Kevin Chen
nkc160130
Due: 26 October 2019
SE 4348
Prof. Ozbirn

Project 2

The gist of this project is to utilize semaphores to enforce mutual exclusion and coordination of the activities of a bank teller, loan officer and customer, and threads for running all the customers, loan officer, and bank tellers concurrently. The basic flow of the program is as follows: First, the main class initializes all the threads for the customers, loan officers, and bank tellers. Then, all the threads are started. The customer first randomly chooses whether they'll deposit, withdraw or loan. Then, a random amount is chosen as to how much to deposit, withdraw or loan. Then, depending on whether they want to withdraw, loan or deposit, the request is directed to the bank teller or loan officer. Either way, it waits for the teller or loan officer to be ready, then they are added to a queue of customers in line, notifying the teller or loan officer that they are ready, and waits for them to signal that they have finished. The teller or loan officer takes a look at the customer's request, processes it by updating the customer's balance (whose access is controlled by a mutex), and then signals to the customer that they are finished. Each customer repeats this process 3 times, randomly choosing between the options, and then the customer thread is joined back and, once all the customers are done, the program exits. Each of the loan officer and teller's actions have a delay, also; for example, processing a deposit takes 4 minutes, or 400 milliseconds in actual time.

I expected that dealing with semaphores and threads would be the biggest challenge for this project. Once I created the pseudocode for the project, coding it and figuring out what semaphores to use and where to use them was relatively trivial. Threads were also simple to use and coordinate, with little difficulties on that end. The biggest snag that I hit on this project was trying to figure out why a customer was always waiting on the teller if it went to teller 0 and

Kevin Chen
nkc160130
Due: 26 October 2019
SE 4348
Prof. Ozbirn
teller 1, or if it visited the loan officer and wanted to visit the loan officer again. I eventually realized that I wasn't putting most of the code in an infinite loop, so whenever the semaphore was triggered and the loan officer and bank teller were done executing it would exit, leaving the customers waiting forever in line.

The biggest thing I learned in this project was creating and dealing with Semaphores and Threads in Java. I have never done concurrent programming in Java and doing this project has not only reinforced my knowledge of semaphores and threads, but also how to implement and utilize them in Java.

I successfully completed the project, with all functionality working. The output I got at the end were as expected; the threads print when they were created, when they have actions, and when the program is finished running. Here is some sample output from the program:

```
Teller 0 created
Customer 0 created
Customer 1 created
Customer 4 created
Customer 2 created
Teller 1 created
Loan Officer 0 created
Customer 3 created
Customer 1 gets loan from loan officer
Teller 0 begins serving customer 0
Teller 1 begins serving customer 4
Customer 0 requests of teller 0 to make a deposit of $100
Customer 4 requests of teller 1 to make a deposit of $100
Loan Officer approves loan for customer 1
Customer 0 gets receipt from teller 0
Teller 0 begins serving customer 3
Customer 4 gets receipt from teller 1
Teller 1 begins serving customer 2
Customer 3 requests of teller 0 to make a withdrawal of $400
Customer 2 requests of teller 1 to make a deposit of $200
Customer 3 gets cash and receipt from teller 0
Teller 0 begins serving customer 1
Customer 2 gets receipt from teller 1
Teller 1 begins serving customer 0
```

Kevin Chen
nkc160130
Due: 26 October 2019
SE 4348
Prof. Ozbirn

```
Customer 3 gets loan from loan officer
Customer 1 requests of teller 0 to make a withdrawal of $100
Customer 0 requests of teller 1 to make a deposit of $200
Customer 1 gets cash and receipt from teller 0
Teller 0 begins serving customer 4
Customer 0 gets receipt from teller 1
Teller 1 begins serving customer 2
Loan Officer approves loan for customer 3
Customer 3 gets loan from loan officer
Customer 4 requests of teller 0 to make a deposit of $200
Customer 2 requests of teller 1 to make a deposit of $400
Customer 4 gets receipt from teller 0
Teller 0 begins serving customer 1
Customer 2 gets receipt from teller 1
Teller 1 begins serving customer 0
Loan Officer approves loan for customer 3
Customer 3 departs the bank
Customer 4 gets loan from loan officer
Customer 1 requests of teller 0 to make a withdrawal of $200
Customer 0 requests of teller 1 to make a withdrawal of $200
Customer 1 gets cash and receipt from teller 0
Teller 0 begins serving customer 2
Customer 1 departs the bank
Customer 0 gets cash and receipt from teller 1
Customer 0 departs the bank
Customer 0 is joined by main
Customer 1 is joined by main
Loan Officer approves loan for customer 4
Customer 4 departs the bank
Customer 2 requests of teller 0 to make a deposit of $200
Customer 2 gets receipt from teller 0
Customer 2 departs the bank
Customer 2 is joined by main
Customer 3 is joined by main
Customer 4 is joined by main


Bank Simulation Summary
+------------+--------------------+---------------+
| Customer # |   Ending balance   |  Loan Amount  |
-------------+--------------------+---------------+
| 0          | $1100              | $0            |
| 1          | $700               | $300          |
| 2          | $1800              | $0            |
| 3          | $600               | $300          |
| 4          | $1300              | $400          |
+------------+--------------------+---------------+
| Totals     | $5500              | $1000         |
+------------+--------------------+---------------+
```