|  |  |  |  |
| --- | --- | --- | --- |
| **Comparison** | **for loop** | **while loop** | **do while loop** |
| Introduction | The Java for loop is a control flow statement that iterates a part of the [programs](https://www.javatpoint.com/java-programs) multiple times. | The Java while loop is a control flow statement that executes a part of the programs repeatedly on the basis of given boolean condition. | The Java do while loop is a control flow statement that executes a part of the programs at least once and the further execution depends upon the given boolean condition. |
| When to use | If the number of iteration is fixed, it is recommended to use for loop. | If the number of iteration is not fixed, it is recommended to use while loop. | If the number of iteration is not fixed and you must have to execute the loop at least once, it is recommended to use the do-while loop. |
| Syntax | for(init;condition;incr/decr){  // code to be executed  } | while(condition){  //code to be executed  } | do{  //code to be executed  }while(condition); |
| Example | //for loop  for(int i=1;i<=10;i++){  System.out.println(i);  } | //while loop  int i=1;  while(i<=10){  System.out.println(i);  i++;  } | //do-while loop  int i=1;  do{  System.out.println(i);  i++;  }while(i<=10); |
| Syntax for infinitive loop | for(;;){  //code to be executed  } | while(true){  //code to be executed  } | do{  //code to be executed  }while(true); |

## JaJava For Loop vs While Loop vs Do While Loop

# Java For Loop

The Java for loop is used to iterate a part of the program several times. If the number of iteration is fixed, it is recommended to use for loop.

There are three types of for loops in java.

* Simple For Loop
* [For-each](https://www.javatpoint.com/for-each-loop) or Enhanced For Loop
* Labeled For Loop

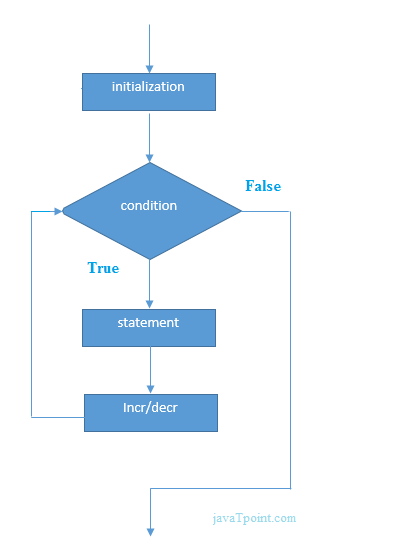
## Java Simple For Loop

A simple for loop is the same as [C](https://www.javatpoint.com/c-programming-language-tutorial)/[C++](https://www.javatpoint.com/cpp-tutorial). We can initialize the [variable](https://www.javatpoint.com/java-variables), check condition and increment/decrement value. It consists of four parts:

1. **Initialization**: It is the initial condition which is executed once when the loop starts. Here, we can initialize the variable, or we can use an already initialized variable. It is an optional condition.
2. **Condition**: It is the second condition which is executed each time to test the condition of the loop. It continues execution until the condition is false. It must return boolean value either true or false. It is an optional condition.
3. **Statement**: The statement of the loop is executed each time until the second condition is false.
4. **Increment/Decrement**: It increments or decrements the variable value. It is an optional condition.

**Syntax:**

1. **for**(initialization;condition;incr/decr){
2. //statement or code to be executed
3. }



1. /Java Program to demonstrate the example of for loop
2. //which prints table of 1
3. **public** **class** ForExample {
4. **public** **static** **void** main(String[] args) {
5. //Code of Java for loop
6. **for**(**int** i=1;i<=10;i++){
7. System.out.println(i);
8. }
9. }
10. }

## Java Nested For Loop

If we have a for loop inside the another loop, it is known as nested for loop. The inner loop executes completely whenever outer loop executes.

**Example:**

1. **public** **class** NestedForExample {
2. **public** **static** **void** main(String[] args) {
3. //loop of i
4. **for**(**int** i=1;i<=3;i++){
5. //loop of j
6. **for**(**int** j=1;j<=3;j++){
7. System.out.println(i+" "+j);
8. }//end of i
9. }//end of j
10. }
11. }

**Pyramid Example 1:**

1. **public** **class** PyramidExample {
2. **public** **static** **void** main(String[] args) {
3. **for**(**int** i=1;i<=5;i++){
4. **for**(**int** j=1;j<=i;j++){
5. System.out.print("\* ");
6. }
7. System.out.println();//new line
8. }
9. }
10. }

|  |  |
| --- | --- |
|  |  |
|  |  |

## Java for-each Loop

The for-each loop is used to traverse array or collection in java. It is easier to use than simple for loop because we don't need to increment value and use subscript notation.

It works on elements basis not index. It returns element one by one in the defined variable.

**Syntax:**

1. **for**(Type var:array){
2. //code to be executed
3. }

**Example:**

1. //Java For-each loop example which prints the
2. //elements of the array
3. **public** **class** ForEachExample {
4. **public** **static** **void** main(String[] args) {
5. //Declaring an array
6. **int** arr[]={12,23,44,56,78};
7. //Printing array using for-each loop
8. **for**(**int** i:arr){
9. System.out.println(i);
10. }
11. }
12. }

## Java Labeled For Loop

We can have a name of each Java for loop. To do so, we use label before the for loop. It is useful if we have nested for loop so that we can break/continue specific for loop.

Usually, break and continue keywords breaks/continues the innermost for loop only.

**Syntax:**

1. labelname:
2. **for**(initialization;condition;incr/decr){
3. //code to be executed
4. }

**Example:**

1. //A Java program to demonstrate the use of labeled for loop
2. **public** **class** LabeledForExample {
3. **public** **static** **void** main(String[] args) {
4. //Using Label for outer and for loop
5. aa:
6. **for**(**int** i=1;i<=3;i++){
7. bb:
8. **for**(**int** j=1;j<=3;j++){
9. **if**(i==2&&j==2){
10. **break** aa;
11. }
12. System.out.println(i+" "+j);
13. }
14. }
15. }
16. }

## Java Infinitive For Loop

If you use two semicolons ;; in the for loop, it will be infinitive for loop.

**Syntax:**

1. **for**(;;){
2. //code to be executed
3. }

**Example:**

1. //Java program to demonstrate the use of infinite for loop
2. //which prints an statement
3. **public** **class** ForExample {
4. **public** **static** **void** main(String[] args) {
5. //Using no condition in for loop
6. **for**(;;){
7. System.out.println("infinitive loop");
8. }
9. }
10. }

Output:

infinitive loop

infinitive loop

infinitive loop

infinitive loop

infinitive loop

ctrl+c

Now, you need to press ctrl+c to exit from the program.