COSC 311 Programming Project #1 Simulation using a queue
pp1008
Distributed: 10/8/2019 Due: 10/29/2019

For a specified population of customers requiring service, how many servers are necessary, i.e., what is the minimum number of servers required, to keep the customers' wait time <= 5 minutes each.

A donut shop wants to know how many servers to hire. You must design and run a simulation experiment to empirically arrive at a reccomendation.

You model the donut shop as a single queue / multiple server system. Any idle server will remove the first element from the queue.

Each customer knows how long his service will take. This is a random value that your simulation will generate.

At each time tick, a random number of customers enter the donut shop and enter the queue.

At each time tick:
Each server that has completed service of a customer "releases" the customer.
Every customer currently in line has their wait time incremented.
A number of new customers enter the system.
Each idle server removes one customer from the queue.

Constraints:
- A customer's wait time starts at 0 and is incremented for each minute he is in queue.
- A customer's wait time stops being incremented after he is assigned to a server.
- The customer's service time is generated for each new customer. Use a uniformly distributed random number.
- The number of customers that arrive at each minute is a Poisson distributed random number.
- The extent of time of interest is 20 minutes.

Customers arrive according to a Poisson distribution. At each tick of time, the Poisson generator will compute the number of arrivals. The mean of a Poisson distribution is the average number of customers arriving per tick. So, a mean of 0.2 indicates 0.2 customers per tick, or 1 customer every 5 ticks; a mean of 3.0 indicates an average of 3 customers per tick.

https://stackoverflow.com/questions/9832919/generate-poisson-arrival-in-java gives David Knuth's algorithm as:

```
private static int getPoissonRandom(double mean) {
```

```
    Random r = new Random();
    double L = Math.exp(-mean);
    int k = 0;
    double p = 1.0;
    do {
        p = p * r.nextDouble();
        k++;
    } while (p > L);
    return k - 1;
}
```

Experiment:

Run the experiment under the following environmental conditions with 1, 2, 4, 8 servers. I.e. you will have a total of 8 experiments.

Service requirement	Customer arrivals	Comment
1 – 12 minutes	2 / minute	heavy demand
1 – 3 minutes	0.25 / minute	low demand

Provide output to confirm your code is correct in the case where service requirement is 1 – 3 minutes, with 2 arrivals/minute, and 4 servers.
DO NOT PROVIDE OUTPUT FOR OTHER CONDITIONS!

You will use ONE random number generator, seeded to 97.

The output will be in this format. At the end of each tick, output these values:
Tick #: 0

Customers in service:
Customers with completed service:
Customers in queue:
Total wait time:
Wait time: minimum, average, maximum

Tick #: 1

Customers in service:
Customers with completed service:
Customers in queue:
Total wait time:
Wait time: minimum, average, maximum

. . .

Tick #: 19

Customers in service:
Customers with completed service:
Customers in queue:
Total wait time:
Wait time: minimum, average, maximum

Turn in:

Hardcopy of code.
Screen shot of execution on required output
UML diagram of classes

Write up:

A paragraph describing your results and recommendation to your client.
A table that clearly presents the result of your 8 experiments.

Further requirement:

Good style is required. No method may exceed 30 LOC. Names must be reasonable.

Reasonable and appropriate comments are required.

All source code must include header with the standard identifying information: Name, assignment id, COSC 311, FALL 2019, …

You must email me your URL and instructions on how to compile/run.