Day 6 assignmt

Q

Ams: [**Skip to content**](https://data-flair.training/blogs/loops-in-java/#content)

* [Blog Home](https://data-flair.training/blogs/)

* [Data Science](https://data-flair.training/blogs/loops-in-java/)

* [Categories](https://data-flair.training/blogs/loops-in-java/)

* [Courses](https://data-flair.training/blogs/loops-in-java/)

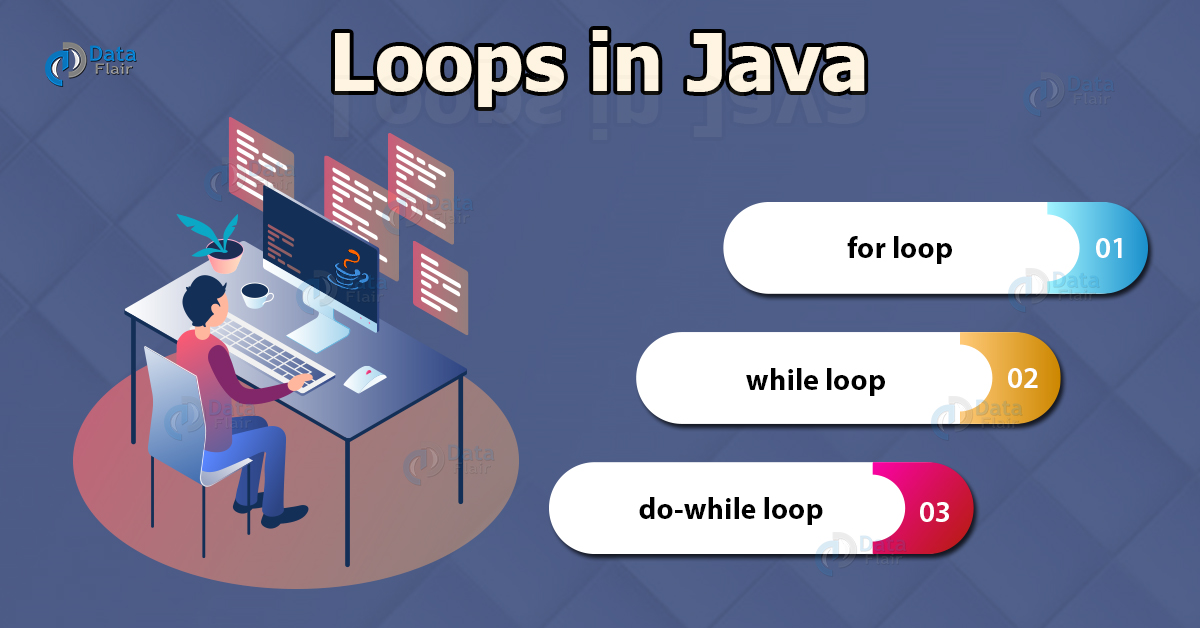
Top of Form

Search for:

Bottom of Form

Loops in Java (for, while, do-while) – Faster Your Coding with Easy Method

Loops are a fundamental concept in programming. In order to understand what loops are, we have to look at some real life cases of loops. Consider your daily routine, you wake up, you brush, you wear clothes and then head off to work, come back, eat and then sleep off. Again the next day, you do the same things in the same order and go to sleep again. This cycle keeps on repeating. This concept of repetitive actions performed time and again is called a loop. In this java tutorial, we will learn about the loops in java.

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2018/01/Loops-in-Java-df.jpg)

Need of Loops in Java

Imagine a program which is required to output a particular value of a variable 800 times. Now we all know that the code for writing output is System.out.println(“Text”); But in order to print this 800 times we will need to write the same line 800 times in the code. That would take up a lot of effort which is particularly just copy pasting the same sentence 800 times. Let us say that you have managed to copy paste the entire thing easily. Now if there is a different program which requires you to print the first 800 natural numbers. The copy paste method would not work because you still would have to go to all these lines and fit a number. That’s where loops come in to play. Loops make it very easy to group all the code that’s needed to be repetitively processed and throw it under scope. The loop does the remaining job.

