Jake Deeks, Nick Nicoletti, David Thompson
Jospeh Helsing
CSCE 2100.002
11/5/2018

Homework 2 Design Document

Design Process

The design process that our group took was figuring out what our algorithm was going to be through discussions and visuals drawn on the whiteboards. There was some initial rework needed to be done as we coded as we found some logic errors. As a group, we decided to make a class called person that would hold each variable that would be needed to be either tracked or displayed during the subsequent days of infection. The variables that we decided on was a character "Status" that would be the display value on what a person would be, an integer "infectThresh" that will hold the read in threshold for a person to be infected, and an integer "timeInfect" that would keep track how long a person has been infected. We proceeded to make a vector of a vector of the class Person to make a 2-dimensional vector to use the class variables in proper fashion. This was what we decided to use to be able to manipulate each position of the vector and progress each day of the infection in the proper manner.

Data structures in system

The data structures we used to start with was a class called "Person" as mentioned in design process. We decided to use the ideology of the class to be able to store multiple things for each position in the input matrix of the display of infection. Next, we created a vector of that class to be able to manipulate and change the positions in the matrix of infection properly. However, to get the proper set up for the two-dimensional matrix needed for the values, we needed to create a second vector of the previously created vector. The reason we did this was to be able to get the 2D functionality of a data structure while keeping the manipulating functions of the vectors and the multiple stored values per position of classes. Vectors was decided to be the best data structure to use for our algorithm due to the easy and manageable manipulations one can do with the built-in functions that come with vectors. We had numerous multiple variables for each of the functions that we used in our program as well as some passed in when calling functions. We did this because some of the variables needed to be used by multiple functions we had while others had no reason to leave their local domain. Out of the local variables, this included the counters that held the ever-increasing value of the day of infection and temporary values that can hold values found and generated by each function.

Functionality of System

The functionality of the system was sort of like our last project's ideology in how we would go about the data stored in the system and how we manipulated it using vectors. We decided to check the matrix of people of infection by initially creating a vector of a vector of the class "Person", so each position can hold multiple values that we can use as flags to properly make our check functions easier. We decided to use ifstreams to properly read in input files and cout to properly print out the necessary

variables. An example of the flags that we can use in our checking and manipulating functions was that after reading in the necessary initial variables needed for the problem. We would read through each position to find out if they were infected, susceptible, recovered, and recovered agents and increase a counter of agents that were found to be infected and then would use an algorithm to increase the threshold counter of each agent surrounding that infected agent using the Moore neighbor configuration, yet we wouldn't increase that position's counter if that agent was either recovered or vaccinated. We would continue to search the matrix until we get through the whole matrix and followed the same process just mentioned. Then, we would go through back through the data structure looking at each specific position's infected threshold that was increased with the previous process. If each position's infected threshold was either equal to or greater than the infected threshold given at the start of the input file, that would mean that position would be changed from susceptible to infected for the next day. This same process can be used to change the infected to its desired agent, as in recovered or vaccinated, for the next day by manipulating the counter of "timeInfect" and once it reached 0 as the days progressed you would change the agent. You would keep repeating the processes mentioned above until there is no more infected left in the matrix.

Work Shares

For the work shares of the project, we Followed the same procedures as we did for the first project since we are in the same group. We continued to meet up as a group using the computer rooms at Discovery Park to meet up and discuss/work on the project during our 2-hour free time between our shared Logic Design and Computer Foundations class on Monday and Wednesday. We did not have to meet on any other days due to not overcomplicating the problem and having to start over from scratch due to solving for the wrong thing. During the meetings, every member in the group was in attendance and properly suggested ways and methods we could use to solve the project. There was examples and suggestions used through visuals on the whiteboard as well as detailed discussions on how we wanted to complete the project.

As we met up as a group quite often, there are some similarities in the work share between each individual member in the group. As a group, we all pitched methods, theories, possible logic checks, how to work the functions, when we met as a group. Nick did most of the typing for the program as we could not work on the program at the same time. Sometimes David did type the program because he debugged or tried to find some in his on time outside of our group meetings and had the most updated version with him. Jake typed up the Design document while David and Nick inputted their thoughts to be able to complete the document.