Loops

Ahsan Ayub

Ph.D. Student, Department of Computer Science

Graduate Research Assistant, CEROC

CSC 1300: Introduction to Programming

Friday, September 24, 2021

Simple Problem-Solving Tasks

Write a C++ program that will display 1 to 5 on screen.

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

int main()
{
        cout << “1” << endl;
        cout << “2” << endl;
        cout << “3” << endl;
        cout << “4” << endl;
        cout << “5” << endl;
        return 0;
}
```

Simple Problem-Solving Tasks

Write a C++ program that will take a int (let's assume n) and display n to n+4 on screen.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        cout << n << endl;
        cout << n+1 << endl;
        cout << n+2 << endl;
        cout << n+3 << endl;
        cout << n+4 << endl;
        return 0;
}
```

Simple Problem-Solving Tasks

Write a C++ program that will take a int (let's assume n) and display `n to n+100` on screen.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        cout << n << endl;
        cout << n+1 << endl;
        cout << n+2 << endl;
        ...
        ...
        return 0;
}
```

Introducing Loops

- Execute certain statements <u>while</u> *certain conditions are met*
- C++ has three kinds of loops
  - `for`
  - `while`
  - `do-while`

`while` Loops

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

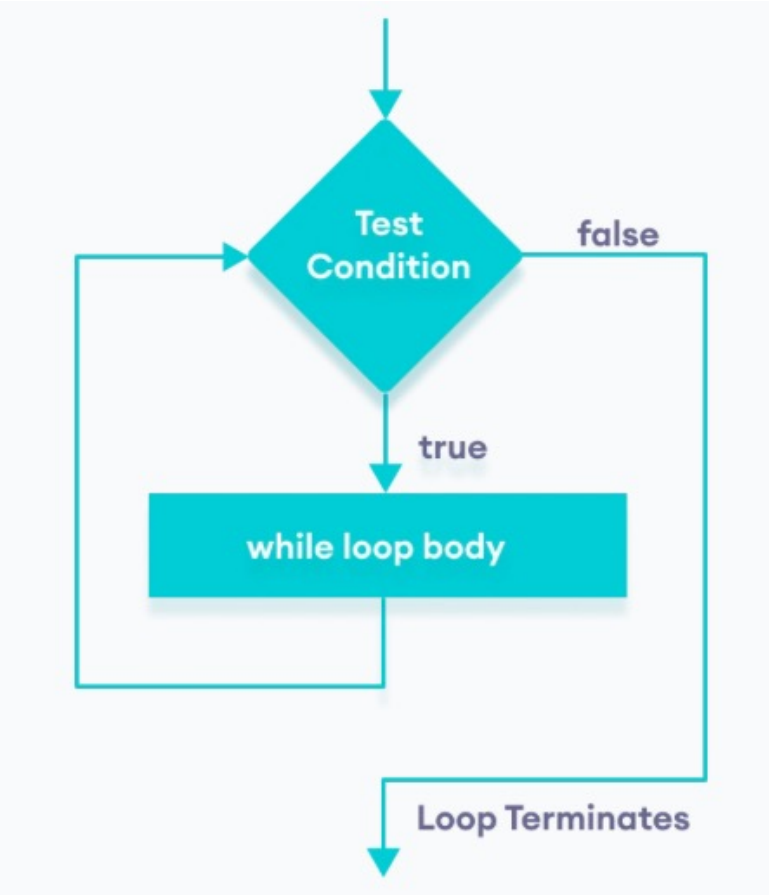

Image Source Programiz – https://www.programiz.com/cpp-programming/do-while-loop

while Loops

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

```
while (expression is true)
{
        action1;
}
```

```
while (expression is true) {
        action1;
}
```

```
while (expression is true)
        action1;
```

```
while (expression is true)
{
        action1;
        action2;
}
```

Must include the parathesis

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+4 on screen.

Tips: Try to find out where the steps are (somewhat) being repeated

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        cout << n << endl;
        cout << n+1 << endl;
        cout << n+2 << endl;
        cout << n+3 << endl;
        cout << n+4 << endl;
        return 0;
}
```

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+4 on screen.

