How to Generate Random Number in Java

In **Java** programming, we often required to **generate random numbers** while we develop applications. Many applications have the feature to **generate numbers randomly**, such as to verify the user many applications use the **OTP**. The best example of random numbers is dice. Because when we throw it, we get a random number between 1 to 6.

In this section, we will learn what is a **random number** and **how to generate random numbers in**[**Java**](https://www.javatpoint.com/java-tutorial).

Random Number

Random numbers are the numbers that use a large set of numbers and selects a number using the mathematical algorithm. It satisfies the following two conditions:

* The generated values uniformly distributed over a definite interval.
* It is impossible to guess the future value based on current and past values.

Generating Random Number in Java

In Java, there is three-way to generate random numbers using the method and classes.

* Using the **random()** Method
* Using the **Random** Class
* Using the **ThreadLocalRandom** Class
* Using the **ints()** Method (in Java 8)

AD

Using the Math.random() Method

The Java **Math** class has many methods for different mathematical operations. One of them is the **random()** method. It is a **static** method of the Math class. We can invoke it directly. It generates only **double** type random number **greater than or equal to 0.0** and **less than 1.0**. Before using the random() method, we must import the java.lang.Math class.

**Syntax:**

1. **public** **static** **double** random()

It does not accept any parameter. It returns a pseudorandom double that is greater than or equal to 0.0 and less than 1.0.

Let's create a program that generates random numbers using the random() method.

**RandomNumberExample1.java**

1. **import** java.lang.Math;
2. **public** **class** RandomNumberExample1
3. {
4. **public** **static** **void** main(String args[])
5. {
6. // Generating random numbers
7. System.out.println("1st Random Number: " + Math.random());
8. System.out.println("2nd Random Number: " + Math.random());
9. System.out.println("3rd Random Number: " + Math.random());
10. System.out.println("4th Random Number: " + Math.random());
11. }
12. }

**Output:**

1st Random Number: 0.17434160924512265

2nd Random Number: 0.4297410090709448

3rd Random Number: 0.4828656381344487

4th Random Number: 0.13267917059488898

**Remember:** Every time we get a different output when we execute the program. Your output may differ from the output shown above.

We can also use the following formula if we want to a generate random number between a specified range.

1. Math.random() \* (max - min + 1) + min

In the above formula, the min value is inclusive while the max value is exclusive.

Let's create a program that generates random numbers between 200 to 400.

**RandomNumberExample2.java**

1. **public** **class** RandomNumberExample2
2. {
3. **public** **static** **void** main( String args[] )
4. {
5. **int** min = 200;
6. **int** max = 400;
7. //Generate random double value from 200 to 400
8. System.out.println("Random value of type double between "+min+" to "+max+ ":");
9. **double** a = Math.random()\*(max-min+1)+min;
10. System.out.println(a);
11. //Generate random int value from 200 to 400
12. System.out.println("Random value of type int between "+min+" to "+max+ ":");
13. **int** b = (**int**)(Math.random()\*(max-min+1)+min);
14. System.out.println(b);
15. }
16. }

**Output 1:**

Random value of type double between 200 to 400:

233.88329802655377

Random value of type int between 200 to 400:

329

**Output 2:**

Random value of type double between 200 to 400:

254.8419979875385

Random value of type int between 200 to 400:

284

Using the Random Class

Another way to generate a random number is to use the [Java **Random** class](https://www.javatpoint.com/post/java-random) of the java.util package. It generates a stream of pseudorandom numbers. We can generate a random number of any data type, such as integer, float, double, Boolean, long. If you are going to use this class to generate random numbers, follow the steps given below:

* First, import the class java.lang.Random.
* Create an **object** of the Random class.
* Invoke any of the following methods:
* **nextInt(int bound)**
* **nextInt()**
* **nextFloat()**
* **nextDouble()**
* **nextLong()**
* **nextBoolean()**

All the above methods return the next pseudorandom, homogeneously distributed value (corresponding method) from this random number generator's sequence. The **nextDouble()** and **nextFloat()** method generates random value between 0.0 and 1.0.

