Jeremiah Spears

CSCE 2100

11/9/18

Project 2 Design Document

Design Process:

When I first started planning out my project, I decided that I was going to follow the suggested methods given in the project instructions. I knew this would be the best way for me to choose the best data types and structures for each problem. I then broke down each step of the program and worked on each step one by one. For example, I knew the first step would be to read the input file, so I started out making a simple program that would do that. I decided to break up the majority of the program's functionality into functions, so that it would be easily readable and more modular.

The main steps I created to design my program (in order):

- Reading the input file.
- Creating a struct to hold element data.
- Creating a 2D vector to store elements.
- Printing a properly formatted grid.
- Counting the number of each state.
- Changing an agent's state to recovered.
- Outputting the data

As I continued to work on the program, I discovered more and more issues/things I had overlooked. Planning out each step helped overcome these issues much more easily. Another method I used to design my implementation, was writing out each process on paper. This made it much easier to understand what algorithms would be required to do functions such as find the neighbors for each agent in the program or printing the grid to the user. It also helped me keep track of what functionality I still needed to add.

Working on Project 1 in the past I had many issues working with my group. For this project I decided that I would work on the project alone, since I did not know anyone well enough to trust them

to be a project partner. This proved helpful as I was able to work out each problem by myself and implement a solution exactly how I wanted.

Data Structures:

For my program implementation I used two different structs. I decided to use structs instead of objects because for the small amount of data that they needed to hold, they were much simpler to implement. The first struct 'agent' holds an integer for the infectious period of the agent's state and it stores the actual state itself as a character. The character states are represented as follows: 's' for susceptible, 'i' for infected, 'v' for vaccinated, 'r' for recovered, and 't' for temporary. The temporary state is set each time the program checks the states for a given day. It is set to all 's' agents that do not have at least the threshold number of infected agents in their neighborhood. This makes sure that the program will only update the agents with the required threshold.

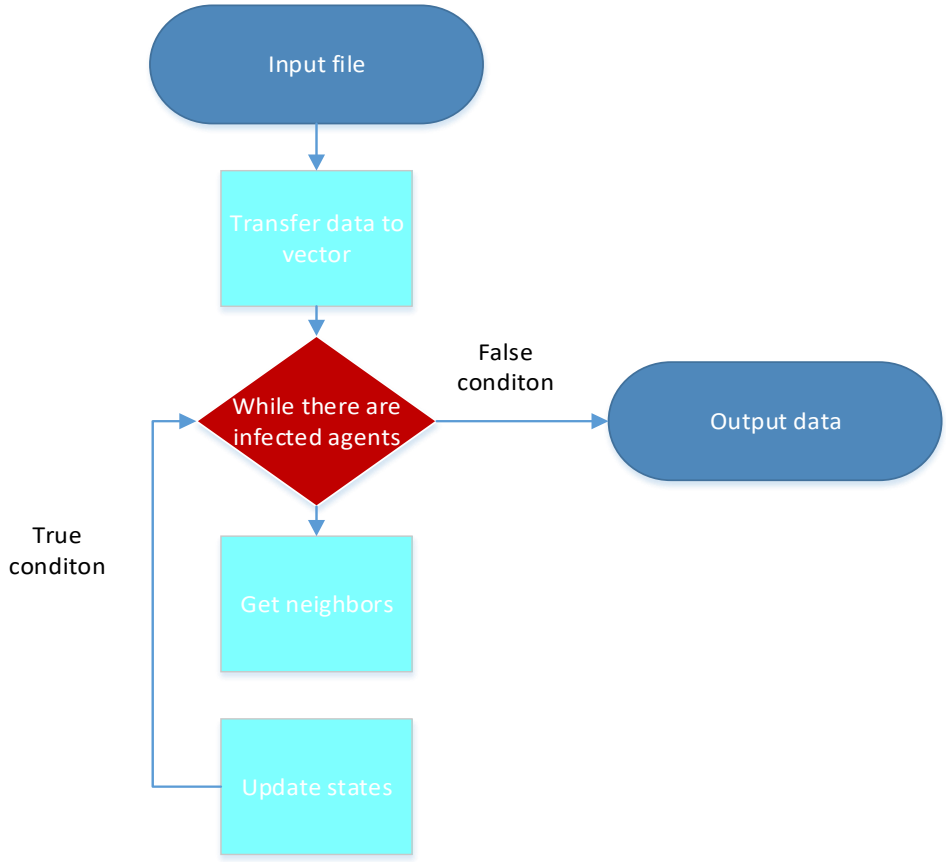
The second struct in my program, peak, holds a pair of integers: peakDay and peakAgents. These integers represent the number of infected agents for each day and the corresponding day. I decided to use a struct for this so that I would be able to store the two values together easily.

The program utilizes many different data types and structures. The main storage for the grid of agents is done by a 2D vector. Using a 2D vector instead of a 2D array allows the program to easily work with any size input file. The vector is a vector of struct agents which allows it to store both the status of each agent, along with the remaining infectious period for any infected agents. To go along with the vector, there is a second vector which is used to translate the data from one day to the next in the program.

Since the program uses so many different variables and functions, I tried to keep the number of global variables to a minimum. This enables to program to be more efficient with its complexity and memory use.

