System Requirements Specification Index

For

Random Number Generation

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

Table of Contents

1 Project Abstract

This assessment focuses on evaluating the understanding and ability to generate random numbers in Java. You need to use both the Math.random() method and the Random class to generate random numbers for different applications.

2 Assessment Tasks

Task 1: Generate a Random Decimal Number:

⟶ Generate a random decimal number between 0 and 1 using Math.random().

⟶ Example: Generate a random number between 0 and 1.

Task 2: Generate a Random Integer within a Range:

⟶ Generate a random integer within a given range using Random.nextInt().

⟶ Example: Generate a random integer between 1 and 100.

Task 3: Generate a Random Floating-Point Number:

⟶ Generate a random floating-point number between a given range using Random.nextDouble().

⟶ Example: Generate a random floating-point number between 5.5 and 20.5.

Task 4: Simulate a Coin Toss:

⟶ Use random number generation to simulate a coin toss, returning "Heads" or "Tails".

⟶ Example: Generate a random number (0 or 1) and print "Heads" for 0 and "Tails" for 1.

Task 5: **Random Selection from an Array:**

→ Select a random element from a given array of strings or numbers.

→ Example: Given an array of colors, select a random color.

3 Template Code Structure

3.1 Package: com.yaksha.assignment.RandomNumberGenerationAssignment

Resources

Class/Interface	**Description**	**Status**
RandomNumberGenerationAssignment (class)	● Main class demonstrating random number generation using `Math.random()` and `Random` class. Includes examples of generating random decimals, integers, floating-point numbers, simulating coin toss, and selecting random elements from an array.	Need to be implemented.

4 Execution Steps to Follow

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) □ Terminal □New Terminal.**
3. **This editor Auto Saves the code.**
4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**
5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
6. **To run your project use command:**

   ```
   mvn compile exec:java
   -Dexec.mainClass="com.yaksha.assignment.RandomNumberGenerationAssignment"
   ```

7. **To test your project test cases, use the command**

   ```
   mvn test
   ```

8. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**