CSCE 350 Programming Assignment 2

--Due Date on Dropbox—

Please reread the instructions for how to submit – the graders would like just the files (Lomuto.java, SwitchGenerator.java, if you've done it completely, and RussianMultiply.java). The grader may deduct 5 points if you fail to upload properly.

The last problem, Switch Generator is worth +10 bonus points **if it is 100% functional.**

Please remember that we've modified the late policy: we will accept it up to one day, exactly 24 hours, late at -15% (-15 points). After that, no submissions will be accepted. Try to do this early.

Setup (General)

- Use Java version 8, update 111 (install Java First), might work fine with others. I believe you can
  - Make sure you update path to your java bin directory (for Windows)
  - Should be able to install different versions and have Eclipse use the right one
- Install Eclipse Version Neon.2 (https://eclipse.org/downloads/)

Import Project

- Download from CSE Dropbox the file (should be a .zip)
- In Ecplise: Follow these directions **exactly**:
  - File->Import->General->Existing Projects into Workspace (not Archive File)->Next
  - Choose "Select Archive" and browse for the project file
  - Click Finish
- (General Info) Under the **src** directory, open the default package and look at the files
  - Your changes will be to source files in this directory
  - Do Not modify MyTests (we probably won't run it… we'll just copy your source into our project to test)

RussianMultiplication

Implement the Russian Peasant multiplication algorithm in RussianMuliply.java.

Your solution can be iterative or recursive.

Lomuto Partitioning

Implement Lomuto Partitioning in Lomuto.java. Be careful to:

- Read 158-159 for details.
- Not change anything outside the partition bounds.
- Use the first item in the partition as the pivot (before partitioning)
- Return the index of the pivot at the end.

Switch Guesser/Generator (See problem 4.3.12) +10 Bonus for fully functional code (nothing if it doesn't work 100%)

Implement a class to determine the next switch to flip/toggle. The testing code creates an instance (which you code in SwitchGenerator.java) and then repeatedly asks for the next switch to toggle.

- SwitchGenerator is a primitive Generator (it doesn't implement all the functions in the right way that a real one would). All it does, with next() is give the 0-based index of the next switch to toggle.
- The SwitchGenerator constructor takes one parameter – the number of switches.
- Your SwitchGenerator doesn't know anything about the state of the game
- You need a minimal change approach – the tests are likely to fail if you cannot generate all the possibilities in under 2^n switches, where n is the number of switches.
- You will probably have to use Gray Codes. It's fine if you build a list in the Constructor.
- You will probably need to define some methods and members for SwitchGenerator.
- Read MyTests.java to see how it's used.

Export Project (for Submission) ←Changed from first time

- Upload each of your files individually.
    - RussianMultiply.java
    - Lomuto.java
    - And, if you got it working and passing the tests, SwitchGenerator.java