UML

Provided as a separate document

Intention of software.

This software intends to model how disease will spread in a population when differing percentages of the adult and child population are vaccinated (or are immune through other channels). It creates families of 2 adults and 2 children (one family contains an infected adult).

People can either be susceptible to the disease (status: 0), immune (status: 1) or infectious (status: 2). People are either susceptible or immune, depending on whether they have been vaccinated. The percentage of adults and children that have been vaccinated can be changed by the user. If people come into contact with an infected person (are a certain distance from them), there is a chance they will catch the disease. If someone catches the disease they are infectious for a certain amount of time before becoming immune. Every day, people come home and mix with their families (mixing between ages) and then go to work/nursery (mix with more people of the same age). This process repeats itself until no one infectious is left. The model records show many people are infectious each day and prints a graph which shows how the disease is progressing.

Thought process (pseudocode)

1. Create a person class. Create child and infected classes of people which inherit all the features of the Human class.
2. Give each class a different infection status property (i.e. if they start of immune to the virus or not), but keep the other properties the same. Give them random starting positions.
3. Create functions (within the class) for people to move about their environment.
4. Put people into lists of family units and work units (that is, people that are in the same family or people that work together). People can be in more than one list.
5. Create an infection class. In this class make functions to see if people are in contact with an infected person, move people around the environment.
6. In the people classes, get the infection status to change depending on whether they are in contact with an infected person in the infection class.
7. Make it so that once people are infected for a certain amount of time they stop being infectious.
8. Put the list of people into the infection class. Alternate between the home and work lists. Continue doing this until no-one is infected anymore.
9. Create images of the people in each list moving around their environment and infecting one another. Assign colours to the people depending on their infection status and get these to change as they become infected or are no longer infectious.
10. Create a graph to show how the infections spread through time.
11. Write the results of the number of people who are infected each day of the model to a csv file.
12. Get the program working in a GUI.
13. Figure out how to show an image (the graph) in the GUI
14. Try and get the user to enter into the GUI the percentage of adults vaccinated, and try and get the model to run from the GUI, and then reprint the graph.

Software development process (tests)

Create Human, Child and Person0 classes of people. Check that the functions are working properly by printing them out. This involves ensuring that the \_\_init\_\_ values are correct and when random generators are used to create them, that the random number is different for each person.

Write into the if, elif, statements an else clause that prints out that something went wrong. This means I know when I have forgotten to cover an eventuality.

Put try, except statements to catch errors but still run the program and print out messages when this happens so I know something has gone wrong.

Put in checks so that the user can’t enter a string or an out of range number for the percentage adults vaccinated.

Stress test the model by entering a series of invalid parameters, run the model with missing variables to see if it will still work.

Print out functions in the Infection class to see if this is working properly. This involves printing out the distance between people, whether they are in contact with an infected person. It then involves checking the values of location (which should have updated) and the infection status in the people classes to check that they have updated properly. From this, I found out that the moving the x and y co-ordinates was not working properly and changed this so it was.

I also print out graphs showing how the people move around their environment and become infected to check this is working and see how the infection status of people changes as the model progresses to check that people are being infected properly.

Put in different values of adults and children that are vaccinated to check the model works properly. Set the number of adults vaccinated to 100%, this means that the disease should only spread round one nursery and this was the case when running the model.

Put in other values of children and adults infected and check that the disease is spreading around the workplaces and nurseries. Print out the results of this in a csv file. Look at the results and check that it looks like the disease is moving between places in a plausible manner.

Print a graph to show the progression of the disease through time. Look for peaks and troughs in this, as this is how I would expect a disease to progress.

How to run the code.

The files provided that are required to run this program are

1. disease\_module.py

This contains a Human, Person0 and Child classes required to create people with the required properties. It also contains an Infection class which determines if people are near another infected person and if they are, it then runs the function on the people classes to update their infection status accordingly.

1. disease\_model.py

This creates people using the classes in the disease\_module. It places people into lists of families, workplaces, and nurseries. It then allows people to interact with other people during the day, spreading disease between them until no-one infectious is left.

Instructions:

1) Open up disease\_model.py and disease\_module.py in spyder,

2) Run disease\_module and then disease\_model.py

3) Enter the number of adults and children you want to be vaccinated (number between 0 and 100)

4) Wait for this to finish running (it can take a while). When it has finished, “finished running” will appear in the console window and a GUI will appear. The GUI will show a graph of how the disease has spread through time.

Issues during development

Adding in an infected person into the model and having a different infection status for children and adults was a problem. I solved this by putting the infected person, adults and children into their own classes.

There was a lag between the percentage vaccinated entered and that percentage showing on the graph in the GUI. This was because the place to enter the value was in the GUI itself, which has already run before asking for the number (as I couldn’t get the GUI to re-run models or reprint values).

Managing to get people in the model to go to different places during the day and night, and for them not to all go to the same place. I solved this by putting people into lists and creating a separate infection class which took in these lists (of agents) which determined if people were in contact with an infected person and allowed this to update the infection status on the classes.

Producing an interactive GUI. I couldn’t get this to work.

References

Most of my knowledge in programming comes from

Dawson, M. 2010. *Python programming third edition for the absolute beginner.* Third Edition. United States of America: Cengage learning.

Lecture material of the core programming module.

Python documentation: Python. 2017. Python 3.6.4 documentation.[Online]. [Accessed 23 December 2017]. Available from: https://docs.python.org/3/

Information on how to create a UML diagram comes from Info from https://www.ibm.com/developerworks/rational/library/3101.html