Loops in Java

***Looping in Java is defined as performing some lines of code in an ordered fashion until a condition is false.*** The condition is important because we do not want the loop to be running forever. As soon as this condition is false, the loop stops.  
In Java there are three primary types of loops:-

**1.** for loop  
**2.** Enhanced for loop  
**3.** while loop  
**4.** do-while loop

1. For loop in Java

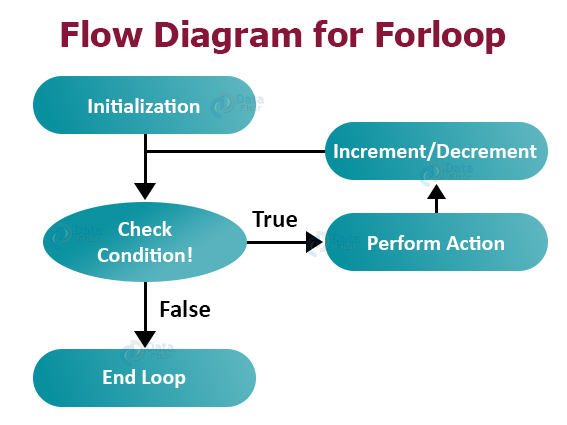
Java for loop consists of 3 primary factors which define the loop itself. These are the initialization statement, a testing condition, an increment or decrement part for incrementing/decrementing the control variable.

**The basic syntax of java for loop goes like this:**

for(initializing statement;testing condition;increment/decrement)  
{  
//code to be iterated  
}

The initializing statement marks the beginning of the loop structure. It contains a variable with some initial value that is defined by the programmer. This is the value of the control variable when the control shifts into the loop. However this statement is executed only once.  
If a particular variable is declared within this part, then the scope of the variable remains limited within the loop.

**The control flow can be explained with a flowchart diagram:**

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2018/01/Flow-Diagram-for-forloop.jpg)

**An example Java program explaining the functioning of a for loop:**

package com.dataflair.loops;

import java.io. \* ;

public **class** ForLoop {

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

int i;

**for** (i = 0; i <= 10; i++) {

System.out.println("Studying for loops at DataFlair");

System.out.println("Value of i = " + i);

}

}

}

**Output**

Studying for loops at DataFlair  
Value of i = 0  
Studying for loops at DataFlair  
Value of i = 1  
Studying for loops at DataFlair  
Value of i = 2  
Studying for loops at DataFlair  
Value of i = 3  
Studying for loops at DataFlair  
Value of i = 4  
Studying for loops at DataFlair  
Value of i = 5  
Studying for loops at DataFlair  
Value of i = 6  
Studying for loops at DataFlair  
Value of i = 7  
Studying for loops at DataFlair  
Value of i = 8  
Studying for loops at DataFlair  
Value of i = 9  
Studying for loops at DataFlair  
Value of i = 10

Observe the value of the loop’s control variable i as it increases from 0 to 10 and then stops because the value of i becomes 11 and the condition becomes false. Hence it goes out of the loop.

a. Enhanced for loop in Java

This is similar to a for loop except that it has some enhanced features. It can be used to iterate over the elements of a collection without knowing the index of each element. You also need not know the size of the collection.

However there are some limitations to this enhanced for loop.This loop object is immutable which means the values which are retrieved during the execution of the loop are read only. You cannot change the values in the collection while using this loop which is easily done by other loops.

**The syntax for Java enhanced for loop is:**

for(<datatype> <variable\_name>:<collection\_name>)  
{  
//Statements;  
}

**Java program to illustrate the use of enhanced for-loop:**

package com.dataflair.loops;

import java.io. \* ;

public **class** EnhancedFor {

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

String array[] = {

"DataFlair",

"Java",

"Python"

};

**for** (String a: array) {

System.out.println(a);

}

}

}

**Output:**

DataFlair  
Java  
Python

2. While loop in Java

While loops are very important as we cannot know the extent of a loop everytime we define one. For example if we are asked to take a dynamic collection and asked to iterate through every element, for loops would be impossible to use because we do not know the size of the collection. Then we would have to use an enhanced for loop or a while loop.

A while loop iterates through a set of statements till its boolean condition returns false. As long as the condition given evaluates to true, the loop iterates.

The condition of the loop structure is checked at first and then the control proceeds into the loop structure only if the condition evaluates to true. Hence it is called an entry-controlled loop. The body of the loop generally contains a variable which controls the boolean condition mentioned.

**The basic syntax of Java while loop is:**

while(boolean condition)  
{  
//statements;  
}

As soon as the condition hits false, the loop terminates.

If you are still confused about the working flow of the while loop, refer to the flowchart below.

