****

Tuyet Pham & Burton Fox

disease modeling

**PROJECT 2**

|  |  |  |
| --- | --- | --- |
| **Data type in main()** | | |
| **Name** | **Functionality** | **Type** |
| **FILE\_** | This function allows the user to read in the file. | fstream |
| **done** | This function allows the user to break out of the do - while loop. |  |
| **choice** | This is use to give the user an option of which file they would like to choose. | char |
| **leave** | This if the user wants to leave the program entirely | bool |

|  |  |  |  |
| --- | --- | --- | --- |
| **Functions** | | | |
| **Name** | **Functionality** | **Parameter** | **Return type** |
| **menu2()** | Prompt the user to type in their input file. | none | string |
| **menu()** | Prompt the user to pick a choice between 1-5. | none | char |
| **exitprompt()** | Ask user if they want to exit. Reset atRow to 0; | none | bool |
| **vfromstring()** | Getting the entire line and separating each char in order to create a 1D vector of person struct and returning it back to fileReader. | string | vector |
| **change()** | To change the chart according to the conditions given | int, int | none |
| **display\_vec()** | To display the vector only. Take in the display condition and print accordingly, day by day. | int | void |
| **fileReader** | To read the file in and create a 2D vector(the grid). And then putting the 2D vector into a 3D vector(to store each day’s grid) | fstream | bool |

|  |  |  |
| --- | --- | --- |
| **Data type in menu2()** | | |
| **Name** | **Functionality** | **Type** |
| **Name\_o\_file** | This is use for the user to put the name of their text file. | string |

|  |  |  |
| --- | --- | --- |
| **Data type in menu()** | | |
| **Name** | **Functionality** | **Type** |
| **Choice** | Holds what the user had picked and return it back to main. | char |

|  |  |  |
| --- | --- | --- |
| **Data type in exitprompt()** | | |
| **Name** | **Functionality** | **Type** |
| answer | Prompting the user to make a choice to continue or exit out the program. | char |
| done | Once completed reading the options giving. This will allow the user to continue throughout the options until a decision is made. | bool |
| atRow | Holds the number of row being made. Will be = 0. If the user wants to exit. | int |

|  |  |  |
| --- | --- | --- |
| **Data types in fileReader()** | | |
|  | | |
| **Name** | **Functionality** | **Type** |
| **line** | This function is being use for the program to grab the entire line of each row. | string |
| **it** | This function is being use to point to a specific point. | iterator |
| **ROWvec** | This function is being use to display the size of the program. | vector |
| **Currvect\_** | A 2D vector that will accepts the ROWvec begin made by function vfromstring | Vector  <vector  <person>> |
| **dayStack** | A 3D vector that will hold the 2D vector(each day). This way if the user wants to display every four day, we can just show every 4th index. | Vector  <vector  <vector  <person>>> |

|  |  |  |
| --- | --- | --- |
| **Data type in vfromstring()** | | |
| **Name** | **Functionality** | **Type** |
| row | This function is being use to look at particular position in the program. | string |
| person | This function will be the starting position of our character as it goes through changes. | vector |

We will discuss the change() function and display() function further below.

Project 2 has been a tedious task for this group. There were numerous of thoughts given about how to approach this project being something new for both members. There were numerous times where we would meet up in the library and just have a discussion about how to even approach the project. Strategies that work for us was talking about what functions will be best to use for this project, if we even needed a function/classes/struct/ or just write everything in main – kidding – but you can’t rule out everything. Through each time in a discussion a milestone was achieve where both members were able to get a full understanding of what needed to be done.

|  |
| --- |
| Tuyet Pham mid-response to project 2 |
| * The implementation is challenging to say the least |
| * I take it as a challenge |
| * It was interesting thinking up some weird ways to go about the implementation |
| * What was most difficult was the changing of each individual person/square. |

|  |
| --- |
| Burton Fox mid-response to project 2 |
| * I personally felt this was difficult. |
| * It was simple to talk about what should happen. |
| * It was difficult to put things together. |
| * My group member really showed me how to go about things. |

We were able to find an approach to the project that gives us the outcome that were searching for. Whereas we start with ‘S’ and those surround by ‘I’ immediately get infected. A continuous process of this continues after a amount of days one will ‘R’ (recover). This will be a continuous process until every individual has reach ‘R’. As a team we have reach the goal where this project is clear, concise and easy for the user to read and follow.

After a few discussions we were able to agree on a few elements that help us achieve this wanted outcome. The first thing we had decided on was a **struct person.** The struct person would act as the individual character in the square region i.e. the ‘s’, ‘i’, ‘r’ and/or ‘v’. Although structs were meant to be public overall, we had to ensure that some variables remain private to prevent any unwanted state changes. The private members are listed below -

* char state = 'Q'

Each person’s state will be initially set to Q. This is to catch a misreading when setting the initial state. If something was wrong, we can fix it from the beginning.

* int initset = 0

This is to ensure that the initial state was already set to one specific state. It cannot be initially set again. The struct and its members ensures that there was an initial state and will only change state linearly when told to do so. i.e.

