Name: Qi Li **Email**: Qi5@kth.se
Name: Hussam Hassanein **Email**: Hussamh@kth.se

Simulation of how an infection spreads in a population

Table of Content

Problem description

What we need to do is to figure out how the infection spread in a population. Our goal is to learn the process of infection and try to create a reasonable mathematical model and system, that will make us capable of simulating the behaviour of this infection, what this infection can result to and how much it can spread, based on the input data that we have. Base on these scientific fact we need to define in what is the best way to simulate the the infection, how should we make sure all the result are verified and validated during the developed phase and data collection phase and why is the simulation we made is accurate and precise.

The pros and cons of the model

The system we have built gives a representation of how an infection simulator could look like, when we are able to control some of the input values in the process. The system simulates what happens during the infection process. The process occurs in several days until the infection ends.

Pros

Modeling, when dealing with an important case such as infection, is really important to have some sort of control on the process. A model works only under the same circumstances, so we can’t just generalize our model to examples that don’t fulfill our requirements for the model to work. That means that it is important to take care of the details in such a model and keep track of the change of requirements during the process.
One of the reasons to use modeling is to try to predict the behaviour of the elements of the model and in our case it is the infection. If this infection is about to spread, we need to compare it to a similar existing model and analyze to try and understand how it could spread and therefore decrease the damage resulted by this infection.

