>
using namespace std;

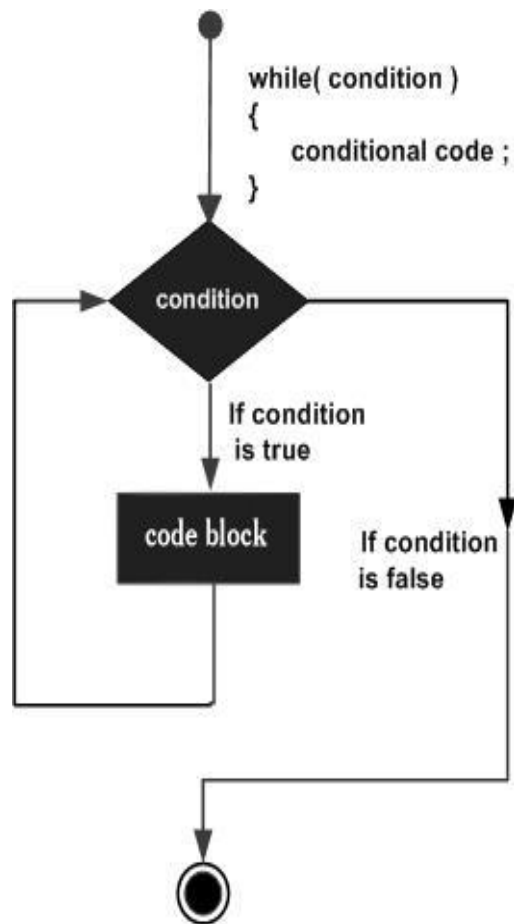
int main ()
{
    // for loop execution
    for( int a = 0; a <= 10; a++ )
    {
        cout << "value of a: " << a << endl
            ;
    }
    system("pause");
    return 0;
}
```

Output:

?????

While Loop:

A **while** loop statement repeatedly executes a target statement as long as a given condition is true.



Syntax

```
while(condition)
{
    statement(s);
}
```

Here, **statement(s)** may be a single statement or a block of statements. The **condition** may be any expression, and true is any non-zero value. The loop iterates while the condition is true.

When the condition becomes false, program control passes to the line immediately following the loop.

```

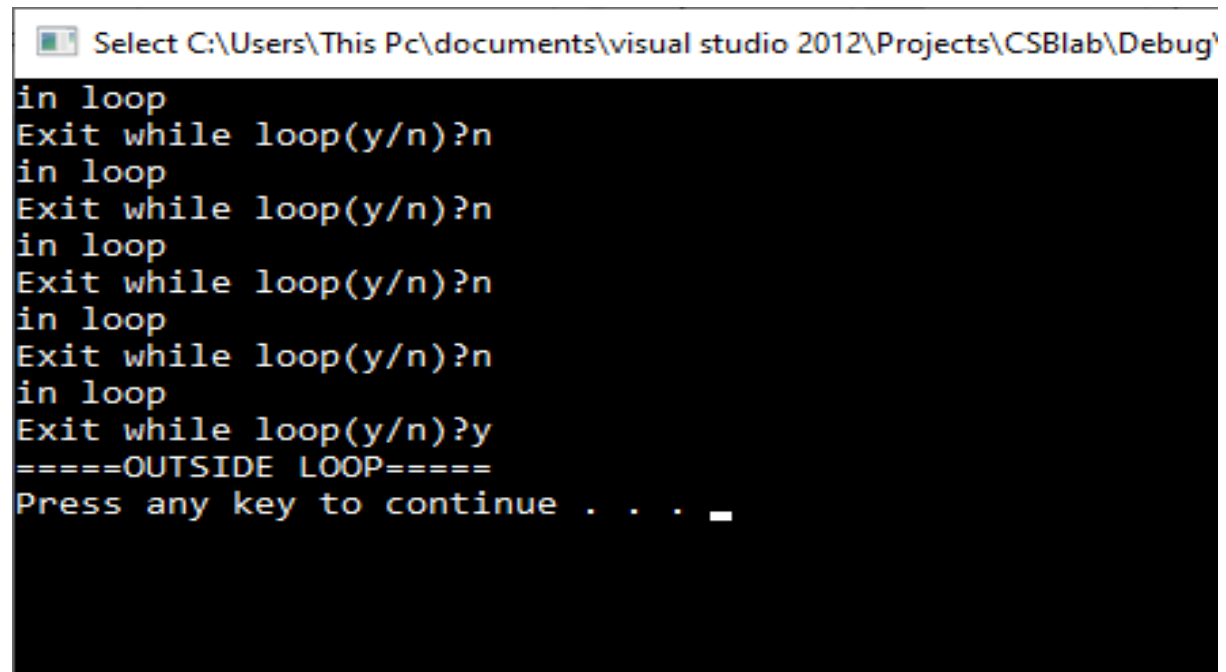
#include <iostream>
using namespace std;

