CSCI 1300 Introduction to Computer Programming
Instructor: Hoenigman
Project 2: Outbreak!
Part 1 Due: Friday, April 18 by 11:59 pm.
Part 2 Due: Wednesday, April 30 by noon.

Simulating the Zombie Apocalypse


Photo by Center for Disease Control

Epidemiology is a field that studies how illness spreads through a population. The illness could range from the common seasonal cold to potentially lethal epidemics, such as SARS. Every year the Center for Disease Control offers predictions about the cold and flu season, and makes recommendations about flu vaccines for the general population to improve public health.

The spread of disease through a population is a complex process, and understanding how diseases spread often requires computer simulation. One model that is often used in disease simulations is the SIR model, which assumes that, at any time, there is some number of Susceptible (S) people, some number of Infected (I) people, and some number of Recovered (R) people. Susceptible people are those at risk of contracting the disease. Infected people are those with the disease. Recovered people are those who have had the disease and are now recovered. People stay in Infected and Recovered states for some amount of time, and go between SIR states when they go from Susceptible to Infected, Infected to Recovered, and Recovered to Susceptible.

In this project, you're going to implement an SIR model, where the disease being spread is Zombie-ism. The project is divided into two parts, and each part will be 50% of the grade. The first part involves understanding and documenting some code that I have provided that includes the base functionality needed for the simulation. The second part will be to change that code, or design your own, to implement the features needed for the SIR model.

The code you will need to start is on Moodle in a zip file called Project2ZombieApocalypse.zip. This code features a simulation of Turtles that can be either zombies or non-zombies in a population. Both types move around randomly in the universe and if a non-zombie turtle gets too close to a zombie turtle, the non-zombie

becomes a zombie. Over time, zombie-ism spreads through the population until all turtles are infected. In this code, there is no recovery! That is functionality that you will implement in Part 2.

The key features to this code are:

- There is a Universe that the Turtles move around in and interact with other Turtles.
- The position of the Turtles can be retrieved at any time step and used to determine which Turtles are in spatial proximity of each other.
- The state of the Turtles, non-zombie or zombie, can be retrieved at any time step, which, along with spatial position, is used to turn regular Turtles into zombie Turtles.
- There is a visual display of the simulation that draws the position and state of each Turtle.

.

Part 1: (Due: Friday, April 18 by 11:59pm)

Download the Project2ZombieApocalypse.zip file and verify that it runs on your machine. It should launch a window with red and blue dots moving randomly in the window. Examine the code, starting in the ZombieApocalypse.java file, and answer the following questions. Your answers should be typed and single-spaced. All answers together should add up to be about two pages in length.

Two things that you can do to help you figure out what this code is doing:

- Comment out the code in question and see what happens
- Change the code and see what happens
- Test your ideas by changing code to see if the result is what you predict it will be

QUESTIONS:

1. What is the purpose of:

   ```
   Universe un = new Universe(zTurtles, z, 600, 600);
   ```

   in the ZombieApocalypse.java file? What do the specific numbers represent here, e.g How many of the Turtles are actually zombies?

2. What is the purpose of the for loop in this same file, and if I wanted to run the simulation for 1000 iterations, which variable would I change?
3. Describe what happens in the Universe constructor. Particularly, how are locations assigned to each *Turtle*, what is the purpose of the *tLocations* variable, and how are some *Turtle* infected with zombie-ism?
4. How does *moveZombies* work? Particularly, what is the purpose of

   ```
   newX = min + (int)(Math.random() * ((max - min)+1));
   if ((oldX+newX < 0) || (oldX+newX >= width))
        newX = 0;
   ```

and

```
tLocations[oldX][oldY] = -1;
tLocations[oldX+newX][oldY+newY] = x;
```

(Note: it could be helpful to work out the first part of this question with a paper and pencil. Plug in different values for the random variable and oldX and worth through the calculation.)

5. What is the purpose of the for loop in *moveZombies*?
6. There are three for loops in *zombieAttack*. What is the purpose of each one of them?
7. How is the radius variable used in *zombieAttack*?
8. How are Turtles infected with zombie-ism in *zombieAttack*? What method is called and what variable is updated?
9. What is the purpose of the code:

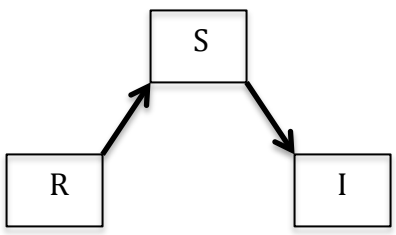
   ```
   turtles[tLocations[x2][y2]]
   ```

   in *zombieAttack*. You will probably need to look at `tLocations[x2][y2]` and how values are set in that array to answer this question.

10. Where in the code is the visual display of the Turtles created? The infected Turtles are red and uninfected Turtles are blue. In what method is this determined?

Part 2 (Due Wednesday, April 30 at noon, before class.)
Either starting from the code provided, or from code you create, you need to add the following functionality to the code downloaded from Moodle.

1. The Turtles need to have three states, Susceptible, Infected, or Recovered and have to be assigned to one of those three states at all times.
2. If a Susceptible Turtle comes within a pre-defined radius of an Infected Turtle, then the Susceptible Turtle has an X% chance of becoming Infected. Currently, the infection rate is 100% in the code provided. This needs to be changed to have X set as a command-line argument in your program.
3. Turtles need to be able to change state from I to R and from R to S after they have been in their current state for a pre-defined number of days. This is the days to recovery and days to susceptibility, respectively. Your code needs to check how many days the Turtle has been in the current state and what that current state is, and reset both the time in current state and the state accordingly. The possible transitions between states is shown here:



(You will need additional instance variables to implement this functionality.)

4. When the Universe is created, only one Turtle should be Infected. All others should be Susceptible.
5. Your simulation will need a way to determine what day it is. You can assume that the loop in ZombieApocalypse.java can provide this, where an iteration of the loop represents one day. You will need to make this information available to your Universe and your Turtles.
6. Your simulation will need to have different colors or markers for Turtles in different states, i.e. Infected Turtles need to look different than Susceptible Turtles.
7. The infection rate, days to recovery, days to susceptibility, and radius of infection all need to be command line arguments.
8. Experiment with different values for infection rate, days to recovery, days to susceptibility, and radius. Try 7 days to recovery, 14 days to susceptibility, and infection rate of 40%. There was an example in a recitation of simulating a coin flip that could be helpful for simulating a % infection rate.
9. You are welcome to experiment with any other changes to this simulation that you would like to implement, including graphical displays, equations, movement, having Infected Turtles attack, adding mortality, etc.