Challenge: Theater Booking and Showtime Simulation

Problem Statement

Design and implement a Java program to simulate the booking and management of a movie theater's showtime using Java multithreading. The simulation must support multiple patrons trying to book seats, manage seat allocation, and handle different showtime activities such as ticketing and ushering.

Problem Details

1. **Theater Setup**:
   - The theater has 50 seats, organized into 5 rows with 10 seats per row.
   - Each seat is assigned a unique number from 1 to 50.
2. **Roles and Responsibilities**:
   - **Patrons**: Simulate 100 patrons trying to book seats. Each patron is represented as a thread attempting to book a seat.
   - **Ticket Counters**: There are 3 ticket counters (threads), each handling a maximum of 5 booking transactions at a time.
   - **Ushers**: 2 ushers (threads) guide the patrons to their seats once their booking is confirmed.
3. **Booking Process**:
   - Patrons arrive at random times and queue for booking.
   - Each patron requests a seat and waits for confirmation.
   - Ensures that no seat is double-booked.
4. **Showtime Activities**:
   - After booking, patrons proceed to their seats in a controlled manner.
   - Ushers guide patrons in different rows sequentially.
5. **Multithreading Aspects**:
   - Implement thread synchronization to ensure proper booking and seat assignment.
   - Use thread-safe collections to handle bookings.
   - Manage threads to simulate delays and ensure orderly seat handling.
   - Ensure threads terminate gracefully after all tasks are complete.

Requirements

- **Concurrency**: Use threading to simulate the parallel processes of booking and seating.
- **Synchronization**: Use appropriate synchronization mechanisms to avoid data consistency issues.
- **Randomness**: Introduce randomness in patron arrival times to simulate a real-world scenario.
- **Error Handling**: Handle potential exceptions from thread interference or incorrect resource handling.
- **Logging**: Implement logging to track seat bookings and patron seat assignments.

Deliverables

- A Java implementation of the described theater simulation.
- Documentation explaining how your program meets the challenge requirements.
- A series of test cases demonstrating successful and unsuccessful booking scenarios.

Tips

- Explore Java's concurrency utilities (`java.util.concurrent` package) for efficient thread and resource management.
- Consider edge cases like all seats being booked or patrons arriving late.
- Use proper thread management techniques (e.g., `sleep` or `join`) to simulate delays realistically and ensure sequential seat filling.

This challenge will deepen your understanding of Java multithreading by applying concepts in a realistic and practical scenario. Enjoy developing your solution!