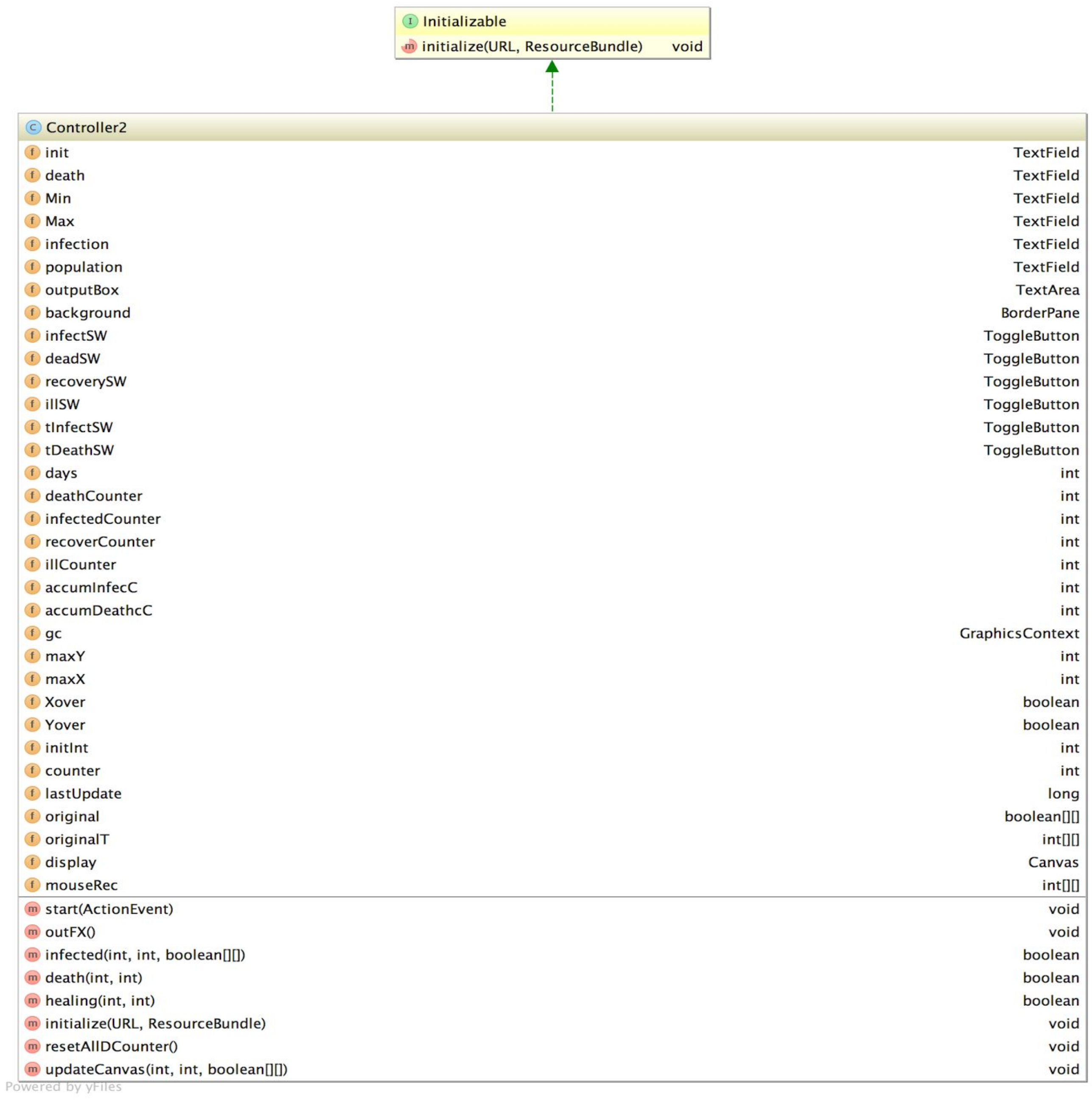
Cons

The system doesn’t fully give a fair representation of the problem if it is considered to be used in real life situation. Some assumptions could be unrealistic sometimes, because every case differs than the other. And every disease has their own behaviour and requirements regarding that.
Also, any change in the proposed requirement to the experiment will make it invalid. The reason is, a model is set to be true/valid under certain circumstances and any change in that will mean that, you can get a different behaviour.

