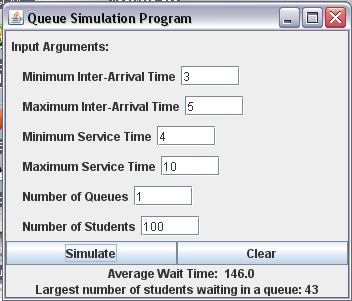
Assignment 4 – Computer Science Tutoring Lab Simulation

Maximum Points = 50

The purpose of this lab is to introduce you to the Queue\*\* data structure using Generics. In this lab you will write a computer simulation\* to represent the CCT450 Computer Lab Tutorial System. The lab manager must decide between having a few lab assistants and potentially incurring the wrath of the students if they have to wait too long, or spend more money to hire additional lab assistants.

Your program will allow the lab manager to run multiple simulations to determine “the optimal number of lab assistants to hire for a specific number of students”. The program will allow the lab manager to set the minimum and maximum amount of service time (time needed to tutor a student). The exact service time for each student will be generated randomly between these two numbers. Rather than keeping track of a clock, we will use the amount of time between student arrivals, by allowing the lab manager to enter the minimum and maximum amount of inter-arrival time (time between students arriving at the lab). The exact time between arrivals for each student will be generated randomly between these two numbers.

Your program will read in the data (minimum and maximum inter-arrival times, minimum and maximum service times, number of queues and students) from a GUI:



Once the data is entered and the lab manager presses simulate, your program will display the average wait time for the students and the largest number of students waiting in one queue during the entire simulation. Since the queue implementation is based on an unbounded array, you should make sure that students in your simulation always join the shortest queue.

I will provide you with the queue package based on Chapter 5 and Appendix F. You may use the classes discussed in the textbook on pages 330-346, with the following modifications:

* You must use my queue package which uses generics
* You must compute and display the largest number of students who were in a queue at the same time (i.e. the longest waiting line)
* Validate the input – minimum fields must be less than maximum fields, arrival times must be at least 0, other input values must be at least 1. If input fields are invalid, display an appropriate error message and allow the user to fix them.
* You may want to improve the appearance of the GUI.

BE SURE TO NOTE ANY CODE THAT IS NOT YOURS.

(Due before class on Thursday, March 4, 2010) Submit a .doc file containing a UML class diagram for all the classes used in your program.

(Due before class on Thursday, March 18, 2010) Submit your .java files containing your program.

 Grades are determined using the following scale:

* Runs correctly..…………………:\_\_\_/10
* Correct output……..……………:\_\_\_/10
* Design of output..………………:\_\_\_/8
* Design of logic…………………:\_\_\_/10
* Standards……………………….:\_\_\_/7
* Documentation.………………...:\_\_\_/5

[Grading Rubric](http://csc.colstate.edu/summers/NOTES/1301/Grading-Rubric-Programs.htm)  ([Word document](http://csc.colstate.edu/summers/NOTES/1301/Grading-Rubric-Programs.doc))

\* A **computer simulation**, a **computer model** or a **computational model** is a [computer program](http://en.wikipedia.org/wiki/Computer_program) … that attempts to [simulate](http://en.wikipedia.org/wiki/Simulation) an abstract [model](http://en.wikipedia.org/wiki/Model_%28abstract%29) of a particular system. [http://en.wikipedia.org/wiki/Computer\_model]

\*\* A **queue** (pronounced /kjuː/) is a particular kind of [collection](http://en.wikipedia.org/wiki/Collection_%28computing%29) in which the entities in the collection are kept in order and the principal (or only) operations on the collection are the addition of entities to the rear terminal position and removal of entities from the front terminal position. [http://en.wikipedia.org/wiki/Queue\_(data\_structure)]