**Java for Loop**

In computer programming, loops are used to repeat a block of code. For example, if you want to show a message 100 times, then rather than typing the same code 100 times, you can use a loop.

In Java, there are three types of loops.

* for loop
* [while loop](https://www.programiz.com/java-programming/do-while-loop#syntax-while)
* [do...while loop](https://www.programiz.com/java-programming/do-while-loop#do-while-loop)

This tutorial focuses on the for loop. You will learn about the other type of loops in the upcoming tutorials.

**Java for Loop**

Java for loop is used to run a block of code for a certain number of times. The syntax of for loop is:

for (initialExpression; testExpression; updateExpression) {

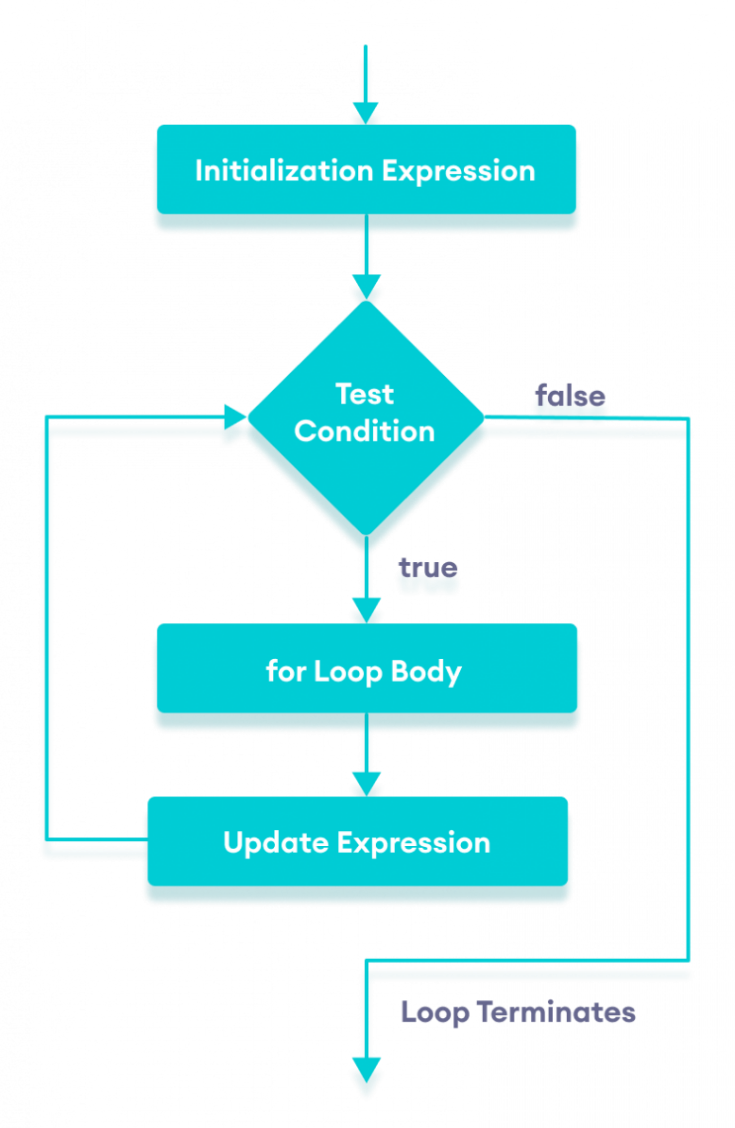
// body of the loop

}

Here,

1. The **initialExpression** initializes and/or declares variables and executes only once.
2. The **condition** is evaluated. If the **condition** is true, the body of the for loop is executed.
3. The **updateExpression** updates the value of **initialExpression**.
4. The **condition** is evaluated again. The process continues until the **condition** is false.

To learn more about the conditions, visit [Java relational](https://www.programiz.com/java-programming/operators#equality-relational) and [logical operators](https://www.programiz.com/java-programming/operators#logical).

