**CS 4348 Project 3 Summary**

The project requires simulating the behavior of a hotel, including guests, front desk employees, and bellhops. The simulation will use threads and semaphores to model the interactions between these entities. To start with, the simulation will create 25 guest threads, two front desk employee threads, and two bellhop threads. Each guest thread will be assigned a random number of bags (between 0 and 5) and will check in at the front desk to receive a room number. If the guest has less than three bags, they will proceed directly to the room. Otherwise, the guest will visit the bellhop to drop off their bags. The guest will later meet the bellhop in the room to retrieve their bags and give a tip.

The front desk employee threads will be responsible for checking in guests and assigning them a room number. They will keep track of the available rooms and we use semaphores to ensure mutual exclusion when accessing the room information. The bellhop threads will be responsible for accepting bags from guests and delivering them to the rooms. They will also receive tips from the guests when delivering the bags. The bellhops will use semaphores to ensure that only one bellhop can deliver bags to a room at a time.

The guest threads will perform a series of actions, including checking in at the front desk, visiting the bellhop, entering the room, and retiring for the evening. They will use semaphores to coordinate with the front desk employees and bellhops, ensuring that they do not attempt to access the same resources simultaneously. The simulation will end when all guests have retired for the evening. The main thread will join all the guest threads and output a summary of the simulation.

The main challenge encountered in this project was designing a robust and efficient system that could handle multiple threads accessing shared data structures simultaneously. To address this issue, semaphores were used to control access to critical sections of the program.

Through this project, we learned how to use semaphores and threads in Java to coordinate and manage multiple concurrent tasks. We also learned how to design and implement a complex system that requires the coordination of multiple threads and semaphores to ensure that the system runs correctly.

The simulation was successful in simulating a hotel environment, with customers checking in, using hotel amenities, and checking out. The employees managed room availability, cleaned rooms, and provided customer service. The simulation also produced meaningful results, such as the total number of customers who stayed at the hotel, the average length of stay, and the average time customers spent waiting for service.

In conclusion, this project provided a valuable learning experience in coordinating multiple threads and semaphores to design and implement a complex system. The simulation successfully modeled a hotel environment, and the results obtained were meaningful and useful. This project highlights the importance of designing and implementing efficient and robust systems to ensure that complex simulations run correctly.