COSC 311, Project #3

Due Date: October 29, 2013

Eastern Michigan University provides remote access to its computing facilities via a bank of modems.  A modem (from this modem bank) is accessed by dialing a telephone number which is made public to the university community.  Suppose that the university has installed a system that queues phone calls when all modems are busy.  Write a Java program to simulate the modem bank using the following simulation parameters: length of simulation, average time between dial-in attempts, number of modems in the bank, average connection time, and the size of the queue.

A simple airline simulation program which you might find helpful is discussed in chapter 4 of your textbook (pages 221-237). Read that section before you start implementing this program.

 In this project, you will have two different queues: a priority queue for the events (dial-in or hang-up), and a regular queue to queue the users who are waiting for modems to become free. The program must maintain a clock, and at each time unit the event queue is checked to determine if any events occur at that time.  There are two possible events: dial-in (users dialing), or hang-up (users leaving). All the events scheduled for the current time unit are removed from the event queue and processed. A dial-in event is placed in the waiting queue until a modem becomes available. A hang-up event signals that a modem becomes free. When a user is finally given a modem, the connection time is calculated and added to the dial-in time. This gives the hang-up time for that user. A hang-up event for that time will be added to the event queue. A key step in this simulation is to determine, at each unit of time, how many users dial in and the length of the connection for each. Empirical studies have shown that the arrival patterns of dial-ins as well as the connection times are random following a Poisson distribution.

PROGRAM INPUT: The simulation parameters are entered interactively and include:

-  the length of the simulation,

-  the average time between dial-in attempts (Poisson distribution),

-  the average connection time (Poisson distribution),

-  the number of modems in the bank, and

-  the size of the waiting queue.

The program must prompt the user for each input, and each input is entered on a separate line. At the end of each simulation, the program asks if another simulation is desired. If the answer is yes, the program prompts the user to input a new set of simulation parameters.

PROGRAM OUTPUT: The outputs are printed both on the screen and to a report file (report.txt).  The screen output for the current simulation is: the percentage of the time the modems were busy, and the average wait time.  Following the execution of the last simulation, the following message is printed to the screen:  “Simulation is complete: Summary in file report.sim”

The output in the file “report.txt” consists of the set of simulation parameters, the percentage of the time the modems were busy, the average wait time, and the number of users waiting in the waiting queue when the current simulation runs out.  The output for each simulation should be printed on a single line, resulting in a tabular listing. A header line should be written to the file at the beginning of the program to label the columns, and possibly a trailer section should be written to the end of the file, describing the meaning of the columns in the table.

PROCESSING REQUIREMENTS: Make allowance for an infinite queue.  If a dial-in comes when the queue is full, discard the request and print an error message on the screen. Real-number output should be specified to two decimal places.

SUBMISSION:  Please submit the following in a manila envelope:

            - A print out of the source code (well documented).

            - A print out of the report.txt file and a sample screen output

            - A print out of the readme file

            - A CD (or DVD) that contains your project folder (this will contain all the

               source files, project files, readme.txt, and report.txt).

GRADING:   Documentation 15%, Program style 5%, Program Correctness:  80%