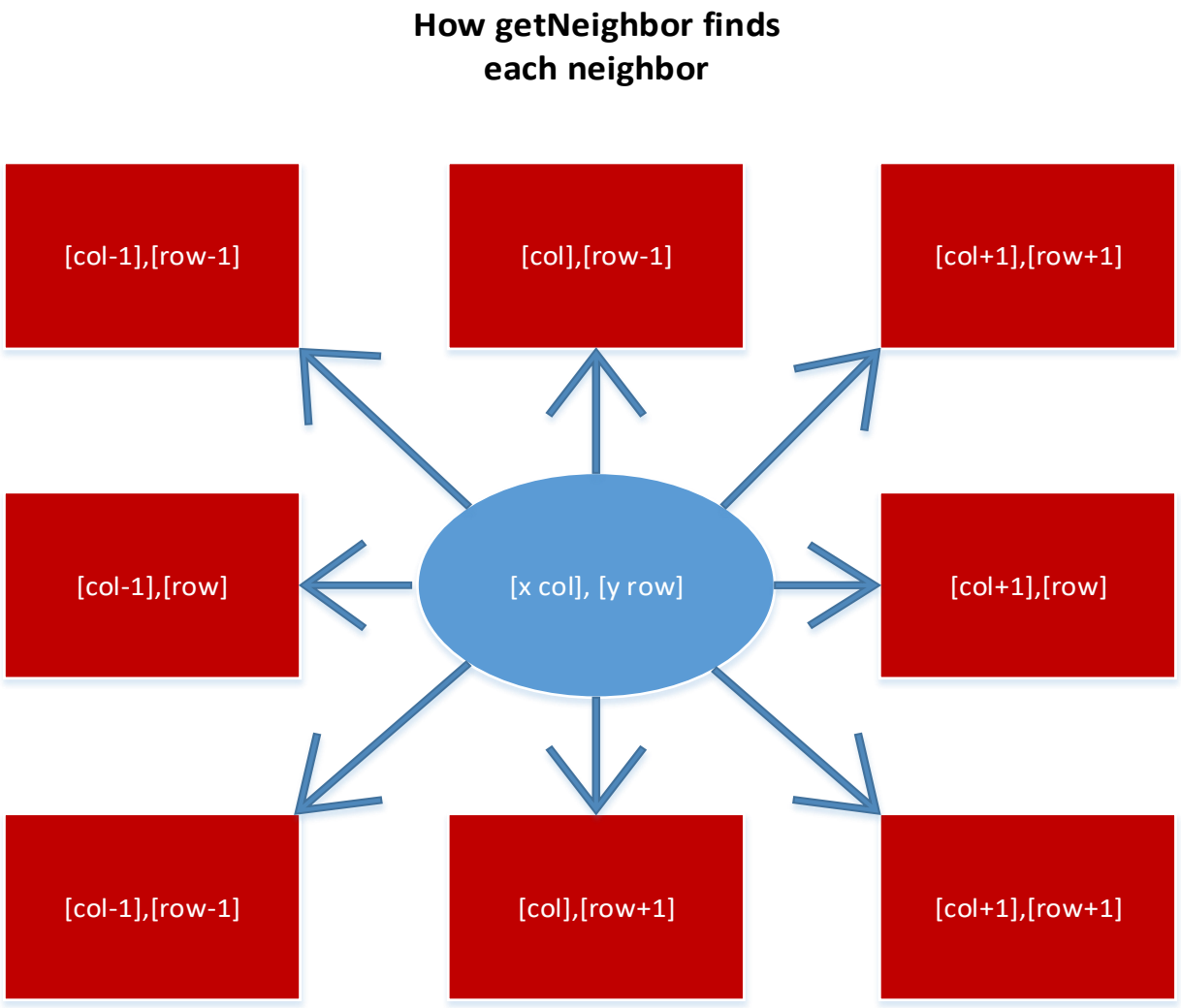
Functionality:

The program can take any user selected input file (with the correct formatting) and then store the data in a vector. It is also able to read the file getting the threshold, infectious period, and display period. The program will also get the size of the grid from the input file. Once the data is stored in the vector of agents (vecA) the 2D vector of agents (vec) will be resized to the correct dimensions for the grid, and it will next be populated with the agent data.



Once the data has been properly stored, the program will then output the initial state of the agents using the function printGrid. The program will then perform the calculations needed using a while loop in the main function. Next the function getNeighbors will find all neighbors for any infected agents, and will update all susceptible agents in it's neighborhood that have the required amount of infected agents in their neighborhood. To do this the program will check all susceptible agents to see if

they have the correct number of infected agents in their neighborhood. If a given susceptible agent does not have threshold number of infected agents nearby it's state will be temporarily changed to temporary so that it will not be affected in the calculation. The getNeighbors function will also decrease any infected agent's infectious period for every day until it reaches 0. After the getNieghbors function, the program will then use the recovery function to update any agents to recovered if their infectious period has reached 0.



After the correct neighbors have been found, the program will then update the state for all agents and store the new grid in the secondary 2D vector (vec2.) After the calculation has been made, it will copy the secondary vector back to the primary vector and will then count the infectious agents in the full grid. The program is able to output the full grid for any given day that fits the display period. If the number of infectious agents for a given day is 0, this lets the program know that it no longer needs to run the main function and it can now output the data.

To output the data the program uses one final if loop. The program will output the following results to the user: the day the outbreak ends, the day of the peak outbreak, and the final counts of each state in the grid.

Group work share:

I worked by myself on the full project and created both the code and design document for the project.

Although I did find it easier to work by myself on the project, in the future I will be more open to working with a group again. I think I would be able to learn more from seeing how different people solve problems through programming.