CS272/463 Lab Assignment #2.

Part 0: Read Chapter 2 of the textbook.

Part 1: Review the running time analysis from Section 1.2 Answer all the questions from the Running Time worksheet (the worksheet in docx format).

Part 2: Generating a series of random numbers can be achieved in Java by using the java.util.Random class. However, the numbers generated are not truly random, they are pseudorandom. That is, they are produced by algorithms that generate a fixed but random-looking sequence of numbers. In this assignment you will learn the basic ideas behind generating pseudorandom numbers and implement your own pseudorandom number generator. Knowing how pseudorandom numbers are generated and their limitations may help you in the future when you use pseudorandom numbers in simulations and other applications.

Do programming projects 11, 12, and 13 on pages 99-100 of the textbook. In project 13 you are asked to experiment with different values of the multiplier, increment, and modulus. Find your own values of constants (different from the ones in the book) that are good (which result in about 10% of the numbers in each interval). Also, find values of multiplier, increment, and modulus that are not good (which result in some intervals having no numbers).

Use Eclipse to create a new project with the name, say, Lab2. In the project create three new classes as described below.

- Your implementation should include the following 3 classes:

  1) **Pseudorandom.java** - implements a class that generates pseudorandom numbers. You must use the principle of information hiding in your implementation (instance variables must be private).

     Instance variables of the Pseudorandom class should be *multiplier*, *increment*, *modulus*, and *seed* (of type *int*).

     You need to implement the following **methods** (that are described in the projects 11 and 12):

     - constructor with initial seed, multiplier, increment, and modulus as parameters,
     - *changeSeed(int new_seed)* ,
     - *nextInt()* that generates and returns a pseudorandom integer number ,
     - *nextDouble()* that generates and returns a pseudorandom *double* number in the range [0..1) (project 12 from the textbook). Note that *modulus* must not be equal to 0 in order for *nextInt* and *nextDouble* methods to work.

  2) **TestPseudorandom.java** - tests Pseudorandom class. It should prompt the user to enter initial values for seed, multiplier, increment, and modulus, and include test calls to all of the constructors and methods.
  3) **TestDistribution.java** - determines distribution of numbers (project 13 from the textbook). It should prompt the user to enter initial values for seed, multiplier, increment, and modulus, and output a table with numbers of occurrences for each range.

- Use your TestDistribution class from project 13 to experiment with different values of the multiplier, increment, and modulus. Prepare a text file (*.txt file) which contains
  - good values of the multiplier, increment, and modulus that you found (should be different from the values provided in the textbook), together with the output of your TestDistribution program when run with these numbers, and

- not good values of the multiplier, increment, and modulus that you found, together with the output of your TestDistribution program when run with these numbers.

Mark which values are good and which are not good in the text file.

- **Specifications** for all your methods should be included as comments in your code. Method specifications should follow the guidelines from Section 1.1 (pp.7-8), that is, include a short introduction, parameter description, preconditions (if any), postcondition or returns condition, the throws list (if any). Also, please use inline comments, meaningful variable names, indentation, formatting and whitespace throughout your program to improve its readability. Tips for naming variables.

- Thoroughly test your code.

What to submit:

1. Submit your answers to the Running Time worksheet. You may print Running Time worksheet exercise, write your answers on it, scan, and submit as a pdf file. Alternatively, you may write you answers in the .docx file, save it as a pdf and submit.
2. Submit your source code (*.java files) electronically on Canvas.
3. Submit a text file (*.txt file) which contains good and not good values of the multiplier, increment, and modulus that you found, together with the output of your TestDistribution program when run with these numbers.