EEN 218
Lab #5
Simulation of a Supermarket Checkout Line

In this assignment, you are going to simulate the checkout line at a supermarket using Queues. In this supermarket, there is only one line, represented by a Queue, and one Cashier at a time. Note that the Cashier is not part of the Queue and the person at the head of the Queue leaves the Queue when he/she goes to the Cahier. You are trying to determine the average wait time for customers of the supermarket.

You need to provide a report that shows (20):

1. Your class designs
2. The algorithms used
    a. This should be in pseudo-code or flowcharts, not just the code you ended up writing
3. Compiling instructions
4. Sample runs (screenshots are fine)

Things you must have (20):

1. (5) A Customer class (customerID, arrivalTime, dequeueTime)
2. (5) A Cashier class (currentCustomer, serviceTime, shiftTime, busy/not busy)
3. (10) A generic Queue class (using templates)
    a. Enqueue, Dequeue methods
    b. A count of number of items in the queue as well as the maximum number of items ever in the queue
4. A random number generator (check code on blackboard)
5. Ability to easily change the range of arrival times and service times

Things that may be useful:

1. A Time tracking class
2. An Event class (eventTime, eventType) to track events
3. An Event priority queue, sorted by eventTime and eventType
4. Ability to write output to a file

Initial Test: Typical Queue (30)

Has a Queue of Customers and a single Cashier who never changes.

Customers arrive at a rate of λ and are served at a rate of μ.

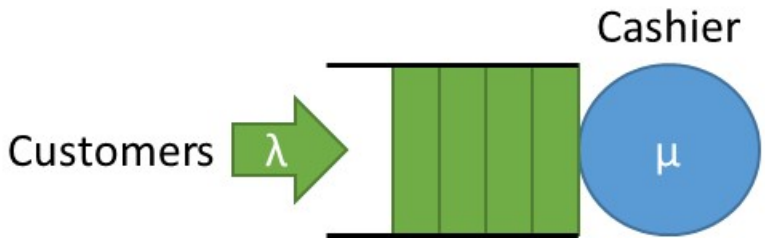

Figure 1 Normal M/M/1 Queue Organization

Algorithm

Customers arrive at random integer intervals of 1-4 minutes. Each Customer also takes random integer intervals of 1-4 minutes to be served. You are going to simulate the operation of the checkout line over the course of 12-hours (720 minutes) using the following algorithm:

1. Choose a random integer from 1 to 4 to determine the minute the first customer arrives
2. At the first customer's arrival time
    a. Determine the customer's service time (random integer from 1 to 4)
    b. Begin servicing the customer
    c. Schedule the arrival of the next customer (random integer from 1 to 4 added to the current time)
3. For each minute of the day:
    a. If a customer arrives:
        i. Print out that a new customer has arrived
        ii. Enqueue the new customer
        iii. Schedule the arrival of the next customer
    b. If service is completed for a customer
        i. Print out that the customer is finished
        ii. Dequeue the next customer to be serviced
            1. Determine this customers service completion time and schedule it
        iii. Remember to handle the case if there are no customers in line
4. When the simulation is done, print out:
    a. Total number of customers served
    b. Maximum number of customers in the checkout line at any time
    c. Longest wait time for any customer
    d. Average wait time for all customers

Modified Dual Queue (30)

For this simulation, we are also going to allow the Cahier to be replaced at given intervals (i.e. they have a fixed work time). The Cashiers are also drawn from a fixed queue of Cashiers set so that they cover the entire 12-hour cycle. You will randomly choose a number of Cashiers and set their shift time so that the total covers the 12-hour cycle. When a Cashiers shift time is over, you will switch them out for the next Cashier in the Cashier Queue.

