While Loop in Java

Looping in programming languages is a feature that facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true. Java provides three ways for executing the loops. While all the ways provide similar basic functionality, they differ in their syntax and condition-checking time. In this article, we will see while loop in Java.

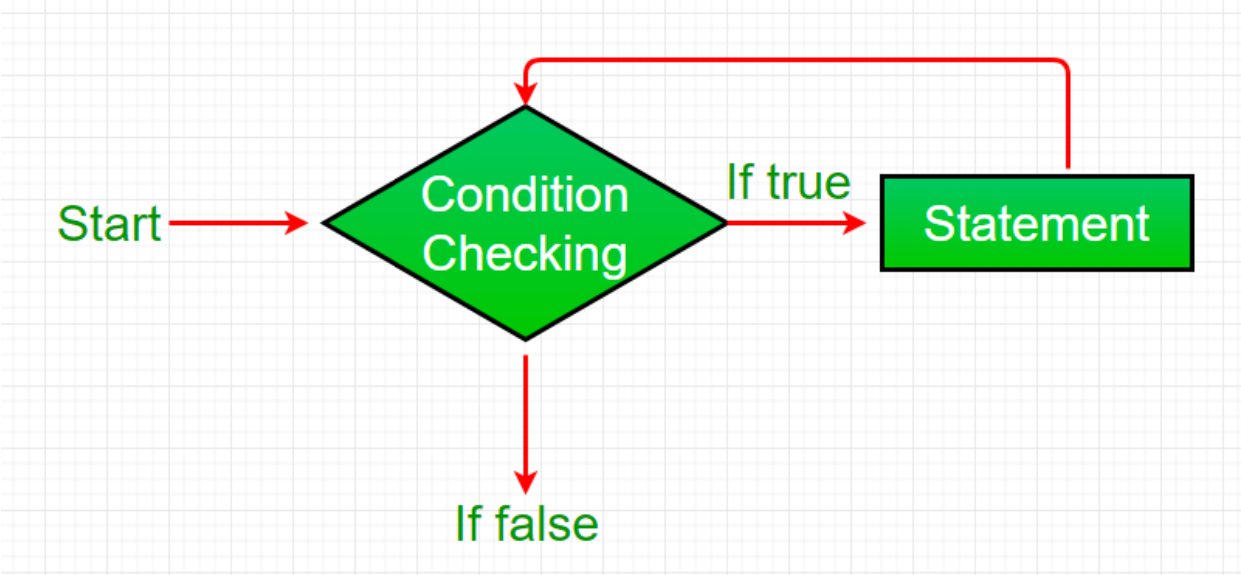
while loop

A while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.

Syntax :

```
while (boolean condition)
{
loop statements...
}
```

Flowchart :



- While loop starts with the checking of condition. If it is evaluated to be true, then the loop body statements are executed otherwise first statement following the loop is executed. For this reason, it is also called the Entry control loop
- Once the condition is evaluated to be true, the statements in the loop body are executed. Normally the statements contain an updated value for the variable being processed for the next iteration.
- When the condition becomes false, the loop terminates which marks the end of its life cycle.

```
// JAVA Code to print GFG 5 times

// using while loop

import java.io.*;

class GfG {

    public static void main(String[] args) {
```

```
        int i = 0;
        while (i < 5) {
            System.out.println("GFG");
            i++;
        }
    }
}
```

Output

```
GFG
GFG
GFG
GFG
GFG
```

Infinite loop: One of the most common mistakes while implementing any sort of looping is that it may not ever exit, that is the loop runs for infinite time. This happens when the condition fails for some reason.

```
//Java program to illustrate various pitfalls.
```

```
class GfG {

    public static void main(String[] args) {


        // infinite loop because condition is not apt

        // condition should have been i>0.

        for (int i = 5; i != 0; i -= 2) {

            System.out.println(i);

        }

        int x = 5;


        // infinite loop because update statement

        // is not provided.

        while (x == 5) {

            System.out.println("In the loop");

        }

    }

}
```

Another pitfall is that you might be adding something into you collection object through loop and you can run out of memory.