|  |
| --- |
|  |
|  |
| **Group 3** |

Assignment 3

Our design included creating a pthread for every seller and customer. Sellers got their own queue; we made separated int arrays to contain identifiers for customers in each queue. In our main() function, we first initialize all the threads and allocate the space for each seller’s queue. In addition, we use the setitimer() method to set the timer duration for sixty real seconds while having the clock make intervals for one minute, creating a virtual hour in duration. After this time has been passed, we use the signal method call, which changes the ‘timesUp’ variable to 1 to signals to all threads that no more tickets may be sold. Any transactions that are pending will be completed if space is available in the concert. Finally, any customers still left in queues that did not meet with a seller are removed and statistics for the run are displayed.

The shared data in this program is the ‘seats’ array that contains seat assignments for each successful transaction. The critical regions are, therefore, all the lines of code that assign a person to an index in the ‘seats’ array. This occurs in the sellerMeetsCustomer() function (once an available seat has been determined). This is also true in the print() function that loops through all the rows in the concert. Another shared piece of data is the seller queues. If a customer has arrived and is being placed into a seller’s queue, no other customer thread must be accessing the queue at the same time to make the assignment. The same is also true for removing customers from a seller queue and beginning the transaction process. We used the lock and unlock pthread mutex methods to ensure no other thread is using these resources in each case above.

Process synchronization is needed when one thread must wait for another thread to complete its execution before continuing. This is used in several parts of our program. Firstly, a customer must finish arriving and being placed in the seller queue before a seller can meet with them. Secondly, if one seller thread has made a successful sale, it must finish printing out the event and the 10x10 matrix before another seller can do the same. Lastly, the program must wait on all the seller threads to complete their sales, even past the 60-minute mark, to finish execution and to print out the end results.