ADVANCED OPERATING SYSTEMS

CSEN383

PROJECT-3 (Multi-Threaded Ticket Sellers)

Group – 3

-Aman Jain, Amarnath S Kaushik, Daryl Ho, Meghana Kale, Sai Kiran Jasti

Structs:

1. **Customer**
2. **Seller**
3. **SellerStats**

Functions:

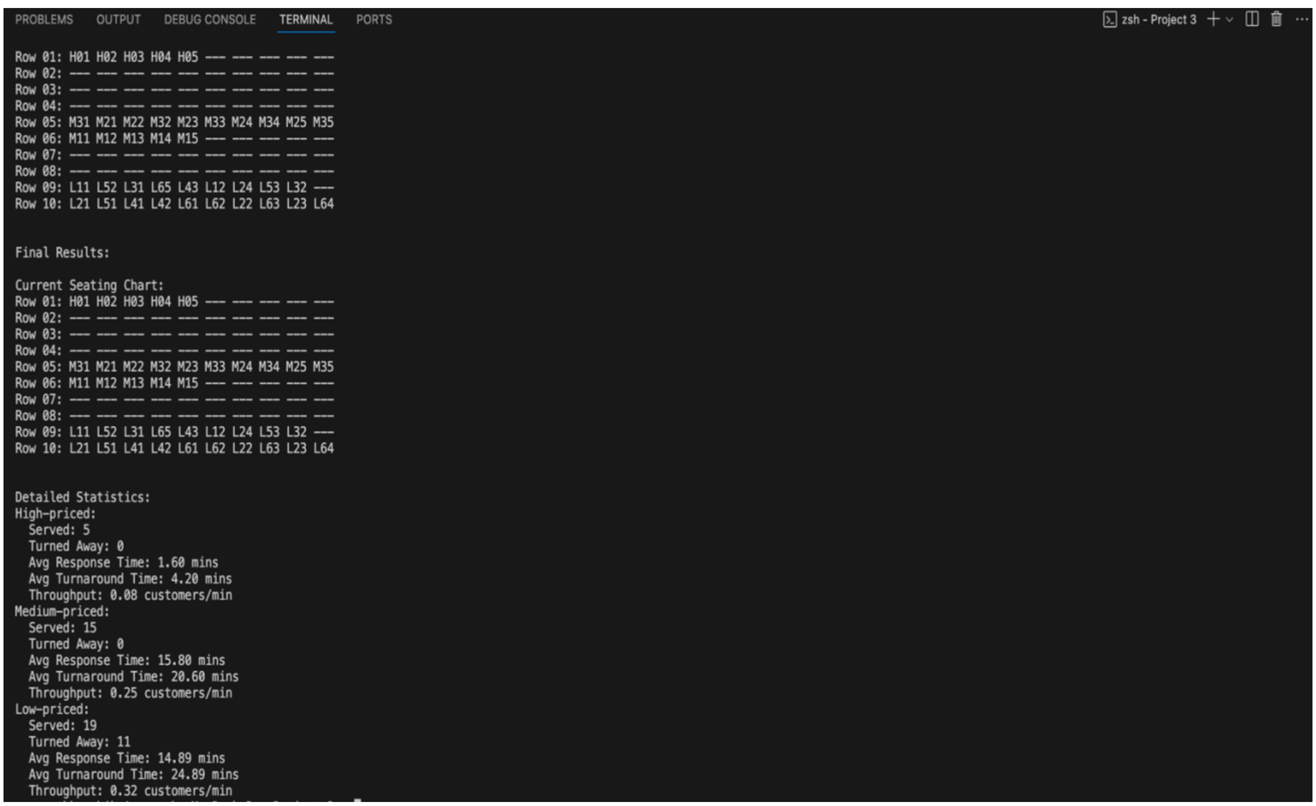
1. **void initialize_sellers(int customers_per_sellers)**
   This function This sets up an array of ticket sellers categorized into high-priced, medium-priced, and low-priced sellers, each assigned a name, type, and a queue of customers. The high-priced seller (sellers [0]) is named "H", while three medium-priced sellers (sellers [1] to sellers [3]) are named "M1" to "M3", and the remaining low-priced sellers (sellers [4] onward) are named "L1", "L2", etc. Each seller's queue is populated with customers_per_seller dynamically allocated Customer instances, each assigned a unique ID and a random arrival time within the simulation time.

