CIS*2460F13 – Assignment 3

Due: Friday November 1 at 11:59pm

Purpose:

To familiarize yourself with the event driven simulator provided and generating statistical measures for such a system.

Questions:

1) The current single-server system provided with the Discrete Event Simulator (available on the Moodle course site) computes the basic statistics: N, NW, WS, WQ, T0 and Tmax. Using these values, create a function that produces the following derived statistics

   - average wait time: $w_q = W_Q / N$
   - probability of waiting: # customers who wait / N
   - probability of idle server: $P_0 = T_0 / T_{max}$
   - probability of busy server: 1 - P_0
   - average service time: $w_s = W_S / N$
   - average service rate per minute: 1 / w_s
   - average service rate per hour
     i.e. average number of customers served per hour

   and stores them in 'stats' and finally returns 'stats' to be stored in a 'results' variable.

   This function should be linked to the simulator through the 'measure' parameter of new.discreteEventSystem(). In other words you should modify the first function in singleServerSystem.R to read

```
new.singleServerSystem <- function(){
    new.discreteEventSystem(firstEvent.name = "A",
                            measure = myDerivedStatsFn,      <-- your function goes here
                            state = new.state(...),
                            stats = new.stats(...),
                            updateFn.stats.default = base.statsUpdate,
                            updateFns = new.updateFns(...)))
}
```

   Call the file with your modifications "sssWithDerivedStats.R".

   Run the system in verbose mode and your function as the system measure (instead of the default 'identity' function) for 100 simulated minutes to demonstrate that your function works.

   Then run the system (silently) for 500 and 1000 and 5000 simulated minutes and compare the results.

2) While T_0 is the total idle time (i.e the T0 variable in 'stats') , it is denoted as T_0 because it also is the total time that there are no customers in the single server system.

T_1 denotes the total amount of time where there is only a single customer in the system (no one in the queue, but the server is busy).

T_2 denotes the total amount of time where there is two customers in the system (one being served, one in the queue), etc.

Modify the single-server system in singleServerSystem.R so that you

- Keep track of T_0, T_1, T_2, … in the single server stats.
- Take the 'stats' returned by the simulator and report these times as well as the percentage of time the system had no customers, one customer, two customers etc. (i.e. T_0/T_{max}, T_1/T_{max}, T_2/T_{max}, …)

Call the file with your modifications "sssWithTL.R"

Run the system in verbose mode and your function as the system measure (instead of the default 'identity' function) for 100 simulated minutes to demonstrate that your function works.

Then run the system (silently) for 500, 1000 and 5000 simulated minutes and compare the results.

3) The file singleTiringServerSysterm.R contains a modified version of the single-server system. In this system the service time of the single server is affected by the how long the previous service event took. The server is more tired the longer the previous service time lasted; this is directly related to the length of the previous task and not to the length of time the server has been serving overall.

- Do the following:
  - Run singleTiringServerSysterm.R in verbose mode with an endTime of 150 minutes.
  - Take note of what happens to the service time (the ST variable) from event to event.
  - Repeat this 15 times.
  - Discuss what you observe. Include an explanation for what is happening.
    - Hint: the formula used to calculate service times was not well designed, leading to undesirable results.
- Next:
  - Modify how the service time is calculated in the update events to produce more realistic behavior in the system.
  - You must demonstrate that your system has the desired effect; don't just produce the code with no analysis.
  - Compare the behavior of your system with the original single-server system using the statistics you calculated in question 1

Call the file with your modifications "stssWithBetterST.R"

Important Note:

- If you have already loaded singleServerSysterm.R, you must reload the entire file singleTiringServerSysterm.R, not just the function new.singleTiringServerSystem(); otherwise the simulator will not work properly.
- Similarly, if you want to run the original single-single system, you must reload singleServerSysterm.R and not just run new.singleServerSystem()

Mark Breakdown

	Q1	Q2	Q3a	Total
Correct Answer	3	3	3	9
Code	3	3	3	9
Testing	3	3	3	9
Discussion	3	3	5	11
Total	12	12	14	38

Instructions:

Code

- All code must be written in R

Statistics

- You do **not** need to perform any statistical tests for either question.
  Just report and describe the behavior you see.

Documentation

- All questions must be answered in single document using output from your programs as evidence.
  - If only code is provided with no documentation actually answering the questions, you will get a grade of **zero** for the assignment.
  - Similarly if you only provide the answer document and don't hand in your source code and executable file, you will get a grade of **zero** for every question with missing programs.
- All documentation must be provided in MS-Word, PDF or PS formats
- Source code must be in text form (of course)

Uploading Files

- When you upload your assignment make sure all files are bundled together into a single file
  - This includes documentation and programs
  - You may use tar or zip formats
- Upload the tarred or zipped files to the assignment page on the course website