Preventing Deadlock:
Deadlock is avoided in this program by carefully controlling the order of semaphore signals and waits. The semaphores are used to control the flow of the car and passengers to ensure that they wait and signal at appropriate times.

To prevent deadlock:
- Semaphores are used to synchronize the boarding, unboarding, and signaling between the car and passengers.
- Mutual exclusion is maintained using a mutex semaphore to protect critical sections of code where shared variables are accessed.
- The order and placement of semaphore signals (sem_post) and waits (sem_wait) are structured in a way that prevents deadlock situations.

Deadlock can occur in a multithreaded system when threads are waiting for each other indefinitely. In this program, deadlock is prevented by ensuring that threads release the necessary resources (semaphores) before waiting for others to proceed.

Note:
- This program is a simple simulation and might not cover all real-world scenarios or edge cases.
- The sleep() function is used to simulate time delays and does not accurately represent real-time.

Assumption:
Not necessary to mention the name of the offboarding passengers