**CMSC203**

**Assignment #2**

**Random Number Guesser**

**Assignment Description**

Build an application that will receive a guess and report if your guess is the random number that was generated. Your application will narrow down the choices according to your previous guesses, and continue to prompt you to enter a guess until you guess correctly.

Notice that if you divide the choices in half each iteration, you will need at most log2(100) ~ 7 guesses.

You will use the utility class RNG.java.

**Concepts tested by this assignment**

* A driver and a utility class
* UML class diagrams or pseudo-code
* Java fundamentals, including decision structures, loops
* Selection control statements
* Repetition control statements
* Input validation loops (in RNG.java)
* Relational and logical operators
* Random number generation (in RNG.java)

**Classes**

**Data Element Class – RNG**

* Provided
* This file will generate a random number between 1 and 100
* Note that the method “rand” is a static method, so the java file does not need to be instantiated to use it. Call rand with the following: RNG.rand(100) to generate a random number between 0 and 99.
* Study this class. You will want to use four methods from this class: rand, resetCount, getCount and inputValidation.

**Driver Class – RandomNumberGuesser**

* Student created
* This is the driver class for RNG that contains a main method.
* The driver is responsible to:
  + print a header
  + ask the user for an initial guess of the Random Number between 0 and 100.
  + Print out the result for that guess using the methods from the RNG class.
  + Allow user to give another guess between the previous low and high guesses.
  + Display the number of guesses
  + When user guesses correctly, ask if the user wants to try another round.
  + Print the Programmer's name at the end
  + Refer to the program sample run for more clarification.
* Data Validation. The following data is validated by the RNG method inputValidation:
  + Guesses must be an integer between the previous low guess and high guess.
* Naming.
  + There should be an attribute named randNum
  + There should be attributes named nextGuess, highGuess, and lowGuess
* Add any necessary methods to modularize your code.

**Assignment Details**

Your program should respond similar to the following sample runs:

== Example Run 1 (one iteration) ===========

A screenshot of a cell phone

Description automatically generated

== Example Run 2 (one iteration, one guess over the limit) ===========A screenshot of text

Description automatically generated

== Example Run 3 (two iterations) ===========A screenshot of text

Description automatically generated

Take screenshots of two runs of your program. Your runs should include feedback on previous low and high guesses, at least one example of a guess not within the previous low and high guesses, and an example of a second try with a new random number. You may use either an IDE or command-line processing. Store your files in the repo you created in GitHub in Lab 1, and take a screenshot.

**Deliverables**

**Deliverables / Submissions:**

Design: UML class diagram or pseudo-code (algorithm)

Implementation: Submit a compressed file containing the follow: The Java application (it must compile and run correctly); Javadoc files in a directory; a write-up as specified below. Be sure to review the provided project rubric to understand project expectations. The write-up will include:

* Screenshots of at least two runs of your code. Your runs should include feedback on previous low and high guesses, at least one example of a guess not within the previous low and high guesses, and an example of a second try with a new random number.
* Final design: UML diagram or pseudo-code (revised from initial design if necessary) – as part of write-up or submitted separately, as desired.
* Javadoc
* In three or more paragraphs, highlights of your learning experience
* Screen shot of Java code in your GitHub repo

**Deliverable format:** The above implementation deliverables will be packaged as follows. Two compressed files in the following formats:

* LastNameFirstName\_Assignment2\_Complete.zip, a compressed file in the zip format, with the following:
  + Write up (Word document) - reflection paragraphs, screenshots
  + Final design: UML Diagram or pseudo-code (Word or image document)
  + doc (directory) – Javadoc
    - * sub-directories as they exist
      * File1.html (example)
      * File2.html (example)
  + src (directory) 
    - * RandomNumberGuesser.java
      * RNG.java
* LastNameFirstName\_Assignment2\_Moss.zip, a compressed file containing one or more Java files:
  + RandomNumberGuesser.java

This folder should contain Java source files only