p3

Bowen Lyons

Project Description

The purpose of this project was to create a graph that modeled the spreading of a disease. To accomplish this required knowledge of graphs and graph traversal strategies. Specifically my program utilized Depth First Search and Dijkstra's algorithm.

This program creates an adjacency list of nodes labels PN where N is an integer and can range from 0 to something less than infinity (hopefully). It then finds all connected components and calculates their average degree and the amount of vertices. After processing the graph information the program simulates an outbreak of a disease and reports back the count of individuals with different states according to the SEIR model. The program stops after the number of infected and latent people is 0. Then the program asks the user for two nodes an infected and a susceptible node. Using Dijkstra's algorithm the program calculates the distance from the infected to the non infected node and returns the amount of days it would take for the former to infect the latter.

Data Structures

For this program I created a struct called node that contained the name of the node, the count of days passed, the state of the node, and a vector of node pointers that held the nodes that it was adjacent to. I then used a vector of these nodes to create the graph and utilizing a Depth First Search I separated that vector of nodes into a vector of vectors of nodes that contained each component of the graph separately. This made finding if two nodes were connected efficient and easy.

I aired on the side of adjacency list vs adjacency matrix implementation almost immediately. This choice was because an adjacency matrix implementation promotes searching and analyzing distance and display of data. (I keep typing semicolons instead of periods so if you see a semicolon please regard it as a period, I guess that comes with the territory).

System Functionality

The program initially can take in the configuration file as a command line arg or will prompt for the user input if one is not passed in. Using getline the program then reads each line of the file and stores the population file, infectious and latent periods, and the source of the infection PATIENT ZERO. After closing the config file the program then opens the population file extracts the name of the node and stores the rest of the string in a vector to be added as the adjacency list in the next loop. After storing each of the nodes the program then enters a for loop that iterates through the adjacency list of each node and adds a pointer to each of the nodes into the vector of node pointers for that node (that was confusing to type I hope I'm explaining properly). After this step the entire file and nodes have been created so utilizing a depth first search the program stores each component of the graph into the vector of vector of node pointers that is the proper representation of the graph.

The program then enters a while loop that continues to iterate until the count of E and I state nodes is 0. It displays the info after each day to the user.

Once again your assignment has proved to be challenging and educational, thank you.