Tips: Try to find out where the steps are (somewhat) being repeated

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        cout << n << endl;
        cout << n+1 << endl;
        cout << n+2 << endl;
        cout << n+3 << endl;
        cout << n+4 << endl;
        return 0;
}
```

n, n+1, n+2, n+3, n+4

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+4 on screen.

Tips: Try to find out where the steps are (somewhat) being repeated

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        cout << n << endl;
        cout << n+1 << endl;
        cout << n+2 << endl;
        cout << n+3 << endl;
        cout << n+4 << endl;
        return 0;
}
```

n+0
n+1
n+2
n+3
n+4

i=0 to 4

n, n+1, n+2, n+3, n+4

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+4 on screen.

Tips: Try to find out where the steps are (somewhat) being repeated

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
0

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
0

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
            cout << n + i << endl;
            i = i + 1;
        }
        return 0;
}
```

while Loops

i
~~0~~
1

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
1

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
1

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
1

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
            cout << n + i << endl;
            i = i + 1;
        }
        return 0;
}
```

while Loops

$\frac{i}{\cancel{x}\ 2}$

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

$\frac{i}{2}$

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

$\frac{i}{2}$

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

$\frac{i}{2}$

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
            cout << n + i << endl;
            i = i + 1;
        }
        return 0;
}
```

while Loops

i
2
3

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
3

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
3

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
3

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
            cout << n + i << endl;
            i = i + 1;
        }
        return 0;
}
```

while Loops

i
~~3~~
4

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
4

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
4

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
4

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
            cout << n + i << endl;
            i = i + 1;
        }
        return 0;
}
```

while Loops

i
4
5

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
5

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
5

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

while Loops

i
5

- A while loop evaluates the condition.
- If the condition (or the expression) evaluates to true, the code inside the while loop is executed.
- The condition is evaluated again.
- This process continues until the condition is false.
- When the condition evaluates to false, the loop terminates.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 4)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display `n to n+100` on screen.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 100)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+100 on screen.

What will happen if we simply change the condition? (Case 1)

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i != 100)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+100 on screen.

What will happen if we simply change the condition? (Case 2)

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i >= 100)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+100 on screen.

What will happen if we simply change the condition? (Case 3)

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i = 100)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

Revisit the Problem

Write a C++ program that will take a int (let's assume n) and display n to n+100 on screen.

What will happen if we forget to modify the loop control variable inside the loop body?

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0; // loop controlling variable
        while (i <= 100)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

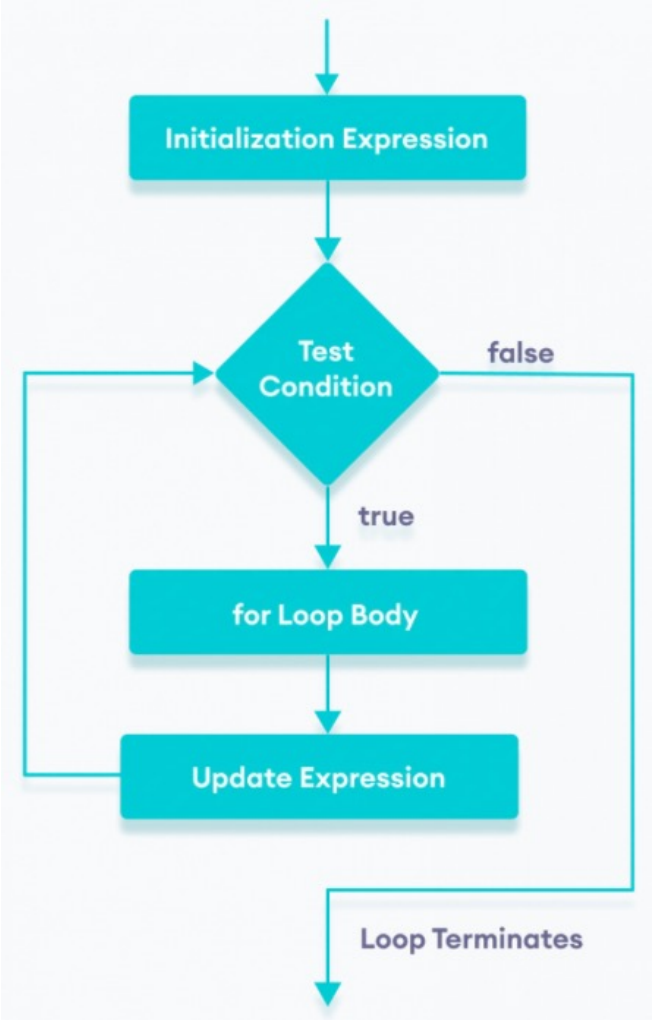

Image Source Programiz – https://www.programiz.com/cpp-programming/for-loop

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

