Challenge: Airline Check-In and Boarding Simulation

Problem Statement

Design and implement a Java program that simulates the check-in and boarding process of an airline using Java multithreading. The simulation involves handling passengers as they arrive at the airport, process through check-in counters, and eventually board the aircraft, all of which must be managed concurrently.

Problem Details

1. **Simulation Setup**:
    - The program simulates a fixed number of passengers (e.g., 50) for a flight.
    - There are multiple check-in counters (e.g., 3) that can handle a limited number of check-ins concurrently.
2. **Roles and Responsibilities**:
    - **Passengers**: Each passenger arrives at the airport, checks in at a counter, and then waits to be boarded onto the aircraft.
    - **Check-In Counters**: These counters, represented by threads, process passengers' check-in requests. Each counter can handle a certain number of passengers at a time.
    - **Flight Attendant**: Manages the boarding process, ensuring all checked-in passengers are guided to their respective areas before the flight departs.
3. **Check-In Process**:
    - Passengers arrive at random intervals and are queued for check-in.
    - Each check-in counter processes a passenger and marks them as checked-in upon successful completion.
    - Implement concurrent processing with locks to ensure the thread-safe operation of check-in counters.
4. **Boarding Zones**:
    - Post check-in, passengers are assigned to specific boarding zones (e.g., zone 1, 2, and 3).
    - The flight attendant threads manage zone-specific boarding, where passengers must wait to be escorted onto the airplane.
5. **Multithreading Considerations**:
    - Synchronize access to shared resources, such as the passenger list and counter queues.
    - Manage thread life cycles to ensure all operations (check-in, boarding) are completed before the simulation ends.
6. **Real-Time Simulation**:
    - Introduce random arrival and processing times to simulate real-world scenarios.
    - Ensure passengers' sequential boarding without data races or synchronization issues.

Requirements

- **Concurrency**: Implement threading to manage parallel check-in and boarding operations.
- **Synchronization**: Use appropriate synchronization techniques to safeguard shared data among threads.
- **Randomization**: Simulate realistic delays in passenger arrivals and processing.
- **Logging**: Include informative logging at each stage to track passenger progress and system state changes.

Deliverables

- A Java implementation of the airline check-in and boarding simulation.
- A detailed README explaining how to run the simulation and what the key learning objectives are.
- A set of test cases illustrating both successful and edge-case scenarios.

Tips

- Explore Java's `java.util.concurrent` package for efficient thread management.
- Focus on ensuring data integrity and smooth transitions between different stages of the simulation.
- Pay attention to the handling of thread termination and resource deallocation.

This challenge will provide hands-on experience with Java multithreading, highlighting its use in simulating real-world processes and the complexities of concurrent programming.