Lab # 7: More Loops

EC-102 – Computer Systems and Programming

Usman Ayub Sheikh

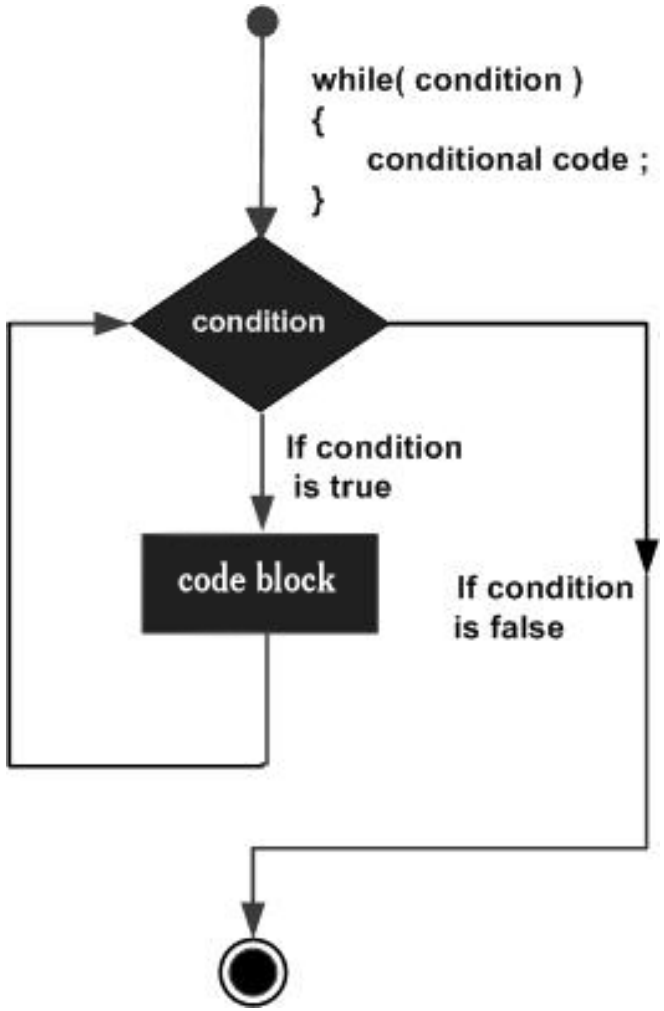
School of Mechanical and Manufacturing Engineering (SMME),
National University of Sciences and Technology (NUST)

October 12, 2015

Outline

The `while` Loop

- The `for` loop does something a fixed number of times
- What happens if we don't know how many times are we going to do something before we start the loop?



The `while` Loop – Syntax

```
while(test)
    {
        statement;
        statement;
        statement;
        statement;
    }
```

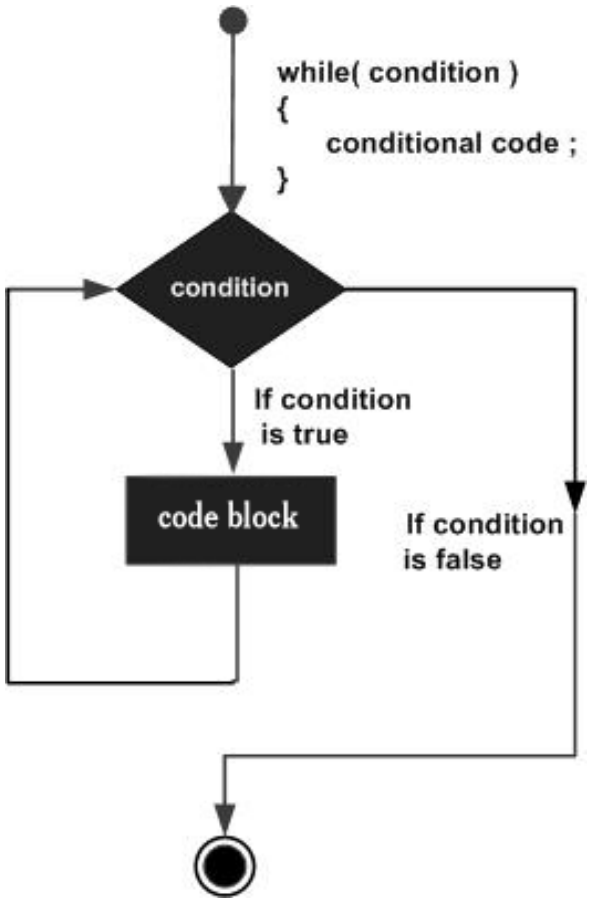
- Keyword `while` followed by a pair of parentheses that contain a test expression
- Although there is no initialization expression, the loop variable must be initialized before the loop begins
- The body of the loop, delimited by the left and right braces, is the code to be executed each time through the loop
- Similarly, the loop body must also contain some statement that keeps updating the value of the loop variable