```
for (initialization; expression; update)
{
        action1;
}
```

```
for (initialization; expression; update)
        action1;
}
```

```
for (initialization; expression; update)
        action1;
```

```
for (initialization; expression; update)
{
        action1;
        action2;
}
```

Must include the parathesis

for Loops

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0;
        while (i <= 5)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i = i + 1)
        {
                cout << n + i << endl;
        }
        return 0;
}
```

for Loops

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        int i = 0;
        while (i <= 5)
        {
                cout << n + i << endl;
                i = i + 1;
        }
        return 0;
}
```

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i = i + 1)
        {
                cout << n + i << endl;
        }
        return 0;
}
```

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i++)
        {
                cout << n + i << endl;
        }
        return 0;
}
```

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i++)
        {
                cout << n + i << endl;
        }
        return 0;
}
```

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i++)
        {
              ➡ cout << n + i << endl;
        }
        return 0;
}
```

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i++)
        {
                cout << n + i << endl;
        }
        return 0;
}
```

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i++)
        {
                cout << n + i << endl;
        }
        return 0;
}
```

for Loops

- A for loop first initializes the loop controlling variable(s) and *only executes this once*.
- If the condition (or the expression) evaluates to true, the code inside the for loop is executed.
- Then, it updates the loop control variable's value.
- The condition is checked again and repeat the process if it's true. It false, then it gets terminated.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input
        // Display the numbers
        for (int i = 0; i <= 5; i++)
        {
                cout << n + i << endl;
        }
        return 0;
}
```

Output Tracing

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

int main()
{
        for(int i = 0; i < 5; i++)
        {
                cout << “We know how loop works now!” << endl;
        }
        return 0;
}
```

```
??
```

Output Tracing

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

int main()
{
        for(int i = 0; i < 5; i++)
        {
                cout << “We know how loop works now!” << endl;
        }
        return 0;
}
```

```
We know how loop works now!
We know how loop works now!
We know how loop works now!
We know how loop works now!
We know how loop works now!
```

Output Tracing

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

int main()
{
        for(int i = 0; i < 5; i--)
        {
                cout << “We know how loop works now!” << endl;
        }
        return 0;
}
```

```
??
```

Output Tracing

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

int main()
{
        for(int i = 0; i < 5; i--)
        {
                cout << “We know how loop works now!” << endl;
        }
        return 0;
}
```

```
We know how loop works now!
We know how loop works now!
We know how loop works now!
We know how loop works now!
We know how loop works now!
…
```

Programming Challenge

Write a C++ program (on pen and paper) that will take a positive integer, `n` and display all the even numbers from `0 to n`

input

```
15
```

output

```
0
2
4
6
8
10
12
14
```

Output Tracing

```
#include <iostream>
using namespace std;
int main()
{
        // Declare the number for the user input
        int n;
        cin >> n;

        if(n <= 0)      // Terminate the program for not positive number
                return -1;

        // Performing the main computation
        for(int i = 0; i <= n; i++)
        {
                if(i % 2 == 0)
                        cout << i << endl;
        }
        return 0;
}
```

Output Tracing

```
#include <iostream>
using namespace std;
int main()
{
        // Declare the number for the user input
        int n;
        cin >> n;
        if(n <= 0)      // Terminate the program for not positive number
                return -1;
        // Performing the main computation
        int i = 0;
        while(i <= n)
        {
                if(i % 2 == 0)
                        cout << i << endl;
                i += 1;
        }
        return 0;
}
```

Output Tracing

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

int main()
{
        // Declare and initialize a variable named n
        int n = 15;
        while (n >= 0)
        {
                if(n % 2 != 0)
                        cout << n << “ ”;
                n -= 1; // Decrementing by 1
        }
        cout << endl;
        return 0;
}
```

```
??
```

Output Tracing

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

int main()
{
        // Declare and initialize a variable named n
        int n = 15;
        while (n >= 0)
        {
                if(n % 2 != 0)
                        cout << n << “ ”;
                n -= 1; // Decrementing by 1
        }
        cout << endl;
        return 0;
}
```

```
15 13 11 9 7 5 3 1
```

do…while Loops

- The body of the loop is executed at first. Then the condition is evaluated.
- If the condition (or the expression) evaluates to true, the body of the loop inside the do statement is executed again.
- The condition is evaluated once again.
- If the condition (or the expression) evaluates to true, the body of the loop inside the do statement is executed again.
- This process continues until the condition evaluates to false. Then the loop stops.