Solution description

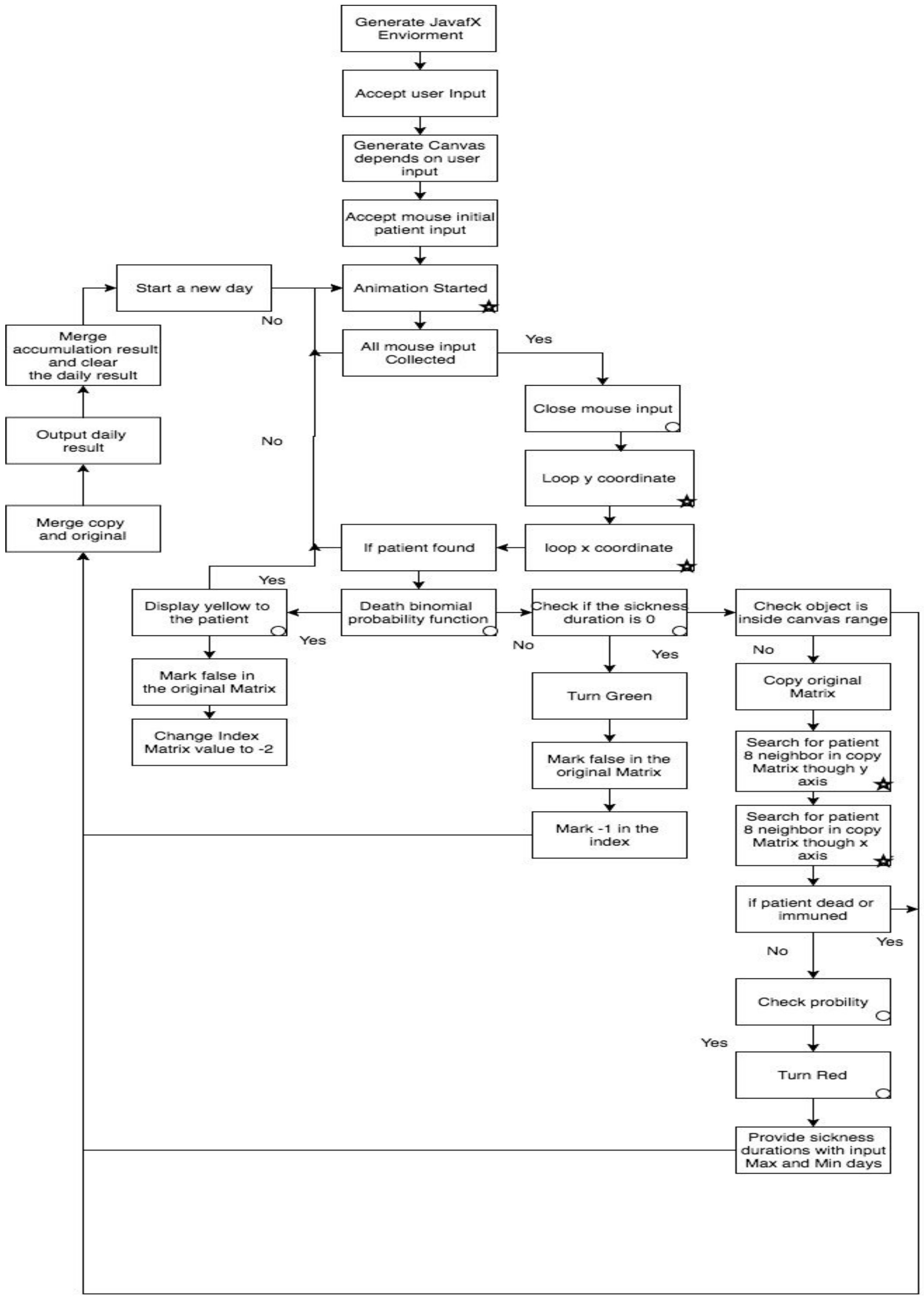
Diagrams

UML diagram



Flow chart

Both Diagram can be found under AFS/HOME/q/i/qi5/untitled2/Diagram and AFS/HOME/h/u/untitled2/Diagram folder

