Justin Cole - W1286374
Tejas Dedhiya - W1605246
Jianqiao Ge - W1609203
Piyush Kulkarni - W1629006
Shivani Deosatwar - W1588465

Advanced Operating System: COEN 383
Project 3: Multi-Threaded Ticket Sellers
Group No. 2

Objective : This project gives us experience with a multithreaded program using Pthreads library.

We have implemented a C program that will run the Pthread library to create threads and mutex. This will simulate ticket sellers simultaneously selling concert tickets in one hour.

Summary : At the start of the program, 10 customer queues are created containing N customers. These queues store the randomly generated customer number, arrival time and service time. All arrival and service times are calculated in multiple of minutes and all customers arrive at the beginning of a minute. Every seller is then given a customer queue and begins to process each customer once they would have arrived. In addition, each of the 10 sellers are assigned a type: L, M, or H. Depending on their type they will serve their customers either faster or slower and will look for different seats in different fashions. Once a seller begins to serve a customer it will look for an available seat based on the seller type. Open seats are denoted by an unlocked mutex. This ensures that only one seller can claim any given seat. Once a seller has successfully claimed a seat mutex they will note down some details at that seat location including the response time, turn around time, and type of seller that sold this seat. All the customers will be served in a similar manner until all seats are filled or an hour has passed. Any of the remaining customers that couldn't be served will be turned away. At the end of the simulation time, all of the sold seats are gone through and the average response time, turn around time, and throughput for each type of seller is calculated.
The total number of customers per seller, the global clock counter, pthread arguments, and seat specific structure variables are shared among all the functions.

We have utilized the code given in the preview and constructed our task over that by simulating clock tick utilizing the main thread and controlling critical region and sell process in the child threads devoted for simulating ticket sales. In our undertaking, we have utilized the main thread to produce clock ticks for consistently simulating time duration of one minute.

We had following presumption for the simulation:
1. Clock Tick: Lowest measurable time quanta is one minute and every child thread has been intended to simulate work that takes one minute. For example, serving clients, trusting that they will finish or finishing the sell.

2. State of seller's thread: Each seller's thread is assumed to be in the following state in any single time quanta.

- Waiting: Waiting for new client
- Serving: Serving new client from the vender's line
- Processing: Processing and setting aside effort to process the sell
- Completing: Completing sell for the client

3. New clock is generated when all the seller's threads have finished their work for that time quanta so as to keep up time synchronization.

4. Concert seat has been simulated with a 2-dimensional matrix with the assumption that just one thread will be manipulating the matrix to keep away from any contention of seat assignment.

The whole workflow for the project is as per the following:

- **Initialization**: We have initialized the initial parameters such as lock mutex, show seat matrix, client queue for every seller and threads.
    - After creating seller threads, all threads were put into waiting for clock tick mode in the wake of wrapping up their initialization.
- **Simulating Clock Tick**:
    - When main thread sends clock tick, all the seller's thread start racing to acquire lock on concert seat matrix relying upon their state.
    - Initially, a seller thread checks for new customers based on arrival time and then begins serving each client in turn.
- **Termination Condition:** Each seller thread will terminate processing if the concert is sold out or reenactment time ran out. In both conditions, the client needs to leave.

Simulation Results:

N = 5

```
Final Concert Seat Chart
========================
 H101    H102    H103    H104    H105       -       -       -       -       -
    -       -       -       -       -       -       -       -       -       -
    -       -       -       -       -       -       -       -       -       -
    -       -       -       -       -       -       -       -       -       -
    -       -       -       -       -       -       -       -       -       -
 M101    M201    M102    M202    M203    M204    M301    M103    M205    M302
 M303    M304    M305    M104    M105       -       -       -       -       -
    -       -    L205    L304    L604    L204    L303    L603    L105    L302
 L405    L104    L505    L103    L203    L404    L102    L602    L504    L403
 L202    L402    L503    L301    L601    L502    L101    L501    L401    L201


Stat for N = 05
 ================================================================================
|    | No of Customers | Got Seat | Returned | Response | Turn Around |
 ================================================================================
|  H |               5 |        5 |        0 | 0.000000 |    1.600000 |
|  M |              15 |       15 |        0 | 0.800000 |    3.600000 |
|  L |              30 |       28 |        2 | 1.464286 |    6.928571 |
 ================================================================================
```

N = 10

```
Final Concert Seat Chart
========================
 H101    H102    H103    H104    H105    H106    H107    H108    H109    H110
 M310       -       -       -       -       -    L510    L210    L609    L309
 M209    M210    L408    L209    L109    L308    L509    L608    L407    L208
 M208    M308    M309    L508    L406    L108    L307    L607    L207    L507
 M207    M110    M306    M307    L606    L405    L306    L506    L107    L605
 M101    M201    M301    M102    M103    M202    M104    M302    M105    M203
 M303    M304    M204    M305    M106    M107    M108    M205    M109    M206
 L404    L206    L305    L604    L106    L403    L205    L505    L304    L105
 L603    L204    L303    L504    L104    L602    L103    L503    L203    L302
 L102    L402    L502    L301    L202    L101    L601    L201    L401    L501


Stat for N = 10
 ================================================================================
|    | No of Customers | Got Seat | Returned | Response | Turn Around |
 ================================================================================
|  H |              10 |       10 |        0 | 0.000000 |    1.500000 |
|  M |              30 |       30 |        0 | 0.900000 |    3.800000 |
|  L |              60 |       55 |        5 | 3.509091 |    9.018182 |
 ================================================================================
```

N = 15

```
Final Concert Seat Chart
========================
 H101    H102    H103    H104    H105    H106    H107    H108    H109    H110
 M213    H111    M214    M113    M310    L409    L609    L209    L508    L309
 M211    M111    M212    M309    M112    L108    L408    L608    L308    L208
 M109    M210    M110    L407    L507    L107    L607    L307    L406    L207
 M308    M209    M106    M107    M108    L106    L405    L306    L606    L206
 M301    M101    M201    M302    M202    M102    M203    M303    M103    M204
 M304    M205    M104    M305    M206    M306    M207    M105    M307    M208
 L105    L305    L205    L605    L506    L404    L104    L604    L304    L204
 L505    L403    L303    L103    L603    L203    L504    L503    L102    L302
 L402    L602    L202    L502    L401    L101    L301    L201    L601    L501


Stat for N = 15
 ================================================================================
|    | No of Customers | Got Seat | Returned | Response | Turn Around |
 ================================================================================
|  H |              15 |       11 |        4 | 0.454545 |    1.909091 |
|  M |              45 |       37 |        8 | 2.324324 |    5.270270 |
|  L |              90 |       52 |       38 | 7.730769 |   13.423077 |
 ================================================================================
```

Conclusion: Looking at the simulation results it is clear that there is a large disparity between the response and turnaround times for the H, M, and L sellers. The increased time to process tickets compounds as it not only increases the service time of each individual ticket, but also makes each ticket wait that much longer to get a turn increasing the response and turnaround time. Another thing of note is that as the number of sellers using a particular strategy increased, so too did the search times. As there were more L sellers, the bottom rows were quickly filled up and L sellers had to start searching higher and higher to find available seating. Meanwhile, the H sellers could almost exclusively stay up top in their searches as there were not that many employing their strategy of starting from the top left. This seems to show that the more threads there are competing for the same resources, the more work each thread has to do to acquire those resources. Spreading out the acquisition strategies would most likely lead to threads doing less work. Finally, this project serves to prove that utilizing pthread conditions and mutexes is an effective way to synchronize many threads and to ensure that only one thread can claim any given resource at a time.