The **nextInt(int bound)** method accepts a parameter bound (upper) that must be positive. It generates a random number in the range 0 to bound-1.

Let's create a program that generates random numbers using the Random class.

**RandomNumberExample3.java**

AD

1. **import** java.util.Random;
2. **public** **class** RandomNumberExample3
3. {
4. **public** **static** **void** main(String args[])
5. {
6. // creating an object of Random class
7. Random random = **new** Random();
8. // Generates random integers 0 to 49
9. **int** x = random.nextInt(50);
10. // Generates random integers 0 to 999
11. **int** y = random.nextInt(1000);
12. // Prints random integer values
13. System.out.println("Randomly Generated Integers Values");
14. System.out.println(x);
15. System.out.println(y);
16. // Generates Random doubles values
17. **double** a = random.nextDouble();
18. **double** b = random.nextDouble();
19. // Prints random double values
20. System.out.println("Randomly Generated Double Values");
21. System.out.println(a);
22. System.out.println(b);
23. // Generates Random float values
24. **float** f=random.nextFloat();
25. **float** i=random.nextFloat();
26. // Prints random float values
27. System.out.println("Randomly Generated Float Values");
28. System.out.println(f);
29. System.out.println(i);
30. // Generates Random Long values
31. **long** p = random.nextLong();
32. **long** q = random.nextLong();
33. // Prints random long values
34. System.out.println("Randomly Generated Long Values");
35. System.out.println(p);
36. System.out.println(q);
37. // Generates Random boolean values
38. **boolean** m=random.nextBoolean();
39. **boolean** n=random.nextBoolean();
40. // Prints random boolean values
41. System.out.println("Randomly Generated Boolean Values");
42. System.out.println(m);
43. System.out.println(n);
44. }
45. }

**Output:**

Randomly Generated Integers Values

23

767

Randomly Generated Double Values

0.37823814494212016

0.998058172671956

Randomly Generated Float Values

0.87804186

0.93880254

Randomly Generated Long Values

-4974823544291679198

3650240138416076693

Randomly Generated Boolean Values

false

true

Using the ThreadLocalRandom Class

The **[ThreadLocalRandom](https://www.javatpoint.com/java-threadlocalrandom)**[class](https://www.javatpoint.com/java-threadlocalrandom) is defined in java.util.concurrent package. It is initialized with an internally generated seed, the same as the random generator of the Math class. It cannot be modified. We can use this class in the following way:

1. ThreadLocalRandom.current().nextX(...)

Where X is Int, Long, etc.

Note: It is impossible to share a ThreadLocalRandom with multiple threads accidentally.

We can generate a random number of any data type, such as integer, float, double, Boolean, long. If you are going to use this class to generate random numbers, follow the steps given below:

* First, import the class by using java.util.concurrent.ThreadLocalRandom.
* Invoke the corresponding method for which you want to generate numbers randomly.
* **nextInt()**
* **nextDouble()**
* **nextLong()**
* **nextFloat()**
* **nextBoolean()**

All the above methods override the corresponding method of the Random class and return the corresponding value.

* **nextInt(int bound)**
* **nextDouble(int bound)**
* **nextLong(int bound)**

The above methods parse a parameter **bound** (upper) that must be **positive**. It returns corresponding randomly generated value between 0 (inclusive) and the specified bound (exclusive). It throws **IllegalArgumentExcetion** if the bound is negative.

* **nextInt(int origin, int bound)**
* **nextDouble(int origin, int bound)**
* **nextLong(int origin, int bound)**

The above methods parse two parameters **origin** and **bound**. The origin specifies the **least value** returned and the bound specifies the **upper bound**. It returns corresponding randomly generated value between the specified origin (inclusive) and bound (exclusive). Also, throws **IllegalArgumentExcetion** if the origin is greater than or equal to bound.

Let's create a program that generates random numbers using the **ThreadLocalRandom class**.

