# Documentation

## Details of Implementation:

I implemented 3rd project of the CmpE322 as a single C++ source file. The main goal of my program is the preserve synchronization of the data. One customer is allowed to send data to specific ticket vending machine at a certain time. If there are other customers want to send data to that machine, they must wait other customer. Also, one ticket vending machine is allowed to send the amount of money to specific company at a certain time. If there are other ticket vending machines want to send amount of money to that company, they must wait other ticket vending machine.

This waiting and synchronization operation is done by mutex locks. I created 10 mutex locks for ticket vending machines. One mutex lock for each ticket vending machine. Also, I created 5 mutex locks for companies. One mutex lock for each company. Then I created one mutex lock for writing to the output file and one for incrementing counter of customer. When I send data from customer to ticket vending machine, I lock the mutex lock for that ticket vending machine, send data and unlock the mutex lock. When I send amount of money from ticket vending machine to a company, I lock the mutex lock for that company, send the money, and unlock the mutex lock. If I write to the output file, I lock write_lock. If I increment customer_counter, I lock counter_lock.

I send the data from customers to ticket vending machines through queues. I created one queue for each ticket vending machine. FIFO part is done by mutex locks already. So I just send data to the queue and get data from that queue easily.

I kept a counter named customer_counter. I initialized it 0 and then for every customer that is completed, I incremented it one. When customer_counter equals to the total number of customers, while loop ends, and threads exit.

I created structs for customer data and ticket vending machine data.