



ORACLE  
Academy



# Oracle Academy Java for AP Computer Science A

4-4

The Random Class

**ORACLE**  
Academy



# Objectives

- This lesson covers the following objectives:
  - Describe the purpose and uses of random numbers in Java programming
  - Identify methods of the Random class that obtain random numbers
  - Obtain random numbers in a range of numbers
  - Understand the purpose of the random number seed



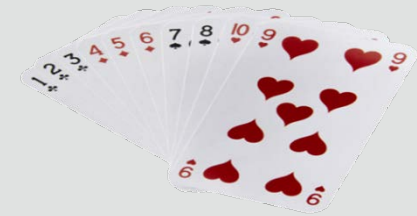
# Purpose of Random Number Generation in Java

- A software application often needs to perform a task based on some randomly obtained value
- A number of applications need generation of random numbers
- Let's look at some applications that use random number generation



# Applications Based on Random Number Generation

- A card game application needs to shuffle a deck of cards randomly and then randomly distribute the cards to the players
- A lottery application requires a randomly generated number that's based on an algorithm
  - The person wins if his number matches the randomly generated number





# Generating Random Numbers in Java

- So far in the previous lessons, you saw that Java comes with a variety of classes that support almost all basic application development features
- For example:
  - String provides the capability for manipulating strings
  - Scanner provides capability for obtaining input from the console
- Another important class in Java is the Random class that's used to obtain random numbers

# What Is the Random Class in Java?

- In Java, you use the Random class to obtain random numbers
- The class is located in the java.util package
- It contains several methods that return randomly obtained integer, double, boolean, float, and long type values

# How Do You Use the Random Class in a Java Program

- Import the Random class from the java.util package
- Create an instance of the Random class, like this:

```
import java.util.Random;
```

import statement to import the Random class from the java.util package

```
public class RandomIntNums {  
    public static void main(String[] args) {  
        Random rndNumber = new Random();  
    } //end method main  
} //end class RandomIntNums
```

Creates an instance of Random class, rndNumber



# Methods Provided by the Random Class

- You can obtain random values by invoking the following methods provided in the Random class:

Method	Produces
<code>boolean</code> <code>nextBoolean();</code>	A true or false value
<code>int</code> <code>nextInt();</code>	An integral value between <code>Integer.MIN_VALUE</code> and <code>Integer.MAX_VALUE</code>
<code>long</code> <code>nextLong();</code>	A long integral value between <code>Long.MIN_VALUE</code> and <code>Long.MAX_VALUE</code>
<code>float</code> <code>nextFloat();</code>	A decimal number between 0.0 (included) and 1.0 (excluded)
<code>double</code> <code>nextDouble();</code>	A decimal number between 0.0 (included) and 1.0 (excluded)

# How Do You Obtain a Random Number?

- You can obtain a random number of integer type by using the nextInt method
- For example:

```
import java.util.Random;
public class RandomNum {
    public static void main(String[] args) {
        Random rndNum = new Random();
        int randomNum = rndNum.nextInt();
        System.out.println("Random Number: " + randomNum);
    } //end method main
} //end class RandomNum
```

- Output:

```
Random Number: 1660093261
```

# How Do You Obtain a Series of Random Numbers?

- You can obtain a series of random numbers by calling the `nextInt` method several times
- For example:

```
public class RandomNumSeries {  
  
    public static void main(String[] args) {  
  
        Random num = new Random();  
        System.out.println("Random Number 1: " + num.nextInt());  
        System.out.println("Random Number 2: " + num.nextInt());  
        System.out.println("Random Number 3: " + num.nextInt());  
        System.out.println("Random Number 4: " + num.nextInt());  
        System.out.println("Random Number 5: " + num.nextInt());  
    } //end method main  
} //end class RandomNumSeries
```

`nextInt()` is called 5 times and  
so 5 random numbers are  
generated





# Generating Random Numbers of Double Type

- You can obtain random numbers of double type by using the `nextDouble` method, like this:

```
public class RandomDouble {  
    public static void main(String[] args) {  
        Random num = new Random();  
        double randomDouble = num.nextDouble();  
        System.out.println("Random Number: " + randomDouble);  
    } //end method main  
} //end class RandomDouble
```

- In this example, the `nextDouble` method returns numbers of the type `double` in the range of 0.0 to 1.0

# Exercise 1

- Create a new project and add the `FlipCoin.java` file
- Examine `FlipCoin.java`:
  - Execute the following program and observe the random number that chance generated
  - If  $\text{chance} < 0.5$ , record the result as "heads"; else record the result as "tails"
  - Repeat this many times