**RandomNumberExample4.java**

1. **import** java.util.concurrent.ThreadLocalRandom;
2. **public** **class** RandomNumberExample4
3. {
4. **public** **static** **void** main(String args[])
5. {
6. // Generate random integers between 0 to 999
7. **int** a = ThreadLocalRandom.current().nextInt();
8. **int** b = ThreadLocalRandom.current().nextInt();
9. // Print random integer values
10. System.out.println("Randomly Generated Integer Values:");
11. System.out.println(a);
12. System.out.println(b);
13. // Generate Random double values
14. **double** c = ThreadLocalRandom.current().nextDouble();
15. **double** d = ThreadLocalRandom.current().nextDouble();
16. // Print random doubles
17. System.out.println("Randomly Generated Double Values:");
18. System.out.println(c);
19. System.out.println(d);
20. // Generate random boolean values
21. **boolean** e = ThreadLocalRandom.current().nextBoolean();
22. **boolean** f = ThreadLocalRandom.current().nextBoolean();
23. // Print random Booleans
24. System.out.println("Randomly Generated Boolean Values:");
25. System.out.println(e);
26. System.out.println(f);
27. }
28. }

**Output 1:**

Randomly Generated Integer Values:

348534891

-1887936727

Randomly Generated Double Values:

0.15644440033119833

0.5242730752133399

Randomly Generated Boolean Values:

true

true

**Output 2:**

Output 2:

Randomly Generated Integer Values:

402755574

295398333

Randomly Generated Double Values:

0.4856461791062565

0.5148677091077654

Randomly Generated Boolean Values:

false

true

Similarly, we can use other methods of **ThreadLocalRandomclass**.

Random Number Generation in Java 8

In Java 8, a new method ints() has been added to the [Random class](https://www.javatpoint.com/post/java-random). We must import the java.util.Random before using the method.

**ints():**

The pseudorandom int values generated the same as calling the nextInt() method. It returns an unlimited stream of pseudorandom int values.

**ints(long streamSize):**

The method parses a parameter **streamSize** of type long. It specifies the number of values to be generated. The pseudorandom int values generated the same as calling the nextInt() method. It also returns a stream of randomly generated int values. It throws **IllegalArgumentException** if the stream size is less than zero.

**ints(long streamSize, int randomNumberOrigin, int randomNumberBound):**

**Parameters:**

* **streamSize:**Number of values to generate.
* **randomNumberOrigin:**Origin of each random value
* **randomNumberBound:**Bound of each random value

It returns a stream of pseudorandom int values with the specified origin and bound. It throws **IllegalArgumentException** if:

* stramSize < 0
* origin > = bound

1. ints(**int** randomNumberOrigin, **int** randomNumberBound):

**Parameters:**

 **randomNumberOrigin:** Origin of each random value

 **randomNumberBound:** Bound of each random value

It returns an unlimited stream of pseudorandom int values with the specified origin and bound. It throws **IllegalArgumentException** if the origin is greater than or equal to bound.

Similarly, we can also generate the stream of long and double types by using the **longs()** and **doubles()** method, respectively.

Let's create a program that generates a stream of integers using the **ints()** method of the Random class.

**RandomNumberExample5.java**

1. **import** java.util.Random;
2. **public** **class** RandomNumberExample5
3. {
4. **public** **static** **void** main(String[] args)
5. {
6. randomInts(5);
7. randomInts(9, 50, 90);
8. //getStreamOfRandomInts(30, 50);
9. }
10. //method that generates a stream of integers having size 5
11. **public** **static** **void** randomInts(**int** num)
12. {
13. Random random = **new** Random();
14. random.ints(num).forEach(System.out::println);
15. }
16. //method that generates a stream of 9 integers between 50 to 90
17. **public** **static** **void** randomInts(**int** num, **int** origin, **int** bound)
18. {
19. Random random1 = **new** Random();
20. random1.ints(num, origin, bound).forEach(System.out::println);
21. }
22. }

**Output 1:**

727900357

-1073118456

306279822

370327182

1366265119

65

75

75

88

76

75

56

86

85

**Output 2:**

-1338107947

-1698771421

594232770

-1224937976

-1625069815

56

69

67

87

64

52

72

75

76