Leia Wang
Jayne Oentoro
Neil Jolly Malayil

Project Final Report

Division of teamwork

Logistics

As with the milestone, we worked together on the different sections of the final deliverables. However, we had to adjust our collaboration style due to the inability to physically meet. There were also other factors that made working on this project difficult. For one, we were all in different time zones. Also, we were unable to contact Neil for the first two weeks of the final project as he was admitted to the self-isolation ward. However, using collaboration software such as Zoom, Facebook, Google Drive etc., we were able to communicate. We also took on different levels of responsibility at different times as needed.

Deliverables

1. Program - Leia first modified the program used for the milestone to fit the updated objective of the final. Jayne then edited it and separated it into 3 separate files. Then all members of the group looked over it for accuracy and to determine the best structure.
2. Report - Leia created the report structure and wrote the initial draft for each section. Jayne and Neil added to and edited the report as well.
3. Jupyter Notebook and Analysis - Jayne and Neil edited the program to output data into csv files to analyze with the Jupyter notebook. Neil and Jayne created a Jupyter notebook including the graphs. Leia then added in the description text and comments.

General Assessment

Overall, we were able to overcome the challenges and worked well as a team. Each team member was supportive and contributed to the best of their abilities, which allowed us to complete the project successfully

Program Design

Structure

We chose to structure our program into 3 separate files so that the program is logical and others can navigate it easily, and so that the code is readable with limited redundancies. We have given an overview of the purpose of each of the separate files below:

1. Main - This file initializes lists to contain group and person objects as well as the headcount for all groups. It imports the class method definitions from final_project_classes.py, which together with the given group csv file, allows the program to create and populate group objects with person objects. Subsequently, it counts the number of people in each group to fill in the headcount list. As this file also imports method definitions from final_project_functions.py, the headcount list is then used to feed the monte carlo method a range of x values to run their corresponding simulations. Ultimately, it finds and prints the maximum x as the output, while also saving the maximum bed demand for each x value for further analysis.
2. Classes - This file contains 2 class definitions:
    a. The Group class takes in variables that characterize the group and has methods that relate to the infection status of the group.
    b. The Person class takes in variables that characterize each person within each group and also has methods that relate to the infection status of that person, including counting the number of days infected.
3. Functions - This file contains the given input data and two functions. The input data includes the number of initial infections, the number of days before hospitalization and recovery and the maximum number of beds a hospital has. The first function is the simulate function which simulates one instance of the epidemic. It does so by continually checking and changing the status of each person in each group, as people change from susceptible, to infected, to hospitalized to recovered, until the population of infected people becomes 0. It also updates the total counts including the number of beds being utilized at the hospitals and closes groups that have a person hospitalized or if the group's headcount exceeds the threshold. The second function is a monte carlo function that runs the simulation multiple times.

Changes Made

The final project differed from the milestone in that we needed to simulate under three scenarios of early, late or very late infection and we also added a hospitalization component. Although our objective was still to find the maximum “x” or headcount of group size allowed to remain open, we now had an additional constraint of the maximum number of hospitalizations allowed at any time. To implement these changes into our program, we made the following changes:

- Added and set the variable “h” to 7 to represent the 7 days of infection before a person begins hospitalization
- Added “hospitalized” as another possible value for a person’s status (data attribute)
- Added and set the variable “max_beds” to 50 as a parameter
- Closed a group if any (member) person’s status was “hospitalized”
- Created a counter for the number of beds_occuppied and added to this number as people became hospitalized and subtracted as they recovered
- Changed the return value of the simulate function to the maximum number of beds needed for the entire simulation of an epidemic for the given “x”
- Added a statement to check whether the return value for the maximum number of beds needed is no greater than 50 to validify that the given “x” is a feasible solution. If the given “x” was feasible and it was larger than the last valid “x”, the given “x” would be saved as the maximum “x”.

To improve our program based on feedback from the professor, we made the following adjustments:

- As our milestone program was slow, we re-optimized the program by cutting down the number of boolean variables we had in the program by introducing “break” in our loops to cut down on processing time after the boolean is satisfied.
- Based on our milestone feedback, the code was changed to only iterate over groups with infected or hospitalized people to check for infections. We did this by creating a list of infected groups at the beginning of each “day”, which ultimately allowed the program to be slightly quicker as it removed the redundancy of checking groups that do not have any possibility of an infection spread. Additionally, we also updated the days infected in a separate loop from the one that had checked for infections to remove the need for the ‘updated’ data attribute of the Person class. As a result, we no longer needed to change the updated data attribute to false for each new day of the pandemic.