1. **How do you ensure specification #12, “Students cannot enter the room (S.3) while the landlord is inside the room (L.2)”?**

The way I approached this specification is by not allowing students to enter nor leave while the landlord is inside the room is done by disabling the progression using semaphores and mutex locks for students. The landlord uses a mutex lock (printMutex) before checking the room and semaphores (enterSemaphore, exitSemaphore) control student entry so that students wait until the landlord completes their check and exits before attempting to enter.

2. **How do you ensure specification #6, “If the landlord breaks up the party, he will not leave (L.4) until all students have left the room”?**

The way I approached this specification is by having a global counter and using a loop and synchronization mechanisms. When the landlord decides to break up the party they will continuously check studentsInRoom inside of a while loop until reaches zero before proceeding. The landlord here remains blocked inside this loop to prevent them from leaving until all students have exited to enforce that eviction completes before the landlord continues execution.

3. **How do you ensure specification #17, “The landlord retiring (L.7) is the last statement printed by the program”?**

The way I approached this specification is by using a controlled execution order and proper thread synchronization where the landlord only prints the final retirement message after all students have been evicted and totalProcesses reach 0. I also use semaphores and mutex locks so that the landlord termination happens only after all student threads have completed.