The while Loop – Solved Example 1

```
1  // demonstrates WHILE loop
2  #include <iostream>
3  using namespace std;
4  int main()
5  {
6      int num, eoi = -1;
7      cout << "Enter a number: ";
8      cin >> num;
9
10     while(num != eoi)
11     {
12         if(num % 2 == 0)
13             cout << "The number is even.\n";
14         else
15             cout << "The number is odd.\n";
16         cout << "Enter a number: ";
17         cin >> num;
18     }
19     return 0;
20 }
```

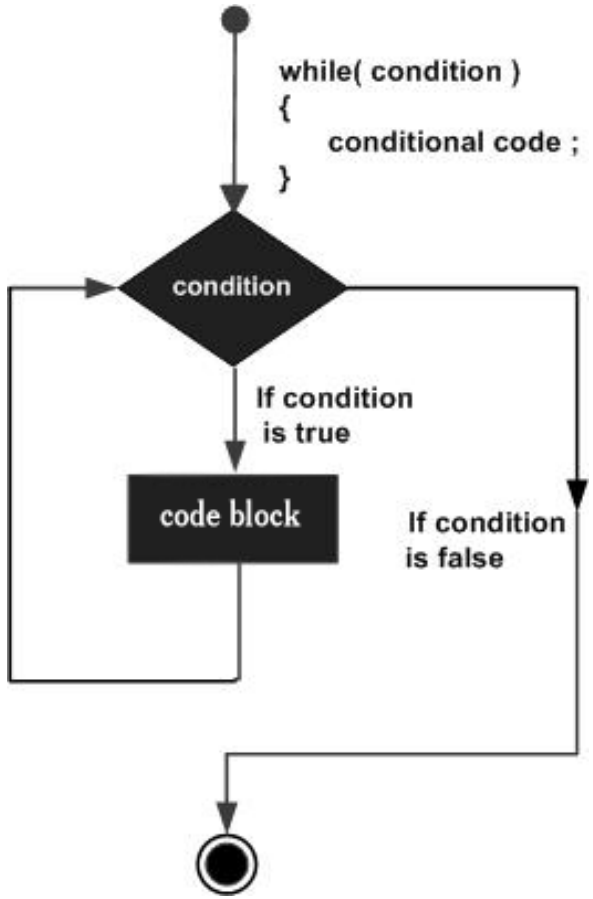
The while Loop – Solved Example 2

```
1  // sum using while loop (buggy)
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7      int num = 0, sum = 0;
8      while(num != -1)
9      {
10         cout << "Enter a number: ";
11         cin >> num;
12
13         sum = sum + num;
14     }
15     cout << "Sum = " << sum << endl;
16
17     return 0;
18 }
```



The while Loop – Solved Example 2

```
1  // sum using while loop (fixed)
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7      int num, sum = 0;
8      cout << "Enter a number: ";
9      cin >> num;
10
11     while(num != -1)
12     {
13         sum = sum + num;
14
15         cout << "Enter a number: ";
16         cin >> num;
17     }
18     cout << "Sum = " << sum << endl;
19     return 0;
20 }
```



Exercise

Write a calculator using `while` loop which keeps on getting a number and an operator as an input until 'q' is entered as an operator. As soon as 'q' is entered, the total result of the calculation is displayed and the execution of the program is stopped.