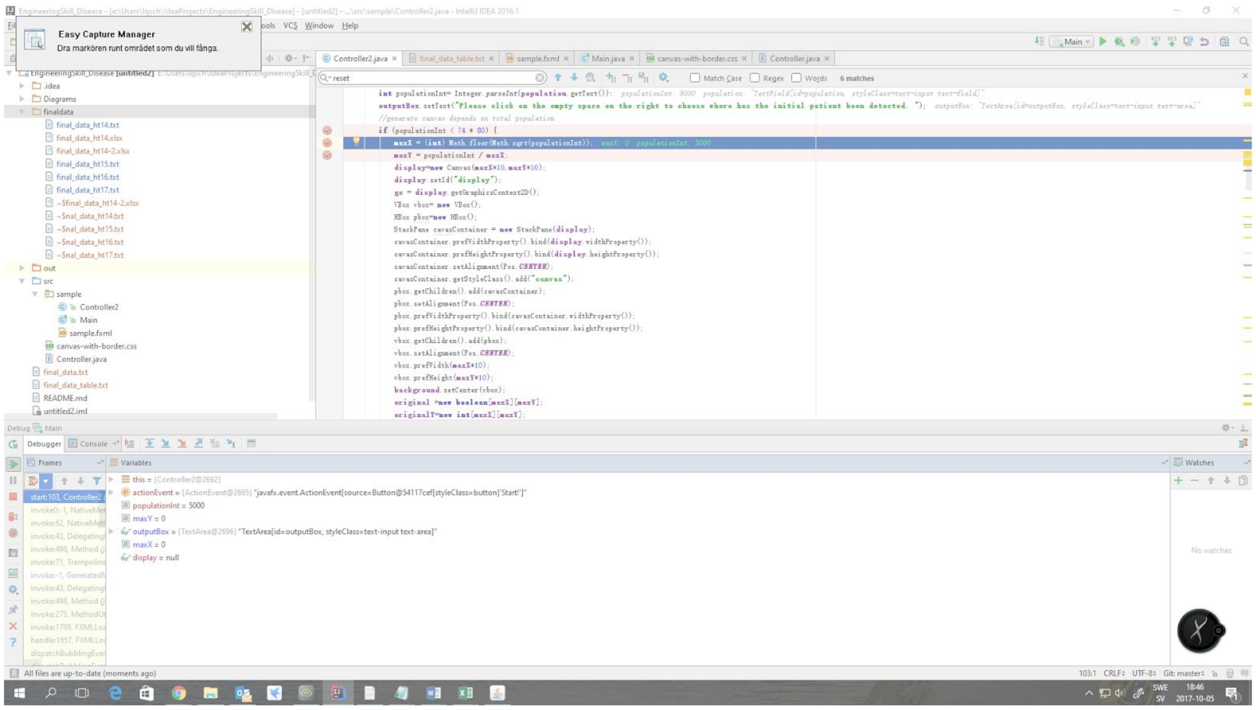


Generate JavafX Enviorment
Accept user Input
Generate Canvas depends on user input
Accept mouse initial patient input
Animation Started
All mouse input Collected
Yes
Close mouse input
Loop y coordinate
loop x coordinate
If patient found
No
Start a new day
Merge accumulation result and clear the daily result
Output daily result
Merge copy and original
Yes
Display yellow to the patient
Mark false in the original Matrix
Change Index Matrix value to -2
Death binomial probability function
Yes
Check if the sickness duration is 0
No
Yes
Turn Green
Mark false in the original Matrix
Mark -1 in the index
Check object is inside canvas range
No
Yes
Copy original Matrix
Search for patient 8 neighbor in copy Matrix though y axis
Search for patient 8 neighbor in copy Matrix though x axis
if patient dead or immuned
Yes
No
Check probility
Yes
Turn Red
Provide sickness durations with input Max and Min days

for higher resolution file

How we approach

The program will first ask user for input value, and then detect if the initial input is correct. This has been done by an if statement with maximum of total 70*74 pixel of the whole canvas. After user has click the start button the program will start counting if the initial patient count has been reached. The output data has been separated into 2 categories, daily value and accumulated value. The simulation will be terminated when all the daily value has been set to 0 which mean no more action will happen during day after but this has exception due to it can be during that round all the probability count has goes to failed in the startup phase. In case this happend I have set if this circumstances has happened 10 times, the system will seem it as done. There will also be a complete message showed afterwards. Because the start up phase has max 3 to 4 infection. Max 10 failed is safe enough to judge the program has done or not. To be able to limit the population the 2 backend matrices and front end canvas should has certain boundary. If there is a spawn outside the boundary has been initiated, this spawn will be killed and dropped. This will happen before all the spawn process started which is during the coordinate calculation phase. To be able to make sure the first day one got infected and he will not infect somebody else, we have set one day as 1000ms and after each of the “box ”has been checked probability once per day the day will be terminated. The way to reach this goal is by using animation timer, timer will tick every 1000ms. So there is impossible for person who just got infect someone to infect someone else because his own binomial distribution function has not even active yet. The most of the validation for each function has been done by java own debug functionality.


```
int populationInt= Integer.parseInt(population.getText());   populationInt: 5000  population: "TextField[id=population, styleClass=text-input text-field]"
outputBox.setText("Please click on the empty space on the right to choose where has the initial patient been detected. ");   outputBox: "TextArea[id=outputBox, styleClass=text-input text-area]"
//generate canvas depends on total population
if (populationInt < 74 * 80) {
    maxX = (int) Math.floor(Math.sqrt(populationInt));   maxX: 0  populationInt: 5000
    maxY = populationInt / maxX;
    display=new Canvas(maxX*10,maxY*10);
    display.setId("display");
    gc = display.getGraphicsContext2D();
    VBox vbox= new VBox();
    HBox pbox=new HBox();
    StackPane cavasContainer = new StackPane(display);
    cavasContainer.prefWidthProperty().bind(display.widthProperty());
    cavasContainer.prefHeightProperty().bind(display.heightProperty());
    cavasContainer.setAlignment(Pos.CENTER);
    cavasContainer.getStyleClass().add("canvas");
    pbox.getChildren().add(cavasContainer);
    pbox.setAlignment(Pos.CENTER);
    pbox.prefWidthProperty().bind(cavasContainer.widthProperty());
    pbox.prefHeightProperty().bind(cavasContainer.heightProperty());
    vbox.getChildren().add(pbox);
    vbox.setAlignment(Pos.CENTER);
    vbox.prefWidth(maxX*10);
    vbox.prefHeight(maxY*10);
    background.setCenter(vbox);
    original =new boolean[maxX][maxY];
    originalT=new int[maxX][maxY];
```

```
this = {Controller2@2692}
actionEvent = {ActionEvent@2695} "javafx.event.ActionEvent[source=Button@54117cef[styleClass=button]'Start!']"
populationInt = 5000
maxY = 0
outputBox = {TextArea@2696} "TextArea[id=outputBox, styleClass=text-input text-area]"
maxX = 0
display = null
```

