SM Cinema Ticketing System

Project Overview

This project simulates a movie ticketing system for SM Cinema Legazpi. It allows users to view available movies by showtime, select a movie, view available seats, and book or cancel seat bookings. The system also supports priority bookings for senior citizens, where senior customers receive VIP seating.

Key Features:

- **Movie Showtimes Management:** Movies are scheduled with specific showtimes and stored in a priority queue (Min-Heap).
- **Seat Management with Binary Search Tree:** Seats for each movie are stored and managed using a Binary Search Tree (BST), allowing efficient seat booking and cancellation.
- **Priority Queue for Senior Citizens:** Senior citizens' booking requests are handled first, using a Max-Heap to prioritize based on age.
- **User Interaction:** The program allows users to interact via a simple menu-driven interface to select a movie, view seats, book/cancel seats, and manage priority booking.

How to Run the Code

Requirements:

- A C++ compiler (e.g., GCC, Visual Studio, etc.)
- The C++ Standard Library (for data structures like priority_queue and map)
  -The code can be downloaded and run it using a c++ compiler.

Description of Each Functionality

1. Movie Showtimes Management (Priority Queue - Min-Heap)

- **Purpose:**
  - Movies are managed by their showtimes. The movie with the earliest showtime is given the highest priority and processed first.
- **Insertion:**
  - Movies are added to a Min-Heap based on their showtime (earliest showtime has the highest priority). The Movie struct overloads the < operator to ensure that the movies are ordered by showtime (in ascending order).

```
priority_queue<Movie> movieHeap;

movieHeap.push(Movie("Hello, Love, Again", 13 * 60)); // 1:00 PM

movieHeap.push(Movie("Moana 2", 14 * 60 + 30));      // 2:30 PM

movieHeap.push(Movie("The Wicked", 17 * 60));        // 5:00 PM
```

- **Traversals (Min-Heap):**
  - The heap structure allows quick retrieval of the earliest showtime. Movies are displayed by showtime in chronological order, with the heap automatically managing the priority.

- **Searching:**
    - Searching in the Min-Heap is not explicitly implemented, but the priority structure ensures that when movies are processed, they are in the correct order by their showtime.

---

2. Seat Management with Binary Search Tree (BST)

- **Purpose:**
    - This class manages the booking and cancellation of seats using a Binary Search Tree (BST). Each seat is represented as a node containing the seatNumber, isBooked status, and left and right pointers for the BST structure.
- **Insertion:**
    - The insert() method in the SeatBST class is responsible for adding a seat node to the tree while maintaining the binary search property (left child smaller than the parent node, and right child larger).
- **Traversals (Inorder, Preorder, Postorder):**
    - **Inorder Traversal:** The inorder() function prints the seats in order, displaying booked and available seats.
    - **Preorder and Postorder Traversals:** These can be added by modifying the traversal function to visit nodes in the specific order (preorder: root, left, right; postorder: left, right, root).
- **Searching:**
    - The search() function allows searching for a seat by its number. It returns the node if found, or nullptr if the seat is not available.
- **Deletion:**
    - The deleteNode() method handles the deletion of a seat node while maintaining the binary search tree property. If a node has two children, the findMin() helper function finds the inorder successor to replace the deleted node.

---

3. Priority Queue for Senior Citizens (Max-Heap)

- **Purpose:**
    - The PriorityQueue class manages customers with priority, typically senior citizens. The queue stores customers based on their age, prioritizing older customers.
- **Insertion:**
    - The addRequest() method adds a customer to the Max-Heap, with priority determined by the customer's age (older customers are given higher priority).

```
priority_queue<pair<int, string>> heap;  // Max-Heap by default using the first element (age)
```

- **Traversals (Heapify, Max-Heap):**
    - **Heapify:** The priority queue is implemented using the priority_queue in C++, which automatically handles heapifying when elements are added or removed.
    - **Max-Heap:** The Max-Heap ensures that the oldest customer (highest age) is served first. This is done using a pair<int, string> where the first element represents the customer's age.

- **Searching:**
    - Searching for specific customers is not explicitly implemented in the heap. However, since the priority queue always prioritizes older customers, the next customer to be processed is the one at the top of the heap.
- **Deletion:**
    - When processing the next customer, the processNextRequest() function removes the customer from the heap, ensuring that the customer with the highest age is always processed first.

4. Main Menu:

- **User Interface:**
    - A simple text-based menu allows users to interact with the system:
        - **View Movies by Showtimes:** Displays all movies sorted by showtime.
        - **Choose Movie:** Lets the user select a movie to interact with.
        - **View Seats:** Displays available and booked seats for the selected movie.
        - **Book Seat:** Books a seat for the selected movie.
        - **Cancel Booking:** Cancels a seat booking.
        - **Add Priority Request:** Adds a priority booking request for a senior citizen.
        - **Process Next Priority Request:** Processes the next priority request.
        - **View Priority Requests:** Displays all pending priority requests.

Code Overview:

1. **Movie Class:**
    - Stores the name and showtime of each movie.
    - The operator< is overloaded to allow the priority_queue to prioritize the movie with the earliest showtime.
2. **SeatBST Class:**
    - Implements a binary search tree for managing the seats of each movie.
    - Contains methods for adding, viewing, booking, and canceling seat bookings.
3. **PriorityQueue Class:**
    - Uses a priority_queue to manage priority customers based on their age, ensuring that older customers are served first.
4. **Main Program Loop:**
    - The program runs in a loop where users are prompted to select options for managing movie bookings, viewing available seats, and handling priority requests.

We have here the sample output of the code:

```
                  ============================================

                        Welcome to SM Cinema Legazpi!

                        Your entertainment destination!

                  ============================================


--- Main Menu ---

1. View Movies by Showtimes

2. Choose Movie

3. View Seats

4. Book Seat

5. Cancel Booking

6. Add Priority Request

7. Process Next Priority Request

8. View Priority Requests

0. Exit

Enter your choice: 1


Movies by Showtimes:

  Hello, Love, Again - 1:00 PM

  Moana 2 - 2:30 PM

  The Wicked - 5:00 PM


Enter your choice: 2

Enter the name of the movie you want to select: Moana 2

You selected: Moana 2

Thank you for choosing SM Cinema. Your chosen movie, 'Moana 2', will start at 2:30 PM.
```