# Generating Random Numbers in a Range of Numbers

- So far, you have generated a random number within the range of an integer data type
- Sometimes, you may want to restrict the range of numbers that can be generated
- To implement this, you can use another version of the `nextInt` method:
  - `nextInt(int maxValue);`
    - The argument determines the highest integer that can be obtained by the `nextInt()` method
    - You can obtain random positive numbers from 0 (included) to a maximum (excluded) of your choice





# Generating Random Numbers in a Range of Numbers: Example

- Here's an example that obtains random numbers in the range of 0 to 19:

```
public class RandomNumRange {  
    public static void main(String[] args) {  
        Random num = new Random();  
        int randomnum = num.nextInt(20);  
        System.out.println("Random Number: " + randomnum);  
    } //end method main  
} //end class RandomNumRange
```

# Generating a Range Starting from 1

- To specify a range that starts with 1, add 1 to the result of the `nextInt()` method
- For example, to pick a number between 1 and 40 inclusively, add 1 to the result:

```
Random rand = new Random();  
int randomnum = rand.nextInt(40) + 1;
```

# Generating a Range Starting from a Higher Number Than 1

- If the range starts from a higher number than 1:
  - Subtract the starting number from the upper-limit number and then add 1
  - Add the starting number to the result of the `nextInt()` method
- For example, to pick a number from 5 to 35, inclusively:
  - The upper limit number will be  $35 - 5 + 1 = 31$  and 5 needs to be added to the result:

```
Random rand = new Random();  
int randomnum = rand.nextInt(31) + 5;
```

# Program for Lottery Application



```
public class Lottery {  
  
    public static void main(String[] args) {  
  
        Scanner numberScanner = new Scanner(System.in);  
        System.out.print("Enter a number between 1 and 10: ");  
        int userNum = numberScanner.nextInt();  
        Random rnd = new Random();  
        int winningNum = rnd.nextInt(10) + 1;  
        System.out.println("Your Number: " + userNum);  
        System.out.println("The winning number is: " + winningNum);  
    } //end method main  
  
} //end class RandomNumRange
```

## Exercise 2

- Create a new project and add the `RockPaperScissor.java` file to the project
- Examine `RockPaperScissor.java`
  - Perform the following:
    - Simulate the RockPaperScissor game by generating a random integer number in the range of 0 to 3
    - Compare the generated number with the following numbers:
      - if number=0 : "rock"
      - if number=1: "paper"
      - if number=2: "scissors"
    - Record the result and repeat many times

# Is the Same Random Number Generated Every Time?

- When you executed the previous examples multiple times, notice that the random number sequence is different each time
- Sometimes you may need to generate the same random number sequence every time





# What Is a Seed of a Random Number?

- You can achieve this by using a constant value called a seed
- When you create an instance of the Random class, pass a constant integer to specify the seed

```
Random rndNumbers = new Random(20L);
```



- You can change the seed by calling the setSeed() method
- Each time you pass the same seed, the same random sequence is returned

# Obtaining a Random Sequence by Using a Seed: Example

```
public static void main(String[] args) {  
    Random rand = new Random(20L);  
    System.out.println("Random Number 1: " + rand.nextInt(100));  
    System.out.println("Random Number 2: " + rand.nextInt(100));  
    System.out.println("Random Number 3: " + rand.nextInt(100));  
  
    System.out.println("Changing seed to change to sequence");  
    rand.setSeed(5L);  
    System.out.println("Random Number 4: " + rand.nextInt(100));  
    System.out.println("Random Number 5: " + rand.nextInt(100));  
    System.out.println("Random Number 6: " + rand.nextInt(100));  
  
    System.out.println("Setting seed 20 produce previous sequence");  
    rand.setSeed(20L);  
    System.out.println("Random Number 7: " + rand.nextInt(100));  
    System.out.println("Random Number 8: " + rand.nextInt(100));  
    System.out.println("Random Number 9: " + rand.nextInt(100));  
} //end method main
```



## Alternative to Random Class: Math.random()

- The **java.lang.Math.random()** method returns a pseudorandom (double type) number greater than or equal to 0.0 and less than 1.0
- When this method is called, it creates a new pseudorandom-number generator, the same as if using the Random class
- Enter the code from the slide note below
- Run the code several times to see how the results differ

# Summary

- In this lesson, you should have learned how to:
  - Describe the purpose and uses of random numbers in Java programming
  - Identify methods of the Random class that obtain random numbers
  - Obtain random numbers in a range of numbers
  - Understand the purpose of the random number seed





ORACLE  
Academy