Java debug function

With setting end point at the LOC that i wanna debug. The program will run into the end point and start step into the function. This will show me the variable and process number of current step in function. For example, if i start input 10000 as population, but at population checking step it is telling me the if statement still providing a true, this will tell me that i did something wrong and the function is returning false result. At same time if function return false as answer, we know we have done it right now.

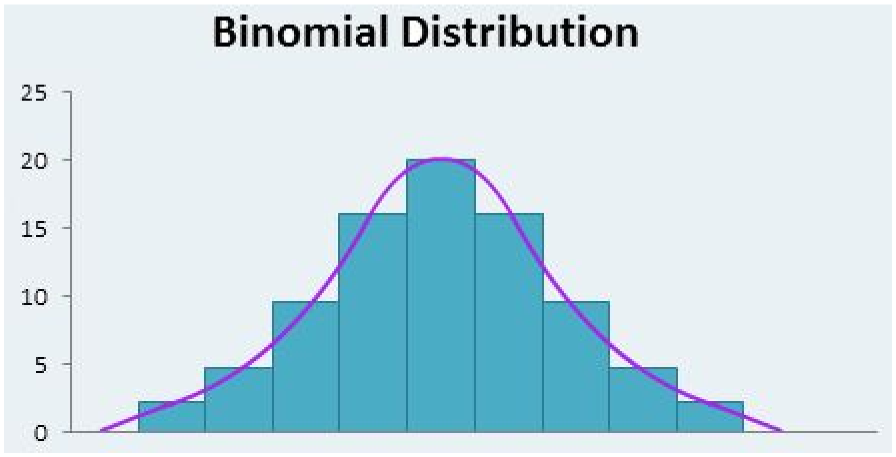
How the validation is planned

Test cases

- Test the output in normal conditions.
- Observe the change in results, given that there are random values involved in the process.
- Test the extreme values where the probabilities are either 0 or 100% and see how does that differ from any random value in between
- Test increasing or decreasing ratios and see how that reflects on the graph and to make sure if it gives stable results.
- Change one of the requirements and observe the results of that.
- Since in the input data there is not a specification of the infection rate. We decide to use constant 0.1 as infection probability for all of the simulation.
- Test each function independently (module testing).
- Test how they work together (integration testing).
- Try to combine each test result with common sense to see if it make sense
- Hand the software to some medical professional people and see how the test result goes.
- Consult disease specialist to see if the graphical and data representation make sense for them.
- Execute all test cases again each time a new functionality has been implemented, after some basic function development.

How the results are validated

- During the whole testing phases we only change 1 variable at a time and make other variable constant for more accurate test result.
- Each function result has been validation through java integrated debug function. We have step into each function and validate if the variable returning right answer and if the process number is correct. This has been done by input normal value and inputting extreme value.



Tutorvista.com. (2017). Normal Approximation to the Binomial Distribution | TutorVista.com. [online] Available at: http://www.tutorvista.com/content/math/normal-approximation-to-the-binomial-distribution/ [Accessed 5 Oct. 2017].

