Working on my hotel simulation project was a truly enlightening experience. I found that the debugging phase took up a significant portion of my time, more so than the actual coding. It was a real challenge to pinpoint where deadlocks were occurring after I implemented the semaphores. Making sure that the correct thread entered the critical area at the right time, showcasing true concurrency, was another major hurdle. One of the most intriguing aspects was the way in which guests were assigned rooms. It wasn't a straightforward "first come, first served" scenario. Instead, rooms were assigned based on when guests arrived at the hotel, adding an interesting layer of complexity. The interaction between the bellhop and the guests with more than two bags added another dimension to the simulation. Watching the bellhop efficiently deliver the bags to the guest's room while also receiving a tip was quite satisfying to see in action. Throughout this project, I gained a deep understanding of how concurrency and multithreading operate. Dealing with deadlocks was, at times, quite frustrating. It was frustrating to see correct outputs during debugging, only to encounter completely unexpected results during actual testing. Figuring out which semaphore was causing the issue and repositioning it in the correct spot was a major breakthrough. I also learned the value of creative debugging methods. Traditional techniques often fell short when dealing with the intricacies of semaphores and multithreading. This experience pushed me to think outside the box and try alternative approaches. In retrospect, this project not only honed my coding skills but also provided a profound education in managing concurrency and multithreading. I'm certain that this newfound knowledge will be invaluable in future projects involving parallel processing and resource management.