CSYE7215: Homework 1:
Due date: September 14, 2016; 5:59 pm

Part A:

Goal: In this project you will complete several classes to find the **maximum Integer** in a LinkedList **in parallel**. These classes should be **thread safe**.

- The stubs for the code are in ParallelMaximizer.java and Parallel and ParallelMaximizerWorker.java and the public tests are in the file PublicTest.java. They are in Course Documents / Code / HW 1 Code.

- The computation should be repeated ten times for the same LinkedList, i.e., the same randomly generated list should be used ten times, each time printing the final result.

Code to implement: In the starter files, the methods to implement have the comment // TODO: IMPLEMENT CODE HERE within them. The Javadoc explains what these methods should do for the max integer (which you can compare against the public test).

- Class ParallelMaximizer, method max
  Note: Since this method invokes several ParallelMaximizerWorkers, it is expected to be thread safe. The method runs numThread number of threads and then joins them. You are responsible for computing the partial maximum from these results by calling getPartialMax() from each ParallelMaximizerWorker.
- Class ParallelMaximizerWorker, method run
  Note: This method should find the maximum for all integers processed by this worker, which can be combined to find the overall maximum. If the list is not empty, the function synchronizes on it to prevent access by other threads and removes the head node, storing its value in the variable number. You are responsible for taking this value and evaluating the new partial maximum.
- Note: It is not acceptable that only one of the workers computes the maximum / average. Insure that all of the workers participate in this process. Show the results of each of the worker.

Part B:

Either modify or develop from scratch the "worker" and the "maximizer" code to implement four "inspectors", whose objectives are as follows:

1. Inspector named "Even": Collect only even numbers in the list.
2. Inspector named "Odd": Collect only odd numbers in the list.
3. Inspector named "Order": Collect only those numbers that are larger than the last collected number.

4. Inspector called “Jack”: Collect numbers until either the sum adds up to 21, or the list is empty.

- As in Part A, the classes should be thread safe.
- For Part B, you will need only four threads, one per inspector.
- Similarly as Part A, use a randomly generated LinkedList.
- Initialize the value of the variable holding the collected number to zero.
- After reading a number from list, remove it from the list, independently of whether it was used by the inspector or not.
- In case the task could not be accomplished, display a message that shows the current members in the list and state that the inspector “Failed”.
- Make sure that the program terminates.
- Provide comments in your code so that they faithfully represent the objectives of the program and of the inspectors.
- The computation should be repeated ten times for the same LinkedList, i.e., the same randomly generated list should be used ten times, each time printing the final result.
- Although it would be nice to have test programs for the four inspectors, too, but this is not a requirement for this assignment.

Submission: Use your Last Name (just one word; you can simplify your last name if you prefer) for the package name, with suffixes “A” and “B” for the two parts, respectively. Submit a .zip file containing your project files to Assignments in Blackboard as specified in the slides for Lecture 1. Remember to include Ant build scripts (build.xml).