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2018/01/Flow-Diagram-for-Whileloop.jpg)

**An example java program to illustrate the use of a while loop:**

package com.dataflair.loops;

import java.io. \* ;

public **class** WhileLoop {

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

int i = 0;

**while** (i < 5) {

System.out.println("Learning Java at DataFlair");

System.out.println("The value of i is = " + i);

i++;

}

System.out.println("The value of i became " + i + " that is why it broke out of the loop");

}

}

**Output:**

Learning Java at DataFlair  
The value of i is = 0  
Learning Java at DataFlair  
The value of i is = 1  
Learning Java at DataFlair  
The value of i is = 2  
Learning Java at DataFlair  
The value of i is = 3  
Learning Java at DataFlair  
The value of i is = 4  
The value of i became 5 that is why it broke out of the loop

3. do while loop in Java

Java do while loop executes the statement first and then checks for the condition.Other than that it is similar to the while loop. The difference lies in the fact that if the condition is true at the starting of the loop the statements would still be executed, however in case of while loop it would not be executed at all.

This is an exit-controlled loop because of the fact that it checks the condition after the statements inside it are executed.

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2018/01/Flow-Diagram-for-Dowhileloop.jpg)

**An example Java program to illustrate the use of do while loop:**

package com.dataflair.loops;

import java.io. \* ;

public **class** DoWhileLoop {

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

int i = 0;

**do** {

i++;

System.out.println("Learning Java at DataFlair");

System.out.println("The value of i is " + i);

}

**while** ( i != 5 );

}

}

**Output**

Learning Java at DataFlair  
The value of i is 1  
Learning Java at DataFlair  
The value of i is 2  
Learning Java at DataFlair  
The value of i is 3  
Learning Java at DataFlair  
The value of i is 4  
Learning Java at DataFlair  
The value of i is 5

Notice that even when the condition is false at i=5(bold), the statements still execute.

Nested Loops in Java

As the name suggests, nested loops are basically one loop functioning inside another one. After the first iteration of the outer loop starts, the inner loop starts. As soon as the innerloop finishes it’s iterations and exits, the first iteration of the outer loop completes and then it goes for the second iteration. This keeps on repeating till the outermost loop finishes its iterations.

However nested loops doesn’t necessarily mean two loops. You can include as many loops as you want inside one another. If there are two loops one inside another, one of them having **N** iterations and the other one having **M** iterations. Then the total number of iterations would be **M x N.**

**Basic syntax for a for loop inside a for loop:**

for(initializer;condition;increment/decrement)  
{  
for(initializer;condition;increment/decrement)  
{  
//code to be nested  
}  
}

**A java program to understand the concept of nested loops:**

package com.dataflair.loops;

public **class** NestedLoop {

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

int i,

j,

k;

k = 0;

**for** (i = 0; i < 6; i++) {

**for** (j = 0; j < 6; j++) {

System.out.print(k + "\t");

k++;

}

System.out.println("");

}

}

}

**Output**

0         1        2        3       4        5

6         7        8        9       10      11

12      13      14      15      16      17

18      19      20      21      22      23

24      25      26      27      28      29

30      31      32      33      34      35

This is a matrix(6×6) which is made by using a nested for loop.

Some Common Mistakes While Coding Loops

a. Infinite loop in Java

While designing loops, we can always commit mistakes like forgetting to update the condition variable or not defining a proper condition which leads to the loop being run infinite number of times. Different IDE’s have different mechanisms to stop live execution of code. It’s important to know your own IDE’s shortcut to stop live execution of a program incase you have the unfortunate incident of [java infinite loop](https://www.cs.amherst.edu/~jerager/cs23/doc/progguide/pitfalls-infiniteLoops.html).

**Example of infinite loop in java:**

package com.dataflair.loops;

import java.io. \* ;

public **class** InfiniteWhileLoop {

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

int i = 0;

**while** (i < 5) {

System.out.println("Learning Java at DataFlair");

System.out.println("The value of i is = " + i);

}

}

}

The program outputs infinite times of the statement and you can see the value of i remaining the same.

b. Running out of memory in Java

Sometimes due to improper planning we can overflow a collection with data using loops.This is due to poor planning and incorrect choice of collection or looping.

package com.dataflair.loops;

import java.util. \* ;

public **class** Overflowloop {

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

int i;

ArrayList < Integer > ar = **new** ArrayList < >();

**try** {

**for** (i = 0; i <= Integer.MAX\_VALUE; i++) {

ar.add(i);

}

} **catch**(Exception e) {

//TODO: handle exception

System.out.println(e);

}

}

}

**Output:**

Exception in thread “main” java.lang.OutOfMemoryError: Java heap space  
at java.base/java.lang.Integer.valueOf(Integer.java:1071)  
at Overflowloop.main(overflowloop.java:12)