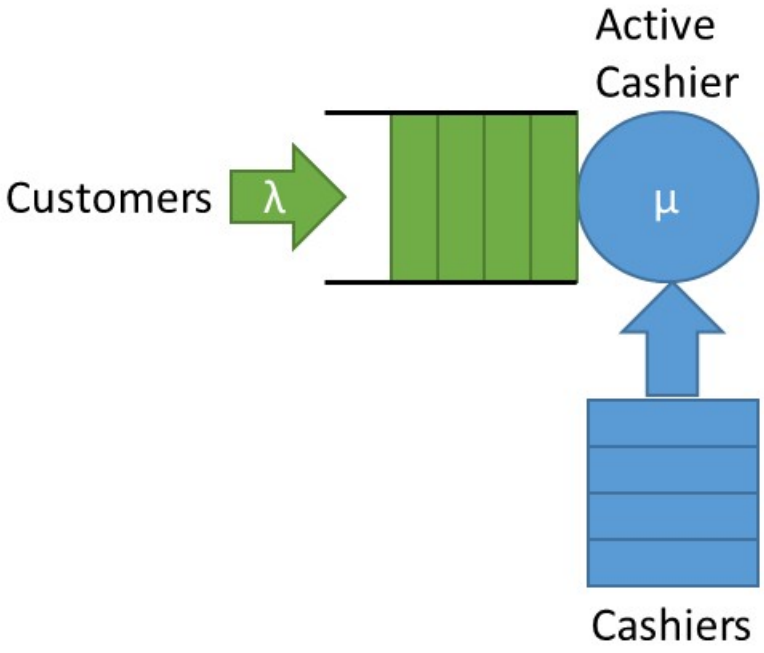

Figure 2 Dual Queue Operation. Note only one Cashier is active at a time

You will need to modify the initial algorithm to account for the changing of the Cashiers.

Sample Output:

```
Started
Time:2:Customer Arrival:Customer[id:1, enqueue:2, dequeue:0, service:3]
Time:2:Customer Dequeue:Customer[id:1, enqueue:2, dequeue:2, service:3]
Time:2:Service Start:Customer[id:1, enqueue:2, dequeue:2, service:3]
Time:3:Customer Arrival:Customer[id:2, enqueue:3, dequeue:0, service:4]
Time:4:Customer Arrival:Customer[id:3, enqueue:4, dequeue:0, service:2]
Time:5:Service End:Customer[id:1, enqueue:2, dequeue:2, service:3]
Time:5:Customer Dequeue:Customer[id:2, enqueue:3, dequeue:5, service:4]
Time:5:Service Start:Customer[id:2, enqueue:3, dequeue:5, service:4]
Time:6:Customer Arrival:Customer[id:4, enqueue:6, dequeue:0, service:1]
Time:8:Customer Arrival:Customer[id:5, enqueue:8, dequeue:0, service:2]
Time:9:Service End:Customer[id:2, enqueue:3, dequeue:5, service:4]
Time:9:Customer Dequeue:Customer[id:3, enqueue:4, dequeue:9, service:2]
Time:9:Service Start:Customer[id:3, enqueue:4, dequeue:9, service:2]
Time:10:Customer Arrival:Customer[id:6, enqueue:10, dequeue:0, service:4]
Time:11:Service End:Customer[id:3, enqueue:4, dequeue:9, service:2]
Time:11:Customer Dequeue:Customer[id:4, enqueue:6, dequeue:11, service:1]
Time:11:Service Start:Customer[id:4, enqueue:6, dequeue:11, service:1]
Time:12:Service End:Customer[id:4, enqueue:6, dequeue:11, service:1]
Time:12:Customer Dequeue:Customer[id:5, enqueue:8, dequeue:12, service:2]
Time:12:Service Start:Customer[id:5, enqueue:8, dequeue:12, service:2]
.
.
.
Time:718:Customer Dequeue:Customer[id:278, enqueue:682, dequeue:718, service:3]
Time:718:Service Start:Customer[id:278, enqueue:682, dequeue:718, service:3]
Time:720:Customer Arrival:Customer[id:292, enqueue:720, dequeue:0, service:1]
Time:721:Service End:Customer[id:278, enqueue:682, dequeue:718, service:3]

++++++++++++++++++++++++++++++++++++++++++++++
Customers served:279
Total Wait Time:8228
Total Serv Time:719
Average Wait Time:29.491
Average Serv Time:2.57706
Max Wait Time: 43
Current Customer Queue Length:14
Customer Max Queue Length:20
```

Extra Credit (10):

Consider the either:

1. An Event queue that tracks each event that will happen in the future. This is a priority queue sorted by the eventTime and then by the eventType
2. A single customer queue, but multiple active cashiers.