The main members that are responsible for these actions are –

changestate()

setinitstate()

changestate()

setinitstate()

setinitstate ()

changestate()

* bool setinitstate(char currstate)
* bool changestate()

setinitstate()

* char getcurrentstate()

The **setinitstate** is responsible for setting the initial state of the struct person. If the initial state is already set, then return false. If it isn’t, check if the character given is an ‘s’, ‘i’, ‘r’, or ‘v’. If so set the initial state as the character given, else return false. The **changestate,** if called is responsible for looking at the current state and changing it to the next state depending on what state it is currently at, the threshold and the infectious period. Finally, the **getcurrentstate**,when called, will return the current state of the struct person.

changestate()

We thought it would be best if the structure changes from ‘i’ to ‘r’ on its own so all that we would have to worry about if the susceptible struct person(‘s’) is next to an infected person or not. To allow them to keep up with changing themselves we have kept track of the days that they are infected internally and compare it to infectious period given by the input. The members that are responsible for these actions are –

* void changestate()
* void setinfectperiod(int period)

Within the **changestate** function, if the current state of the person is infected(‘i’) the person will change to state resistant(‘r’) if and only if the internal clock is equals to the infectious period given. If this isn’t the case, then the internal clock will increment and get ready for another **changestate** call. It is good to mention here that the internal clock will be incremented the moment the state changed from an ‘s’ to and ‘i'.

We’ve also found a way to keep track of turning a susceptible(‘s’) person into infected(‘i’) depending on the threshold. We did so by using the same method to turn the ‘i’ into an ‘r’, by keeping track how many infected people they are touching in a singular day. . The members that are responsible for these actions are –

* void changestate()
* void setthreshold(int thres)
* void markturned()
* void markUnturned()

The **changestate** is responsible for turning a susceptible(‘s’) person into infected(‘i’) person only if the person is marked enough to be turned. Within the struct it keeps count of how many times it is marked by incrementing the int threscount so long as the threscount is not greater than or equal the threshold given. Once the threscount is greater than or equals to the threshold give, check will be turned on. Once checked is turned on the susceptible person is allowed to become infected.

Apart from the struct we have several helper functions listed above in table format. We will not go on to explain the shorter functions such as the char menu(), string menu2() and bool exitprompt(), since these functions are simple menu functions. In the following I will explain about the functions below –

* bool fileReader(fstream&)
* vector<person> vfromString(string)
* void change()
* void display\_vec(int)

bool fileReader(fstream&) : The overall purpose of this function is to read in the first day( DAY 0 ) of the input file, which will be a 2D vector of person structs. It will then store the 2D vector into a 3D vector of 2Ds vectors of person structs. The 3D vector will essentially hold all the days made by the void change() function and will be used later in the void display\_vec(int) function.

In order to make the 2D vector this function will get each line of the chart, erase the commas, and it will then send the line without the commas in to the vector<person> vfromString(string) function, which will turn each line into a 1D vector of person structs. You can think of this as the row. Each row will then get pushed back into the 2D vector to create the initial chart. Once this process is done, DAY 0 should be created and store for further manipulation.

vector<person> vfromString(string) : The overall purpose of this function is to take in a string; this string should be the rows of the chart that was given, without commas, and a 1D vector creating n amount of struct person using each individual char in that string. Apart from setting initial state of the person, the function also sets the infectious period and the threshold. It will then return the 1D vector.

void change() : This function takes in the threshold and the infectious period of the model

and apply it to how the state will change. It will do this until all of the days are made and stored into a 3D vector. It will continue to go through a do-while while the count of infectious(‘i’) individuals are still on the graph. First the do-while loop will make a 2D vector from the last day of the 3D vector. This will ensure that every time we change the graph it will be the last changed graph. If the current state is susceptible(‘s’) then check if [i][g] where if g equals 0 then –

* Check left first column
* Check right of first column
* Check left not first column

If g equals to last column then –

* Check right of last column
* Check right not last column

We will also check –

* Up
* Down
* Check bottom right corner if [0][0]
* Check upper left corner if [9][9]
* Check bottom left corner if [0][9]
* Check upper right corner if [9][0]
* MIDDLE OF GRAPH

If the susceptible person is next to an infected person the markturned() function will be called to see if they are allowed to turn depending on the threshold. Every time a day passed the infectious person’s internal clock will increment to allow them to change their state depending on the infectious period. When all is done to the graph it will get pushed into the 3D vector.

void display\_vec(int) : The overall purpose of this function is to take in a string, the string is the display condition given. This is allow the function to know when and what day you would like to view. The for-loop will go through each display amount of the days requested. The function will keep count of the day that has the most infected persons, and display what that day was at the end of the model. It will also display the –

* Resistant
* Susceptible
* Infected
* Vaccinated

Once the model have finish the 3D vector is set to clear and the conditions vector is set to clear. This is so we can run another model if the user wishes.

|  |  |
| --- | --- |
| Responsibilities | |
| Tuyet Pham | Burton Fox |
| Change() | Design Document 60% |
| Design Document 40% | Main() |
| display\_vec() | exitprompt |
| fileReader | menu2 |
| vfromString | menu |
| Struct person |  |