int main () {
    // Local variable declaration:
    char c = 'n';

    // while loop execution
    while( c != 'y' )
    {
        cout << "in loop"<<endl;
        cout<<"Exit while loop(y/n)?";
        cin>>c;
    }
    cout<<"====OUTSIDE LOOP===="<<endl;
    system("pause");

    return 0;
}

```



```

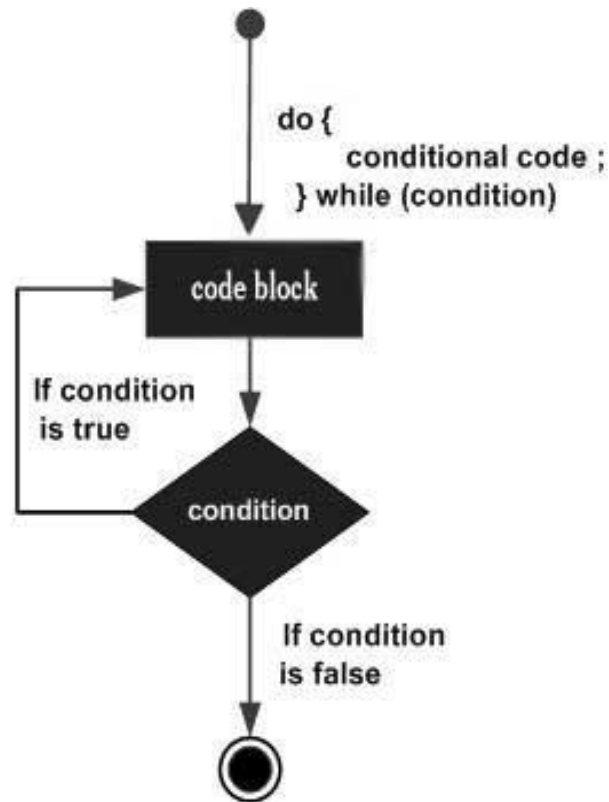
Select C:\Users\This Pc\documents\visual studio 2012\Projects\CSBlab\Debug'
in loop
Exit while loop(y/n)?n
in loop
Exit while loop(y/n)?n
in loop
Exit while loop(y/n)?n
in loop
Exit while loop(y/n)?n
in loop
Exit while loop(y/n)?y
====OUTSIDE LOOP====
Press any key to continue . . . _

```

Do-while Loop:

Unlike **for** and **while** loops, which test the loop condition at the top of the loop, the **do...while** loop checks its condition at the bottom of the loop.

A **do...while** loop is similar to a while loop, except that a do...while loop is guaranteed to execute at least one time.



Syntax

```
do
{
    statement(s);
}
while( condition );
```

Notice that the conditional expression appears at the end of the loop, so the statement(s) in the loop execute once before the condition is tested.

If the condition is true, the flow of control jumps back up to do, and the statement(s) in the loop execute again. This process repeats until the given condition becomes false.

