JAV745 Fall 2019: Lab 5 - (4%)

Dr. Eden Burton
School of ICT, Seneca College of Applied Arts and Technology
Fall 2019

Due Friday November 22, 2019 - 11:30 pm

Instructions

Please read the instructions carefully and follow the naming conventions specified for each question. Solutions must be submitted in the Blackboard Dropbox created for Lab 5.

Note that the deadline is strictly enforced. The system tracks the exact time that submissions are uploaded. **There is a 10% per day penalty for late submissions.**

Additional Notes

- You may use any IDE for development but note that demonstrations and professor testing will be done exclusively on the command line. Ensure that you test your programs on the command line before submission.
- ensure that your programs are documented using *JavaDoc* standards

Question Descriptions

Question 1) In this question, you will implement and compare common algorithms used on data structures.

Use the given Reading class to create objects. *For those interested, this is an example of the Factory pattern*

Create a class called *Tester* which you will use to conduct your experiments. In the *main*() method, you should create a configurable number of objects. Allow the value to be passed in to the program as a parameter. As you create the objects, place them in an array (not a collection). You will invoke one of the sort routines (specified below) and print the sorted array into a text file. One row per object with a format as defined in the *Reader.toString*() method. The user should be able to select the sort method. Your timing results should be displayed on the console.

In another static method, implement the insertion sort as described in section 19.7 of the text. Order is based on the windspeed, then the temperature. Measure the elapsed time for the algorithm takes to execute using something like System.nanoTime(). Do the same for the merge sort (as in 19.8)

Run the application a few times and summarize your results. Explain your results in a separate written document (submit in pdf format).

Figure 1: High Level View of Data Flow for Producer/Consumer Application

Question 2) In this question, you will create a multithreaded application which has multiple threads accessing a shared resource.

Review section 23.5-23.6 of the text. You will implement a instance of this Producer/Consumer pattern where the Producer generates *Reader* objects subject to the following conditions

(a) the Buffer object should store 100 objects. For simpility, it uses the ArrayBlockingQueue object for storage. (https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ArrayBlockingQueue.html). Advanced students can try to use non-thread safe data structures and manage synchronization themselves.

(b) the Producer class should generate 2000 Reader objects, insert them into the Buffer object and then terminate

(c) the Consumer removes an Reader object from the Buffer, and writes its contents to a file. It is to write the contents of the Reader objects (as in the toString() method) into a file. Each Consumer has its own file it writes to. This process should terminate when it must wait for longer than 1 minute to remove an object.

(d) the Tester class should create 5 Producers threads and 3 Consumer threads.