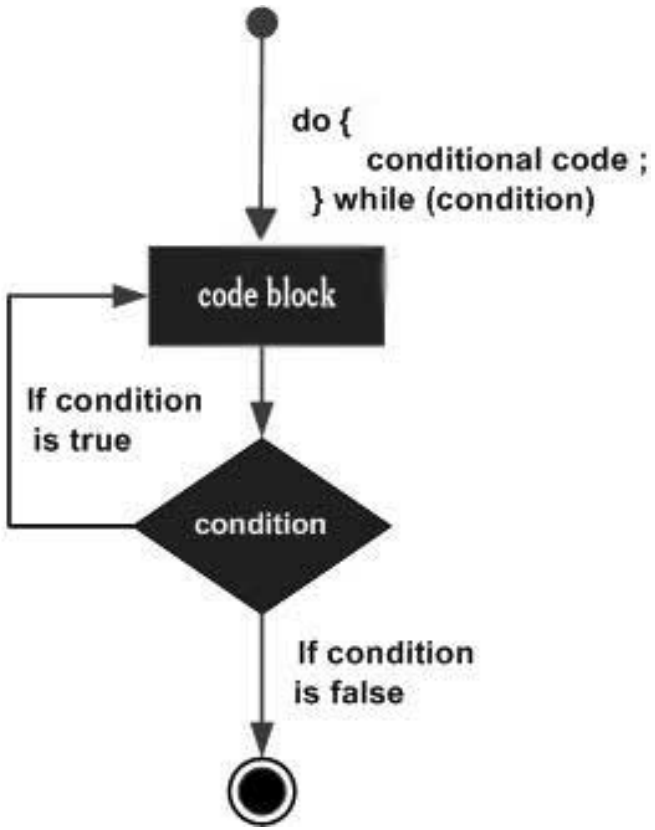
Here goes a sample interaction with the program:

```
Number: 5
Operator: +
Number: 5
Operator: *
Number: 3
Operator: -
Number: 2
Operator: /
Number: 2
Operator: q

Answer = 14
```

The do Loop

- In some situations, you want the test expression to be evaluated at the beginning of the loop
- But sometimes you want to guarantee that the loop is executed atleast once, no matter what the initial state of the test expression
- When such is the case, do loop should be used



The do Loop – Syntax

```
1 do
2     {
3         statement;
4         statement;
5         statement;
6         statement;
7     }
8 while(test);
```

- Keyword do marks the beginning of the loop
- As with other loops, braces delimit the body of the loop
- Finally, a `while` statement provides the test expression and terminates the loop

The do Loop – Solved Example

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int dividend, divisor;
6      char ch;
7      do
8          {
9              cout << "Enter dividend: ";
10             cin >> dividend;
11             cout << "Enter divisor: ";
12             cin >> divisor;
13             cout << "Quotient is " << dividend / divisor;
14             cout << ", remainder is " << dividend % divisor;
15             cout << "\nDo another? (y/n): ";
16             cin >> ch;
17         }
18     while(ch != 'n');
19 return 0;
20 }
```

Exercise 1

Write a program using do loop that repeatedly asks for a number and calculates its factorial until the user enters 0, at which point it terminates.

Here goes a sample interaction with the program:

```
Enter a number: 7
The factorial of the number is: 5040
Enter a number: 6
The factorial of the number is: 720
Enter a number: 5
The factorial of the number is: 120
Enter a number: 0
The factorial of the number is: 1
```

Exercise 2

Write a program using `do` loop that prints first `m` odd numbers that are not divisible by `n`.

Here goes a sample interaction with the program:

```
No. of odd numbers: 10
that are not divisible by: 3
1
5
7
11
13
17
19
23
25
29
```

When to Use Which Loop?

- Which loop type to use is more a matter of style than of hard-and-fast rules
- The `for` loop is appropriate when you know in advance how many times the loop will be executed
- The `while` and `do` loops are used when you don't know the number of times a loop will be executed
  - The `while` loop when you may not want to execute the loop body even once
  - The `do` loop when you're sure you want to execute the loop body at least once