Flowchart of Java for loop

**Example 1: Display a Text Five Times**

// Program to print a text 5 times

class Main {

public static void main(String[] args) {

int n = 5;

// for loop

for (int i = 1; i <= n; ++i) {

System.out.println("Java is fun");

}

}

}

[Run Code](https://www.programiz.com/java-programming/online-compiler)

**Output**

Java is fun

Java is fun

Java is fun

Java is fun

Java is fun

Here is how this program works.

|  |  |  |  |
| --- | --- | --- | --- |
| Iteration | Variable | Condition: i <= n | Action |
| 1st | i = 1 n = 5 | true | Java is fun is printed. i is increased to **2**. |
| 2nd | i = 2 n = 5 | true | Java is fun is printed. i is increased to **3**. |
| 3rd | i = 3 n = 5 | true | Java is fun is printed. i is increased to **4**. |
| 4th | i = 4 n = 5 | true | Java is fun is printed. i is increased to **5**. |
| 5th | i = 5 n = 5 | true | Java is fun is printed. i is increased to **6**. |
| 6th | i = 6 n = 5 | false | The loop is terminated. |

**Example 2: Display numbers from 1 to 5**

// Program to print numbers from 1 to 5

class Main {

public static void main(String[] args) {

int n = 5;

// for loop

for (int i = 1; i <= n; ++i) {

System.out.println(i);

}

}

}

[Run Code](https://www.programiz.com/java-programming/online-compiler)

**Output**

1

2

3

4

5

Here is how the program works.

|  |  |  |  |
| --- | --- | --- | --- |
| Iteration | Variable | Condition: i <= n | Action |
| 1st | i = 1 n = 5 | true | 1 is printed. i is increased to **2**. |
| 2nd | i = 2 n = 5 | true | 2 is printed. i is increased to **3**. |
| 3rd | i = 3 n = 5 | true | 3 is printed. i is increased to **4**. |
| 4th | i = 4 n = 5 | true | 4 is printed. i is increased to **5**. |
| 5th | i = 5 n = 5 | true | 5 is printed. i is increased to **6**. |
| 6th | i = 6 n = 5 | false | The loop is terminated. |

**Example 3: Display Sum of n Natural Numbers**

// Program to find the sum of natural numbers from 1 to 1000.

class Main {

public static void main(String[] args) {

int sum = 0;

int n = 1000;

// for loop

for (int i = 1; i <= n; ++i) {

// body inside for loop

sum += i; // sum = sum + i

}

System.out.println("Sum = " + sum);

}

}

[Run Code](https://www.programiz.com/java-programming/online-compiler)

**Output**:

Sum = 500500

Here, the value of sum is **0** initially. Then, the for loop is iterated from i = 1 to 1000. In each iteration, iis added to sum and its value is increased by **1**.

When i becomes **1001**, the test condition is false and sum will be equal to 0 + 1 + 2 + .... + 1000.

The above program to add the sum of natural numbers can also be written as

// Program to find the sum of natural numbers from 1 to 1000.

class Main {

public static void main(String[] args) {

int sum = 0;

int n = 1000;

// for loop

for (int i = n; i >= 1; --i) {

// body inside for loop

sum += i; // sum = sum + i

}

System.out.println("Sum = " + sum);

}

}

[Run Code](https://www.programiz.com/java-programming/online-compiler)

The output of this program is the same as the **Example 3**.

**Java for-each Loop**

The Java for loop has an alternative syntax that makes it easy to iterate through [arrays](https://www.programiz.com/java-programming/arrays) and collections. For example,

// print array elements

class Main {

public static void main(String[] args) {

// create an array

int[] numbers = {3, 7, 5, -5};

// iterating through the array

for (int number: numbers) {

System.out.println(number);

}

}

}

[Run Code](https://www.programiz.com/java-programming/online-compiler)

**Output**

3

7

5

-5

Here, we have used the **for-each loop** to print each element of the numbers array one by one.

In the first iteration of the loop, number will be 3, number will be 7 in second iteration and so on.

To learn more, visit [Java for-each Loop](https://www.programiz.com/java-programming/enhanced-for-loop).

**Java Infinite for Loop**

If we set the **test expression** in such a way that it never evaluates to false, the for loop will run forever. This is called infinite for loop. For example,

// Infinite for Loop

class Infinite {

public static void main(String[] args) {

int sum = 0;

for (int i = 1; i <= 10; --i) {

System.out.println("Hello");

}

}

}

[Run Code](https://www.programiz.com/java-programming/online-compiler)

Here, the test expression ,i <= 10, is never false and Hello is printed repeatedly until the memory runs out.