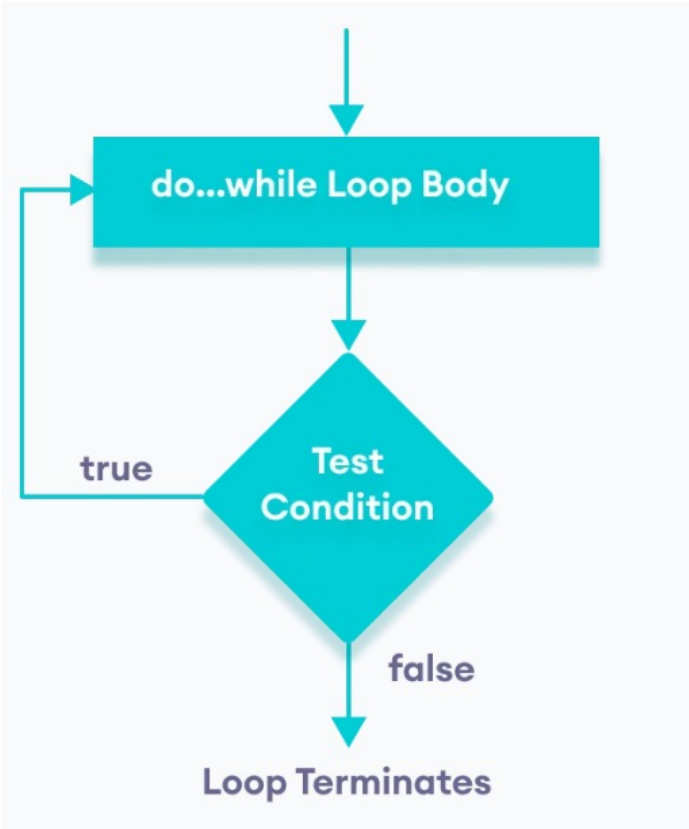

Image Source Programiz – https://www.programiz.com/cpp-programming/do-while-loop

do…while Loops

- The body of the loop is executed at first. Then the condition is evaluated.
- If the condition (or the expression) evaluates to true, the body of the loop inside the do statement is executed again.
- The condition is evaluated once again.
- If the condition (or the expression) evaluates to true, the body of the loop inside the do statement is executed again.
- This process continues until the condition evaluates to false. Then the loop stops.

```
do
{
        action 1;
        action 2;
} while (expression);
```

```
do {
        action 1;
        action 2;
} while (expression);
```

```
do {
        action 1;
        action 2;
}
while (expression);
```

Image Source Programiz – https://www.programiz.com/cpp-programming/do-while-loop

Output Tracing

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

int main()
{
        // Loop controlling variable
        int i = 0;
        do
        {
                cout << “We know how loop works now!” << endl;
                i += 1;
        } while (i < 5);
        return 0;
}
??
```

Output Tracing

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

int main()
{
        // Loop controlling variable
        int i = 0;
        do
        {
                cout << “We know how loop works now!” << endl;
                i += 1;
        } while (i < 5);
        return 0;
}
```

```
We know how loop works now!
We know how loop works now!
We know how loop works now!
We know how loop works now!
We know how loop works now!
```

Output Tracing

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

int main()
{
        // Loop controlling variable
        int i = 0;
        do
        {
                cout << “We know how loop works now!” << endl;
                i += 1;
        } while (i > 5);
        return 0;
}
```

```
??
```

Output Tracing

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

int main()
{
        // Loop controlling variable
        int i = 0;
        do
        {
                cout << “We know how loop works now!” << endl;
                i += 1;
        } while (i > 5);
        return 0;
}
```

```
We know how loop works now!
```

Note: The do..while loop will execute the loop body before even checking the condition / evaluating the expression.

