# Programming Exercise 03

*Authors: Ariel, Jack, Vince, Julia, Connor, Andrew, Brittney*

## Problem Description

Hello, and welcome! Please make sure to read all parts carefully.

This assignment will test your basic knowledge of Branching, Iteration, the Random class, and the Math class.

## Solution Description

Your program will consist of 3 files: `FloorOrMax.java`, `SquareRoot.java`, and `BigOrSmall.java`. A description for each class is given below. Please read all the steps and look at the Example Output carefully before you begin.

### Overall Notes

- Every print statement should be followed by a new line character such that every time something is printed it starts on a new line.
- In order to receive full credit from autograded tests, the `random` method from the `Math` class must generate the specified ranges of numbers in the **simplest way possible**.

### *FloorOrMax.java*

- Notes: You must use the `floor` and `max` methods found in the `Math` class for this part. Refer to the Java API for information on these two methods. Also, when we say 54.0 is exclusive, we mean that the resulting double should not be able to be 54.0 exactly, but it could be, say, 53.9999.
- Create a class called `FloorOrMax`, and inside this class create a `main` method. Inside the `main` method do the following:
    1. Using the `random` method found in the `Math` class, generate a random **floating-point** number between 4.0 **(inclusive)** and 54.0 **(exclusive)** and save it to a variable named `myRand`.
    2. Print out "The value is: [value]" rounded to 2 decimal places where [value] is replaced with `myRand` rounded.
    3. If the `floor` of `myRand` is even and greater then 15, floor it and print out the result as an integer.
    4. If the `floor` of `myRand` is odd, print out the maximum between myRand and the number 32.
    5. Otherwise, print out `myRand` as is.

### *SquareRoot.java*

- Notes: You must use the `Random` class (which can be imported from `java.util.Random`) for this part. Refer to the Java API for information on this class.
- Create a class called `SquareRoot`, and inside this class create a `main` method. Inside the `main` method do the following:
    1. Create a `Random` variable called `rand`.
    2. Using the `Random` object passed into the method, generate a random **whole number** between 1_000_000_000_000_000 (**15 zeros; inclusive**) and 1_000_000_000_000_300 **(inclusive**) and

save it into a variable named `bigRandNum`. **HINT**: Think about what data type is required to store a value of this size.

3. Print out `"The value is: [value]"` where [value] is replaced with `bigRandNum`.
4. Using a `while` loop, print out the square root of `bigRandNum`, and then the square root of that result, and so on successively until the result reaches the value of 1.
5. Determine how many iterations of the `while` loop were required to reduce this number to 1. After the while loop is done executing, print out `"Count: [count]"` (with the actual count replacing the bracketed section).

### *BigOrSmall.java*

- Create a class called `BigOrSmall`, and inside this class create a `main` method. Inside the `main` method do the following:
    1. Using the `random` method found in the `Math` class, generate a random whole number between 1 (**exclusive**) and 5 (**inclusive**) and save it to a variable named `smallRandNum`.
    2. Print out `"We will run [value] times!"` where [value] is replaced with the value of `smallRandNum`.
    3. Using the generated number, run a `for`-loop that the `smallRandNum` number of iterations.
    4. In each iteration of the for-loop, use the `random` method from the `Math` class to generate a random **whole number** between 1 (**exclusive**) and 20 (**inclusive**) and save it to a variable.
    5. Print out `"The value is: [value]"` (with the generated number replacing the bracketed section).
    6. If that number is greater than 10, print out `"Yay!"`.
    7. Otherwise, print out `"Nay!"`.

### *Example Outputs*

Example partial output for each of the three files is shown below.

`FloorOrMax` output:

```
The value is: 52.73
52
```

`SquareRoot` output:

```
The value is: 1000000000000111
31622776
5623
74
8
2
1
Count: 6
```

`BigOrSmall` output:

```
We will run 2 times!
The value is: 10
Nay!
The value is: 11
Yay!
```

## Import Restrictions

To prevent trivialization of the assignment, you may only import `java.util.Random`.

## Feature Restrictions

There are a few features and methods in Java that overly simplify the concepts we are trying to teach or break our autograder. For that reason, do not use any of the following in your final submission:

- `var` (the reserved keyword)
- `System.exit`

## Collaboration

No collaboration is allowed on this assignment. See syllabus for more details.

In addition, note that you are not allowed to upload your code to any sort of public repository. This could be considered an Honor Code violation, even if it is after the homework is due.

## Turn-In Procedure

To submit, upload the files listed below to the corresponding assignment on Gradescope:

- `FloorOrMax.java`
- `SquareRoot.java`
- `BigOrSmall.java`

Make sure you see the message stating "PE03 submitted successfully." From this point, Gradescope will run an autograder on your submission verifying all necessary files are included and working as intended. Note: the nature of some assignments means they are **not fully autogradable** and will require some manual grading after submission. Although all autograder tests will be made visible on this assignment, this is NOT always your final grade. You can submit as many times as you want before the deadline, so feel free to resubmit as you make progress on the homework. We will only grade your last submission be sure to **submit *every file* each time you resubmit.**

## Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit `.class` files
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- Check on Piazza for all official clarifications