Nate Idica

CS-273: Data Structures

Dr. Pete Tucker

19 July 2019

Final Summary

The simulation I decided to pick for the final project of Data Structures, was a week in the Emergency room. Admitted patients were at random and based on the rate that the user was inputting. Patients were then passed through the waiting room to the treatment room to discharge via a doctor or nurse queue, depending on the patient’s illness levels.

Some of the challenges or things that I had noticed about the code was that first, the amount of first names and last, or sur, names were not equal. There were 2000 first names and 1000 last names, which lead to an error. To fix this, I copied all 1000 last names and doubled the text file to include two of each last name, so that the amount of first names matched the amount of last names. Another miniscule problem I had was trying to generate the RandomGenerators class, where I had to refresh my memory on srand, the right headers to use, and using other functions deal with the uncertainty of the class being, for the most part, random.

When comparing the average patient visit time when the emergency room has 1 doctor and 1 nurse, 1 doctor and 2 nurses, and 2 doctors and 1 nurse, the most efficient or fastest average patient time would be with 1 doctor and 2 nurses. The way that I had programmed the code, the patients will illness levels of 1 to 10, would go to nurses and levels of 11 to 20, to go to doctors. The program was more highly more likely to give a patient an illness level between 1 to 10, which can explain why the program would run more efficient and have faster average patient visit times in the emergency room with 1 doctor and 2 nurses.

Further comments or things I would have done completely differently is coming up with a better pseudo-code and using other names for variables, vectors, and queues in the program. Looking back at the code that I had written for this simulation project, I think that using other names for variables, vectors, and queues in the program would have made the program look a little less confusing. For example, I would use DischargeQueue \*dischargeQueue. I also think that coming up with a more extensive and detailed pseudo-code would have helped me organize my thoughts a little more, when it came to writing the code for the project. After finish the code implementation for the Simulation Project and realizing what I could have done to better this, I could use for future learning experiences to better myself in learning more coding.