```

#include <iostream>
using namespace std;

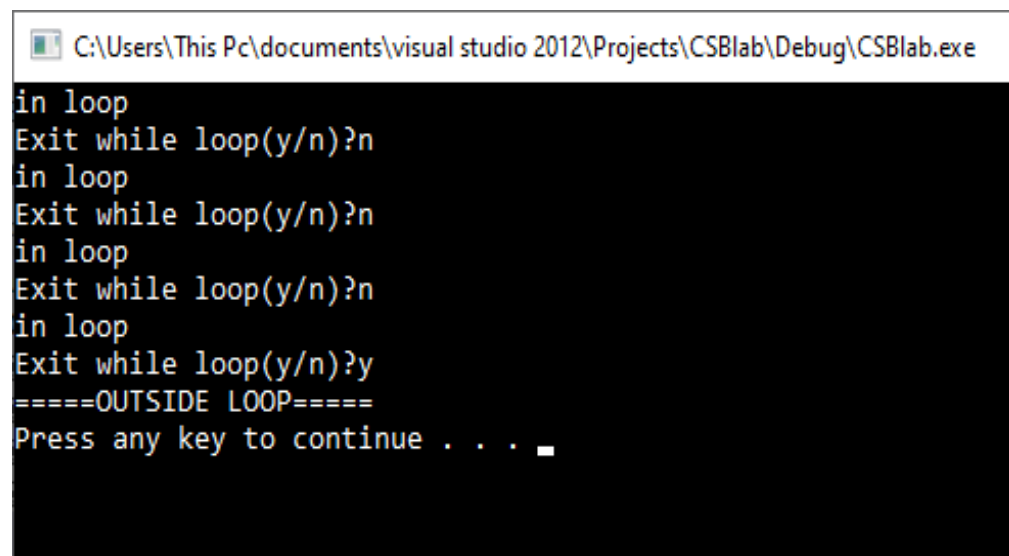
int main ()
{
    // Local variable declaration:
    char c ;

    // do-while loop execution
    do
    {
        cout << "in loop"<<endl;
        cout<<"Exit while loop(y/n)?";
        cin>>c;
    }while( c !='y' );

    cout<<"====OUTSIDE LOOP===="<<endl;
    system("pause");

    return 0;
}

```



```

C:\Users\This Pc\documents\visual studio 2012\Projects\CSBlab\Debug\CSBlab.exe
in loop
Exit while loop(y/n)?n
in loop
Exit while loop(y/n)?n
in loop
Exit while loop(y/n)?n
in loop
Exit while loop(y/n)?y
====OUTSIDE LOOP====
Press any key to continue . . . 

```


Nested Loop:

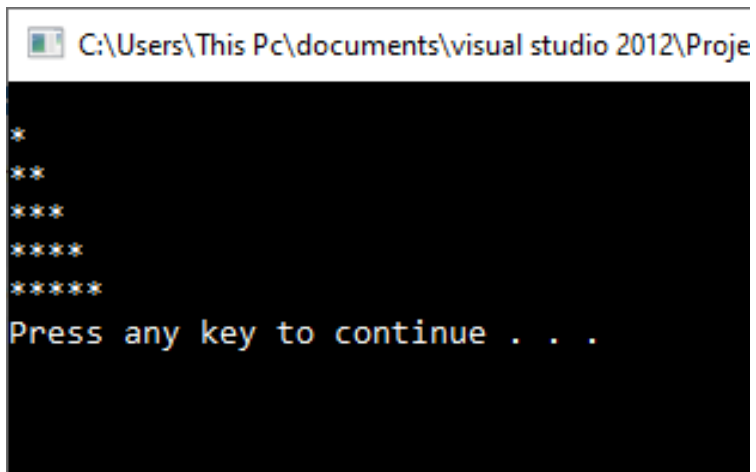
A loop inside another loop is called a nested loop.

```
#include <iostream>
using namespace std;

int main () {
    for (int i=0;i<=5;i++)
    {
        for (int j=0;j<i;j++)
        {
            cout<<"*";

        }
        cout<<endl;
    }
    system("pause");

    return 0;
}
```



Summary

1 for loop

Execute a sequence of statements multiple times and abbreviates the code that manages the loop variable.

2	<p>while loop</p> <p>Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body.</p>
3	<p>do...while loop</p> <p>Like a 'while' statement, except that it tests the condition at the end of the loop body.</p>
4	<p>nested loops</p> <p>You can use one or more loop inside any another 'while', 'for' or 'do..while' loop.</p>