2. **int assign_seat(Seller seller, int customer_id)**
   This function allocates a seat to a customer based on the seller's pricing category while ensuring thread safety using mutex locks. For high-priced sellers, it assigns seats starting from the front rows, checking each seat sequentially until an available one is found. Low-priced sellers allocate seats starting from the back rows, moving forward, while medium-priced sellers begin from the middle rows (row 5) and expand outward symmetrically and updates the seating chart if a seat is available, and ensures concurrent access control with mutex locks to prevent race conditions. It returns 1 if a seat is successfully assigned and 0 otherwise.

3. **Void* seller_thread(void* arg)**
   This function simulates a seller processing customers in a queue while tracking service metrics. Each seller runs while global time is within simulation time, serving customers whose arrival times have passed. If a seat is assigned via assign seat, the function simulates service time (sleep), updates response and turnaround times, and tracks served or turned-away customers using mutex-locked statistics. The queue is then adjusted by shifting customers forward. Successful transactions print the updated seating chart, ensuring real-time simulation updates.

Ticketing Systems Averaged Output for 5, 10, 15 Customers:

1. **5 Customers**



```
Row 01: H01 H02 H03 H04 H05 --- --- --- --- ---
Row 02: --- --- --- --- --- --- --- --- --- ---
Row 03: --- --- --- --- --- --- --- --- --- ---
Row 04: --- --- --- --- --- --- --- --- --- ---
Row 05: M31 M21 M22 M32 M23 M33 M24 M34 M25 M35
Row 06: M11 M12 M13 M14 M15 --- --- --- --- ---
Row 07: --- --- --- --- --- --- --- --- --- ---
Row 08: --- --- --- --- --- --- --- --- --- ---
Row 09: L11 L52 L31 L65 L43 L12 L24 L53 L32 ---
Row 10: L21 L51 L41 L42 L61 L62 L22 L63 L23 L64


Final Results:

Current Seating Chart:
Row 01: H01 H02 H03 H04 H05 --- --- --- --- ---
Row 02: --- --- --- --- --- --- --- --- --- ---
Row 03: --- --- --- --- --- --- --- --- --- ---
Row 04: --- --- --- --- --- --- --- --- --- ---
Row 05: M31 M21 M22 M32 M23 M33 M24 M34 M25 M35
Row 06: M11 M12 M13 M14 M15 --- --- --- --- ---
Row 07: --- --- --- --- --- --- --- --- --- ---
Row 08: --- --- --- --- --- --- --- --- --- ---
Row 09: L11 L52 L31 L65 L43 L12 L24 L53 L32 ---
Row 10: L21 L51 L41 L42 L61 L62 L22 L63 L23 L64


Detailed Statistics:
High-priced:
  Served: 5
  Turned Away: 0
  Avg Response Time: 1.60 mins
  Avg Turnaround Time: 4.20 mins
  Throughput: 0.08 customers/min
Medium-priced:
  Served: 15
  Turned Away: 0
  Avg Response Time: 15.80 mins
  Avg Turnaround Time: 20.60 mins
  Throughput: 0.25 customers/min
Low-priced:
  Served: 19
  Turned Away: 11
  Avg Response Time: 14.89 mins
  Avg Turnaround Time: 24.89 mins
  Throughput: 0.32 customers/min
```

2. 10 Customers

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                zsh - Project 3

Row 01: H01 H02 H03 H04 H05 H06 H07 --- --- ---
Row 02: --- --- --- --- --- --- --- --- --- ---
Row 03: --- --- --- --- --- --- --- --- --- ---
Row 04: --- --- --- --- --- --- --- --- --- ---
Row 05: M21 M11 M22 M12 M31 M32 M13 M23 M33 M14
Row 06: M24 M34 M15 M25 M35 M16 M26 --- --- ---
Row 07: --- --- --- --- --- --- --- --- --- ---
Row 08: --- --- --- --- --- --- --- --- --- ---
Row 09: L25 L61 L33 L52 L26 L62 L13 L53 L34 L42
Row 10: L21 L41 L31 L22 L11 L23 L12 L24 L51 L32


Final Results:

Current Seating Chart:
Row 01: H01 H02 H03 H04 H05 H06 H07 --- --- ---
Row 02: --- --- --- --- --- --- --- --- --- ---
Row 03: --- --- --- --- --- --- --- --- --- ---
Row 04: --- --- --- --- --- --- --- --- --- ---
Row 05: M21 M11 M22 M12 M31 M32 M13 M23 M33 M14
Row 06: M24 M34 M15 M25 M35 M16 M26 --- --- ---
Row 07: --- --- --- --- --- --- --- --- --- ---
Row 08: --- --- --- --- --- --- --- --- --- ---
Row 09: L25 L61 L33 L52 L26 L62 L13 L53 L34 L42
Row 10: L21 L41 L31 L22 L11 L23 L12 L24 L51 L32


