Kunlun Song
A01
V00927475

1. I will use 1 main thread that has all the customers which will be the length given by the first value of the text file. Then I will have a thread for every clerk which is 5.
2. My threads will work independently.
3. I am going to use 2 mutexes one for business and economy, then I will use one mutex for every clerk which is 5 and lastly I will use one mutex to control the clerks not getting all the customers in one moment.
4. The main thread will read the file input, initialize the mutex and conditional variables, create the clerk and customer threads, wait for the clerk and customer threads, destroy the mutex and conditional variables. It will also calculate the average waiting time for everyone and the two classes (economy, business).
5. Customers will be a linked list queue having the customer id, class, arrival time and service time.
   ```
   struct customer_info{
      int user_id;
      int class_type;
      int service_time;
      int arrival_time;
     struct customer_info *next;
   };
   ```
6. To stop my data structures don't get modified concurrently I will be using mutexes and conditional variables to make sure nothing is being accessed at the same time by locking and granting access.
7. I am going to have 8 conditional variables. These are all going to be associated with their very own mutex too. The conditional variables will represent the 5 clerks, 2 class queues and lastly one to signal the clerks when the customer is ready to be served and to make sure that only one customer is being served at a time. The clerks need one to be signaled that they are done/ready and same with the queues. As I said, all of these conditional variables will have their own mutex that is named similarly to itself. When using the pthread_cond_wait() for the check if ready conditional variable I will wait to see if the customer is ready and then call pthread_cond_signal() or pthread_cond_broadcast(). I will also use these signals to alert the queues that the clerks are done.
8. First I will have the main thread which will read the file input, initialize the mutex and conditional variables, create the clerk and customer threads, wait for the clerk and customer threads, destroy the mutex and conditional variables. It will also calculate the average waiting time for everyone and the two classes (economy, business). Then I will have the customer thread which will first sleep for the customer's arrival time, separate it into business and economy class and then signal the clerk that it is ready to be served. From there it will check which clerk is ready to serve and sleep for the service time before telling the clerk that it is done by signaling it so that it can serve another customer. The clerk thread will choose which queue to take a customer from. It will prioritize the

business queue and signal the customer at the front of the queue it chooses. Then it will wait for the customer to finish being served finally when it is done it will increase the counter to know how many people have been served and when to end.