For 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. Loops are useful to do something repeatedly, for example, printing a statement multiple times or printing multiple dots so that they can make a line. Loops are also used to iterate through multiple items stored in containers like arrays, ArrayList, etc., or to run some services indefinitely like web servers and operating system services. In this article, we will see for loops in Java.

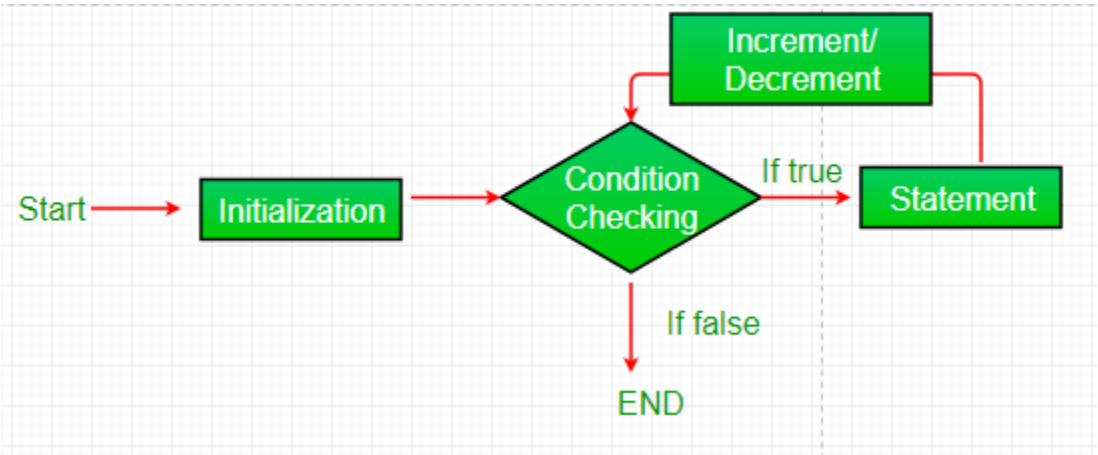
for loops:

for loop provides a concise way of writing the loop structure. Unlike a while loop, a for statement consumes the initialization, condition, and increment/decrement in one line thereby providing a shorter, easy-to-debug structure of looping.

Syntax:

```
for (initialization condition; testing condition; increment/decrement)
{
statement(s)
}
```

Flowchart:



- Initialization condition: Here, we initialize the loop variable in use. It marks the start of a for loop. An already declared variable can be used or a variable can be declared, local to loop only.
- Testing Condition: It is used for testing the exit condition for a loop. It must return a boolean value. It is also an Entry Control Loop as the condition is checked prior to the execution of the loop statements.
- Statement execution: Once the condition is evaluated to be true, the statements in the loop body are executed.
- Increment/ Decrement: It is used for updating the variable for the next iteration.
- Loop termination: When the condition becomes false, the loop terminates marking the end of its life cycle.

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

import java.io.*;

class GfG {

    public static void main(String args[]) {

        for(int i = 0; i < 5; i++) {

            System.out.println("GFG");

        }
```

```
    }
}
```

Output

```
GFG

GFG

GFG

GFG

GFG
```