Programming Challenge

Write a C++ program that will display the following using loop.

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

Introducing Nested Loops

- It is possible to create loops inside loops by simply placing these structures inside the statement blocks. This allows for more complicated program behavior.

```
for (initialization; expression; update)
{
        for(initialization; expression; update)
        {
                // body
        }
}
```

Programming Challenge – Solution

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

int main()
{
        // First loop to navigate the lines
        for(int i = 1; i <= 5; i++)
        {
                // Second loop to print the stars
                for(int j = 1; j <= i; j++)
                {
                        cout << “*”;
                }
                cout << endl;
        }
        return 0;
}
```

Programming Challenge

Write a C++ program that will display the following using loop.

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

Programming Challenge – Solution

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

int main()
{
        // First loop to navigate the lines
        for(int i = 5; i >= 1; i--)
        {
                // Second loop to print the stars
                for(int j = 1; j <= i; j++)
                {
                        cout << “*”;
                }
                cout << endl;
        }
        return 0;
}
```

`for` vs `while` Loop

- A `for` loop is usually used when the number of iterations is known.
- `while` and `do...while` loops are usually used when the number of iterations is unknown.

```
// This loop is iterated 5 times
for (int i = 0; i < 5; i++)
{
        //body of the loop
}
```

```
// This loop is iterated until
// the condition is met
while (condition)
{
        //body of the loop
}
```

Source Programiz – https://www.programiz.com/cpp-programming/do-while-loop

Infinite `while` Loop

- If the condition of a loop is always true, the loop runs for infinite times until the memory is full.
- To terminate (or exit out) from the loop, we use `break` keyword.

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

int main()
{
        // Declare a variable to take input
        int number;
        while(true) // Infinite loop
        {
                cin >> number;

                // Loop terminating condition
                // Check for negative number
                if(number < 0)
                        break;
        }
        return 0;
}
```

Source Programiz – https://www.programiz.com/cpp-programming/do-while-loop

C++ break Statement

```
for (init; condition; update) {
    // code
    if (condition to break) {
        break;
    }
    // code
}
```

```
while (condition) {
    // code
    if (condition to break) {
        break;
    }
    // code
}
```

Image Source Programiz – https://www.programiz.com/cpp-programming/break-statement

Programming Challenge

Write a C++ program that will display the following using loop.

```
*****

*****

*****
```

Programming Challenge – Solution

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

int main()
{
        for(int i = 1; i <= 5; i++) // First loop to navigate the lines
        {
                for(int j = 1; j <= 5; j++) // Second loop to print the stars
                {
                        if(i % 2 == 0) // Loop terminating condition
                        {
                                break;
                        }
                        cout << “*”;
                }
                cout << endl;
        }
        return 0;
}
```

C++ continue Statement

```
for (init; condition; update) {
    // code
    if (condition to break) {
        continue;
    }
    // code
}
```

```
while (condition) {
    // code
    if (condition to break) {
        continue;
    }
    // code
}
```

Image Source Programiz – https://www.programiz.com/cpp-programming/continue-statement

Revisit the Old Problem

Write a C++ program that will take a int (let's assume n) and display `n` to `n+10` on screen <u>where the value is odd</u>.

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

int main()
{
        //Declare the variable, n
        int n;
        cin >> n; // Take the input

        // Display the numbers
        for(int i = 0; i < 10; i++)
        {
                if((n + i) % 0 == 0) // is even?
                        continue;
                cout << n + i << endl;
        }
        return 0;
}
```

Programming Challenge

Write a C++ program that will display the following using loop.

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

Programming Challenge – Solution

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

int main()
{
        for(int i = 1; i <= 5; i++) // First loop to navigate the lines
        {
                for(int j = 1; j <= 5; j++) // Second loop to print the stars
                {
                        if(j % 2 == 0) // Put space and don’t show stars on even positions
                        {
                                cout << “ ”;
                                continue;
                        }
                        cout << “*”;
                }
                cout << endl;
        }
        return 0;
}
```

Remarks

- Reference
  - ZyBooks, TNTech CSC 1300: Introduction to Problem Solving and Computer Programming
  - MIT 6.096 Introduction to C++
  - Kanetkar, Yashavant P. "Let Us C.”