Detailed Statistics:
High-priced:
  Served: 7
  Turned Away: 3
  Avg Response Time: 9.14 mins
  Avg Turnaround Time: 11.86 mins
  Throughput: 0.12 customers/min
Medium-priced:
  Served: 17
  Turned Away: 13
  Avg Response Time: 16.35 mins
  Avg Turnaround Time: 21.71 mins
  Throughput: 0.28 customers/min
Low-priced:
  Served: 20
  Turned Away: 40
  Avg Response Time: 6.80 mins
  Avg Turnaround Time: 16.35 mins
  Throughput: 0.33 customers/min
```

3. 15 Customers

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS                                zsh - Project 3

Row 01: H01 H02 H03 H04 H05 H06 H07 H08 H09 ---
Row 02: --- --- --- --- --- --- --- --- --- ---
Row 03: --- --- --- --- --- --- --- --- --- ---
Row 04: M39 M14 M21 M15 M21 M31 --- --- --- ---
Row 05: M31 M21 M22 M23 M24 M32 M25 M33 M26 M34
Row 06: M27 M35 M36 M11 M37 M12 M38 M13 M28 M29
Row 07: --- --- --- --- --- --- --- --- --- ---
Row 08: L33 L16 L43 L62 L56 --- --- --- --- ---
Row 09: L41 L54 L14 L24 L32 L55 L42 L25 L15 L61
Row 10: L31 L21 L51 L11 L52 L22 L12 L53 L23 L13


Final Results:

Current Seating Chart:
Row 01: H01 H02 H03 H04 H05 H06 H07 H08 H09 ---
Row 02: --- --- --- --- --- --- --- --- --- ---
Row 03: --- --- --- --- --- --- --- --- --- ---
Row 04: M39 M14 M21 M15 M21 M31 --- --- --- ---
Row 05: M31 M21 M22 M23 M24 M32 M25 M33 M26 M34
Row 06: M27 M35 M36 M11 M37 M12 M38 M13 M28 M29
Row 07: --- --- --- --- --- --- --- --- --- ---
Row 08: L33 L16 L43 L62 L56 --- --- --- --- ---
Row 09: L41 L54 L14 L24 L32 L55 L42 L25 L15 L61
Row 10: L31 L21 L51 L11 L52 L22 L12 L53 L23 L13


Detailed Statistics:
High-priced:
  Served: 9
  Turned Away: 6
  Avg Response Time: 9.44 mins
  Avg Turnaround Time: 11.78 mins
  Throughput: 0.15 customers/min
Medium-priced:
  Served: 26
  Turned Away: 19
  Avg Response Time: 18.88 mins
  Avg Turnaround Time: 24.85 mins
  Throughput: 0.43 customers/min
Low-priced:
  Served: 25
  Turned Away: 65
  Avg Response Time: 19.96 mins
  Avg Turnaround Time: 30.40 mins
  Throughput: 0.42 customers/min
```

Conclusion:

In this project, we implemented a **multi-threaded ticket-selling system** to simulate a real-world ticket booking scenario using **concurrent programming techniques**. By categorizing sellers into **high-priced, medium-priced, and low-priced tiers**, we ensured a structured approach to seat allocation. **Mutex locks** were utilized to maintain thread safety and prevent race conditions when updating shared resources such as the seating chart and seller statistics.

The simulation results demonstrated how different pricing categories affect seat allocation and customer experience. **High-priced sellers** efficiently filled seats from the front rows, while **medium-priced sellers** distributed customers evenly from the middle rows, and **low-priced sellers** assigned seats from the back. The results also highlighted how queue size and customer arrival times influence service efficiency, response time, and customer satisfaction.

By running multiple simulations with varying numbers of customers (5, 10, and 15), we observed that **customer wait times increased** as more customers joined the queue, leading to **higher turnaround times and potential rejections** when all seats were filled. The implementation successfully demonstrated the impact of **thread synchronization and resource contention** in a real-time booking environment.

Overall, this project provided valuable insights into **concurrent programming, thread management, and synchronization**, showcasing the complexities of handling real-world multi-threaded applications in ticket-selling scenarios.