Programming Fundamentals with C++

Lecture 11 – Loop Statements

Dr. Muhammad Sajjad

RA: Asad Ullah

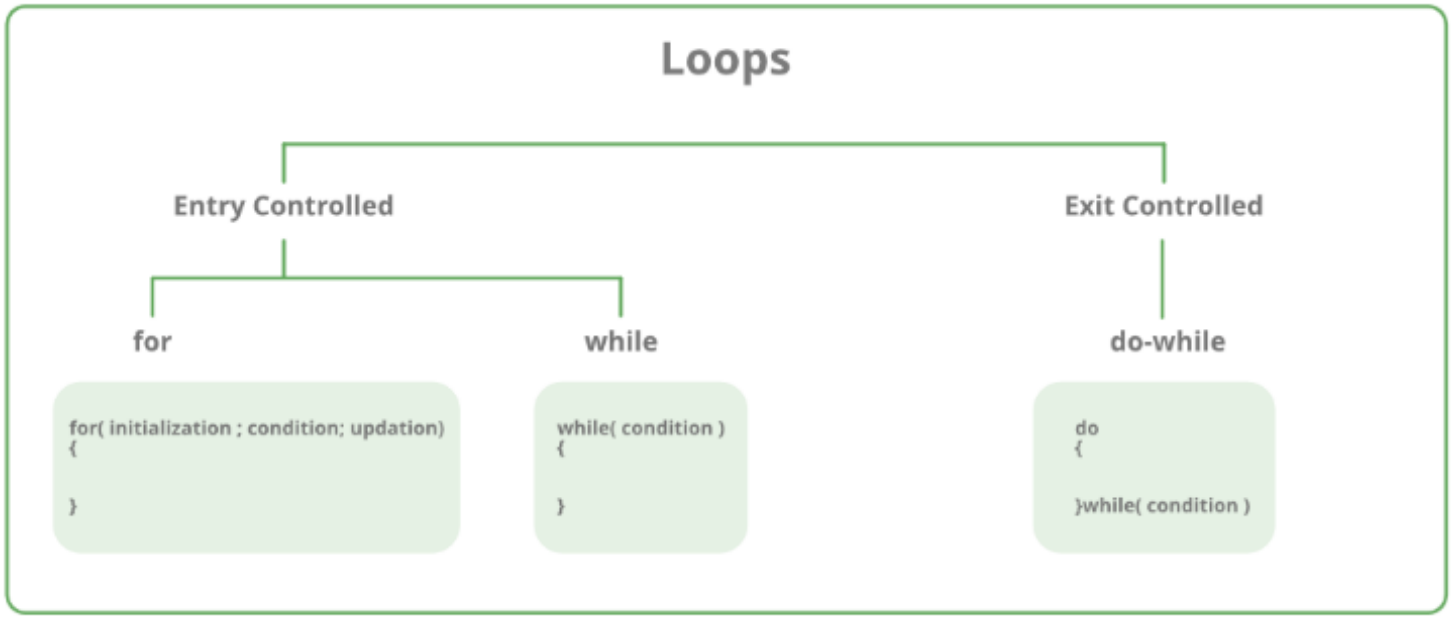
Overview

- **The do-while Loop**
  - What is do-while loop
  - Syntax of do-while loop
  - How do-while loop works
  - Flow Chart of do-while loop
  - Code Example

Types of Loops

There are mainly two types of loops:

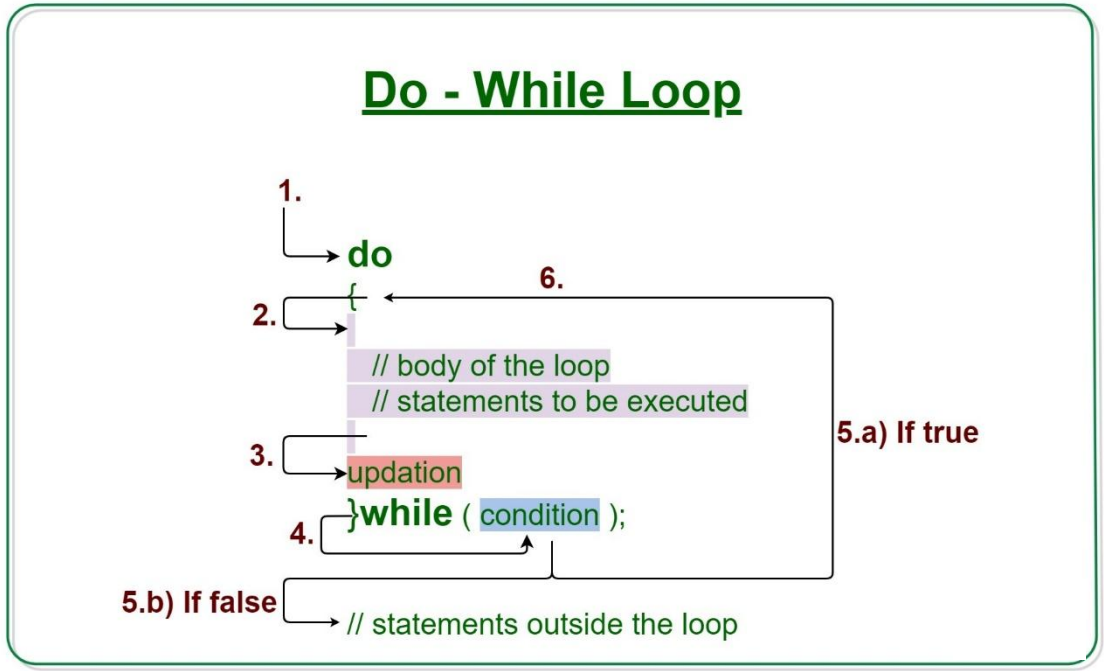
1. **Entry Controlled loops**: In this type of loop, the test condition is tested before entering the loop body. **For Loop** and **While Loop** is entry-controlled loops.
2. **Exit Controlled Loops**: In this type of loop the test condition is tested or evaluated at the end of the loop body. Therefore, the loop body will execute at least once, irrespective of whether the test condition is true or false. the **do-while loop** is exit controlled loop.



The do-while Loop

What is do-while loop

- Loops come into use when we need to repeatedly execute a block of statements.
- Like **while** the **do-while loop** execution is also terminated on the basis of a test condition. The main difference between a do-while loop and a while loop is in the do-while loop the condition is tested at the end of the loop body, i.e do-while loop is exit controlled whereas the other two loops are entry-controlled loops.



The do-while Loop

Syntax of do-while loop

```
do {
        // loop body
        update_expression;
}
While(test_expression);
```

The various parts of the **do-while loop** are:

- **Test Expression:** In this expression, we have to test the condition. If the condition evaluates to true then we will execute the body of the loop and go to the update expression. Otherwise, we will exit from the while loop.
- **Update Expression:** After executing the loop body, this expression increments/decrements the loop variable by some value.
- **Body:** It is the Collection of statements i.e, variables and functions, etc. The condition is not satisfied until the condition is executed automatically after a successful iteration. do-while loop, code can be used to print simple names, execute complex algorithms, or perform functional operations.

