Group 3

March 1, 2016

Homework3 Report

We did the extra credit.

Our overall design was based on OOD to simulate the Customers and various TicketSellers. The most shared data was the seats array, used by each of the ten TicketSeller threads. This had to be locked each time a TicketSeller had a customer was attempting to assign them as seat. We also locked the queue for each TicketSeller, as this was shared by both the TicketSeller itself, as well as with all its potential Customers. Each Customer was predetermined as to which queue it would join, then randomly assigned a time to join the queue. A thread was then split with a pointer to the Customer that would sleep until the assigned arrival time, at which point this thread would attempt to lock and then join the queue. This would conflict with the TicketSeller, who is trying to grab the first Customer (who may be the same object). As we did the extra credit, the Customer would also spawn a secondary thread, which would try to remove itself from the queue if the ticket seller had not served it within 10 minutes (seconds). The last thing we had to lock was the output, cout. Without locking it the different threads would take their turn writing characters into the output, making a huge mess. By creating a global mutex for this and passing it as a static variable to our classes, each was able to make use of the lock.