Ian Brown
SE 4348.003

Project 2 Summary

This project focused on the task of simulating the operations of a post office. Specifically, this simulation focused on the relationship of customers, who are all trying to access the post office at the same time and complete their assigned task, and the postal workers who are in charge of helping the customers to complete these tasks. In order to achieve synchronization between these two types of objects, threads were needed to simulate the customers and postal workers and semaphores were needed to keep customers and workers from accessing certain resources/critical sections before it was their turn.

Difficulties Encountered

This project presented quite a few difficulties for me. The greatest difficulty that I experienced resulted from my lack of experience dealing with threads and semaphores. When I began this project, the only experience that I had with these two topics ranged from learning about threads for less than a week in CS 2336 and the theoretical application of these two topics from Operating Systems Concepts class. Because of this, I found it challenging, although fairly enjoyable, to take this mostly theoretical knowledge and try to apply it to actually programming concepts. One of the biggest obstacles I ran into that related to these concepts was trying to coordinate semaphores using the PostOffice class as the “main” class, where all of the semaphore handling occurred. After trying for a few hours to find out why the program would pause after handling one complete action of customer and postal worker interaction, I decided that I needed to restructure my code to allow the customers and postal workers to handle the semaphore-related tasks in order to clarify my code to the point where I could see where I would acquire or release a permit and never have a corresponding action for it.

Another difficulty that I encountered during this project was trying to think through the general design of the program. This resulted from trying to start the project without having any type of pseudocode to go off of due to me originally believing that I didn't need pseudocode to efficiently code the project. After a little while of not being able to remember all of the steps I initially thought were necessary for the program, I decided that I needed to create pseudocode before continuing with my programming, which made it significantly easier later on to develop my code.

The last major difficulty I faced with this project was finding out how to make the postal worker threads end when there were no more customers to be served. I ran into this problem early on in my project when I was trying to get individual methods to work correctly and I thought that it would resolve itself as I got closer to having the code complete. It, however, continued to plague me after I had completed my code. After trying things like having volatile variables or hoping that joining the postal worker threads would stop the running threads, I eventually had to ask Professor Ozbirn, who in a few short sentences showed me the setDaemon method that makes a thread where it will not prevent exiting from the main program after the customer threads had finished.

Knowledge Gained

While I had my code structured for PostOffice to do the majority of the work, it was quite frustrating to not be able to get the code to work the way I wanted. However, it was greatly rewarding to finally get it working. I also believe that this obstacle helped me to be able to better understand how the threads were interacting with the semaphores and how to correctly facilitate it through code. I feel a lot more confident with these concepts now as a result of this project. Also, through my stubborn insistence that I didn't need to write pseudocode before my actual code and then being proven wrong, I realized that it is almost always easier to spend the few hours up front thinking through the program design than to spend many more hours later on trying to figure out what is causing a program to not work correctly.

Results

I ended up having the vast majority of my program working a day ahead of when the project was due, with the exception of joining customer threads and actually having the program exit properly. After class today, I had my program working properly, which was nice to not have to rush to finish the project. Overall, I would say that this project, while it had many moments where it could be frustrating, was highly rewarding and a good way to learn how to implement threads that are synchronized using semaphores.