The do-while Loop

How do-while loop works

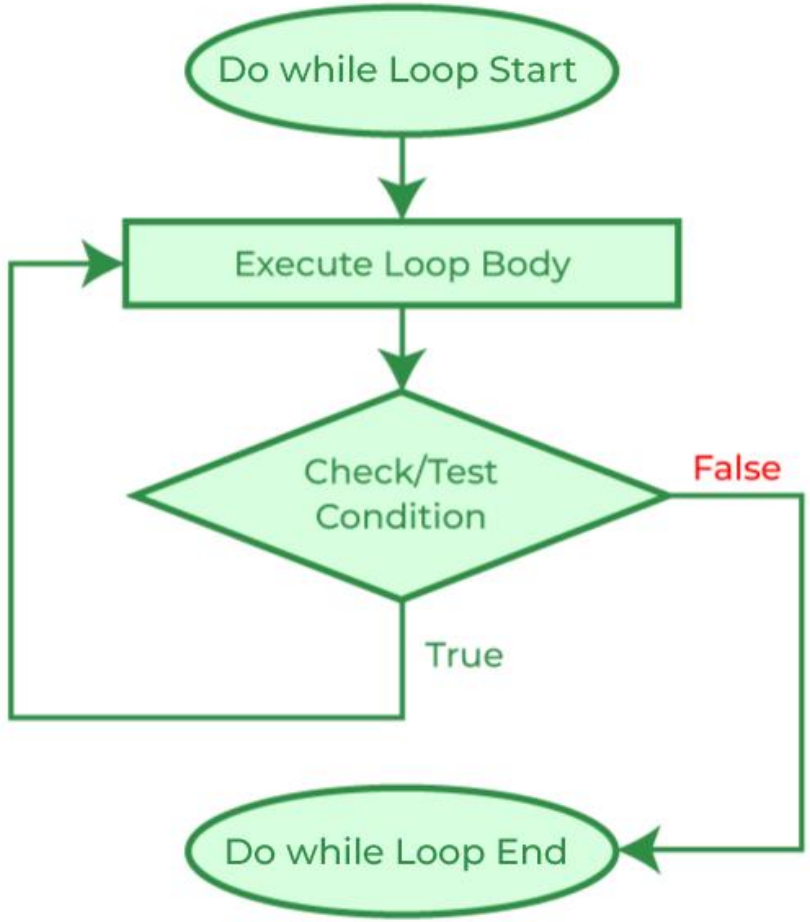
1. Control falls into the do-while loop.
2. The statements inside the body of the loop get executed.
3. Updation takes place.
4. The flow jumps to Condition
5. Condition is tested.
   - If the Condition yields true, go to Step 6.
   - If the Condition yields false, the flow goes outside the loop
6. The flow goes back to Step 2.
7. The do-while loop has been ended and flow has gone outside the loop.

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

int main() {
	int i = 0;
	do {
		cout<<i<<end;
		i++;

		} while(i<11);
}
```

The do-while Loop

Flow Chart of do-while loop

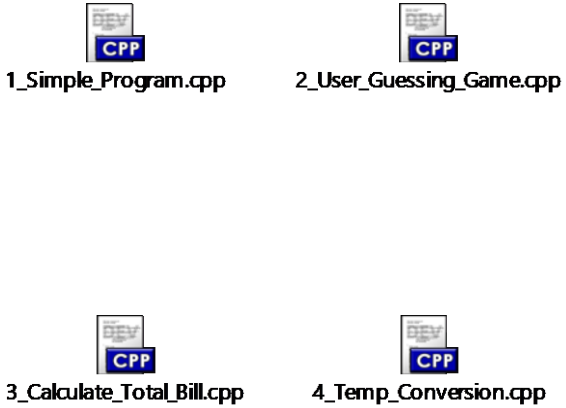
- 1. Start the program.
- 2. Execute the loop body inside the `do` block; this step happens unconditionally at least once.
- 3. Evaluate the condition specified in the `while` clause after executing the loop body.
- 4. If the condition is true, return to step 2 and repeat the loop body.
- 5. If the condition is false, exit the loop and continue with the rest of the program.
- 6. End the program.



The do-while Loop

Code Example

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

int main() {
    int number;
    do {
        cout << "Enter a number (positive to continue, negative to
quit): ";
        cin >> number;
        cout << "You entered: " << number << endl;
    } while (number >= 0);

    cout << "Loop terminated because you entered a negative
number." << endl;
    return 0;
}
```



Thank You