Vandana Anand, Fareya Ikram

*****SUMMARY*****

The program simulates customers randomly arriving at a bank at different times. There are two types of queues; one is where all the customers stand in one queue and wait for the teller to call them and another is where there is a separate line for each teller and a customer picks the shortest line. The purpose of this program is to compare the efficiency of a single line versus a separate line for tellers. The output show the statistics how many customers were in the bank, the total time to serve the customers, the number of tellers, the type of queuing, the mean and standard deviation of the amount of time a customer spent in the bank, the maximum wait time for the customer, the total amount of teller service time and idle time.

*****HOW TO RUN THE PROGRAM*****

In order to run the program, you have to compile it first. To compile it, you have to type in make in the terminal. Another way to compile it is to type in g++ -Wall qSim.cpp in the terminal. Then, to actually run the program, you have to type in the name of the file, which should be qSim, the number of customers, the number of tellers, the simulation time(in minutes), the average service time(in minutes), and a seed to indicate where to start the random number generator, which is optional after typing make. An example of what to type is the following:
./qSim 156 6 45.2 3.5
After calling g++ -Wall qSim.cpp, you would type in ./a.out (then the commands indicated above) to run the program.

*****PROBLEMS*****

Some problems we encountered were understanding the overall program and how to use the abstract classes. We went to Professor Lauer and got these things clarified. We did not have to use abstract classes as we didn't learn about it. Professor Lauer also clarified what we had to do overall to start the project. After that, we started understanding it.

*****ALGORITHM*****

We looked at a website that had an ice cream simulation. We looked at how they organized their code and organized ours in a similar way. The program is organized in the following way:

a) qSim.cpp/qSim.h

There is a random generator to generate random numbers. The main function processes the user's command line prompts(which are outlined in the how to run it section). There are a few for loops that run through the tellers and customers and have them in the simulation. Delete function is called to erase it. Help function is available to guide the user about what to enter. We also have a **IDLE_MAX** constant variable at the top with all the other header files and include statements that sets the maximum teller idle time to 600.

b) Event Class

The event class basically processes the whole event of a customer coming into the bank and leaving. The customer arrives at the bank at a random time, the customer may have to wait if the teller is busy, and the customer leaves at some time.

Some things to note about an event class:

Int time takes care of the time of event

startTime is when the event starts

ID is the customer identity

TID is the teller of the customer ID

Boolean cort determines customer or teller

Int eventType , 1 means arriving customer, and free teller

Lastly this works with processEvent (THIS IS THE EQUIVALENT OF **action method**), this is also the core of everything

c) Linkedlist Class (THIS IS OUR IMPLEMENTATION EVENT QUEUE)

This class processes all the queues. The insert node function adds in all of the customers to the queue when they arrive at the bank, the remove node removes the customers when they leave the bank, there is a print list function that prints out the details, and then delete is called to clear the linked list. When an event is "finished" the head of the list is deleted inside the simulation class, using a linked list function.

d) Simulation Class

The simulation class runs both the multiple queue simulation and the single queue simulation. It makes sure to implement the linked list. There is a boolean value that determines which simulation to run. True for the first, and false for the second. Linkedlist is deleted using linked list functions.

e) Statistics Class

The stats class calculates the outputs that are required to be displayed, such as the mean, and the standard deviation for which we have a for loop to go through the customers and find their time, then averages and finds the standard deviation. The class also calculates the max wait time for the customer. It then prints the stats that was found.

*****ANALYSIS OF RESULTS*****

Increasing tellers will decrease customer waiting time which our data supports. For example, the customer waiting time is 2 minutes in a single queue in one of our test cases, but it is only 1 minute in multiple queues. However, increasing customers will make the wait time higher. The teller service time is also much higher for multiple queues than for single queues. Multiple queues is probably more appropriate because it looks like a more uniform distribution and tellers would be more efficient knowing there are less people in line. With a single queue, it would be more tense for tellers. When there are more customers in the bank, this approach would help. When there are less customers in the bank, a single queue might be better so that a teller can just help them when they are not busy instead of having them wait in line for a teller who won't be appearing for a while.

*****OUTPUT FOR TEST CASES*****

```
vanand@CS-2303-VirtualBox: ~/PA4_fsikram_vanand
vanand@CS-2303-VirtualBox:~$ cd PA4_fsikram_vanand
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ make
g++ -g -Wall -lm -o qSim qSim.cpp qSim.h
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ ./qSim 20 25 60 15
End of Single Queue Simulation
Customers Average waiting time 8
Customers Max waiting time 14
Customers SD time 4
Tellers service time 162
Tellers idle  time 318
End of Multiple queue Simulation
Customers Average waiting time 8
Customers Max waiting time 1
Customers SD time 4
Tellers service time 322
Tellers idle  time 318
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ g++ -Wall qSim.cpp
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ ./a.out 65 65 1000 15
End of Single Queue Simulation
Customers Average waiting time 6
Customers Max waiting time 3
Customers SD time 4
Tellers service time 450
Tellers idle  time 14504
End of Multiple queue Simulation
Customers Average waiting time 7
Customers Max waiting time 11
Customers SD time 3
Tellers service time 944
Tellers idle  time 14504
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ make
make: 'qSim' is up to date.
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ ./a.out 555 400 60 15 3
End of Single Queue Simulation
Customers Average waiting time 7
Customers Max waiting time 11
Customers SD time 4
Tellers service time 4115
Tellers idle  time 6082
End of Multiple queue Simulation
Customers Average waiting time 7
Customers Max waiting time 11
```

```
vanand@CS-2303-VirtualBox: ~/PA4_fsikram_vanand
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ ./qSim 20 25 60 15
End of Single Queue Simulation
Customers Average waiting time 8
Customers Max waiting time 14
Customers SD time 4
Tellers service time 162
Tellers idle  time 318
End of Multiple queue Simulation
Customers Average waiting time 8
Customers Max waiting time 1
Customers SD time 4
Tellers service time 322
Tellers idle  time 318
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ g++ -Wall qSim.cpp
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ ./a.out 65 65 1000 15
End of Single Queue Simulation
Customers Average waiting time 6
Customers Max waiting time 3
Customers SD time 4
Tellers service time 450
Tellers idle  time 14504
End of Multiple queue Simulation
Customers Average waiting time 7
Customers Max waiting time 11
Customers SD time 3
Tellers service time 944
Tellers idle  time 14504
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ make
make: 'qSim' is up to date.
vanand@CS-2303-VirtualBox:~/PA4_fsikram_vanand$ ./a.out 555 400 60 15 3
End of Single Queue Simulation
Customers Average waiting time 7
Customers Max waiting time 11
Customers SD time 4
Tellers service time 4115
Tellers idle  time 6082
End of Multiple queue Simulation
Customers Average waiting time 7
Customers Max waiting time 11
Customers SD time 4
Tellers service time 8321
Tellers idle  time 6082
```