- We observe that there is a slight difference between the outputs under the same circumstances, because of the Disease spread is randomized. For precise and accurate simulation. We have decide to use mathematical probability theory during our simulation which in this case we have used binomial distribution. A single person can be sick or not is depends on the probability user input and system will determine the probability from probability diagram. If it falls inside the area under the probability, it's a hit otherwise this person survived.
- We have tried 2 extreme value in both probability input which are infection probability and death probability. The result fits the real life cases. During minimum of infection probability nobody got sick and during maximum probability everybody got sick. Same case goes for the death probability. Nobody is dying death probability is 0 and each infected people are dying dying death probability goes to maximum. These result have been showed from the GUI interface of simulation process. We have set grey as isn't involved , yellow as dead, red as infected and green as immuned/recovered. GUI can help us to verify each cases in a more visualized and obvious way. For example during the maximum infection probability testing phase we can see that every person turned red in the end with few person dying and nobody is in grey which is survived the disease.
- The test case for each function is built on the fact that the result of each step make sense in a theoretical, mathematical and human sense point of view.
- The test for integration include, input all the possible cases include all the extreme possible we can thought of.
- We have handed the software to one of the Karolinska student and observe her reaction. She has outlined a few problems including the data inaccuracy and the spread speed. In general, the graphical interface and data made sense to her. She has also told us that “ it make sense that when it getting closer and closer to the edge of the canvas, it means the less people the disease can infect and more people are either immune or dead so less red spot (infection) we can see on the graph. She also said that it make sense that it goes to one side of the canvas first because those people are somehow in connection. Afterwards it spread to other side of the canvas. This is also why disease controller

always start quarantine certain group of people that has connection with initial infection patient instead just quarantine himself. Because this can effectively cut down the disease spread to the other side of the population.

- After a basic function has been implemented, we have decide to implement some extra features, to be able to benefit the future data collection, and eliminate a few human mistakes. After each of these functions has been implemented, we run all the test case all the way from beginning with the help of java debug tool.

Results and conclusions

During the simulation around 4 different input values, we realize that the bigger the population and sickness duration is, the longer time the disease spreads. The sickness duration has also an effect on the recovery time. The green spot on the window is clearly getting less or is taking a longer time to show up, when we increase the duration time. This can also happen when we rise the minimum sickness days. One exception do happen during the ht17 since result of the sickness duration is getting lower but the total days of disease spread is getting higher. The reason of that is unknown. We have tried several simulation and parently that it is always the same. The threshold of the infection can exist and this is depends on the infection probability,sickness duration and death probability. When a patient is dead, they will no longer be able to infect other people. This state is basically same as immuned but dead. This also told us that sickness duration has to do something with this too.The short the duration the more possible can be survive before infect other people. We have tried to put infection probability as a certain number and keep raising death probability or shorten sickness duration. The result is quite obvious. The most extreme circumstances is that there are only 1 or 2 spawned and dead or immuned right away before he infect anybody else. These circumstances are completely independent to population too. This remind me something about what nazies do during world war 2 when they have killed too much and those dead body start spreading disease. They either kill them or if some of them are rich they priority cure them with expensive vaccine. This can effectively shorten the threshold of the infection and avoid epidemic happens. This is sometime quite annoying for my simulations during developer phase too. Because sometimes this seems like it is crashing and program only spawn one or two patients before it is dead. The way that we find to solve this issue is to use the debug tool by java and step in the function to see if it is doing the right thing or not. If this situation happened, i define it as extreme circumstances and i put them in exception during developing, result collection and result validation phase.

How to compile and run/use the program

To be able to compile the program, it is required to have to Official Oracle Java JDK 1.8.0 installed with JavaFX plug in.OpenJDK will not work unless use build OpenJFX on the server. For how to do it please visit this website-https://wiki.openjdk.java.net/display/OpenJFX/Building+OpenJFX.
Here is the quote from the previous website link.

Windows

You need to have the following tools installed:

- Cygwin. Some packages to make sure are installed are:
  - openssh
  - bison
  - flex
  - g++
  - gperf
  - make
  - makedepend
  - mercurial
  - perl
  - zip
  - unzip
- DirectX SDK June 2010. Microsoft DirectX SDK (June 2010) headers are required for building the JavaFX SDK. This DirectX SDK can be downloaded from Microsoft DirectX SDK (June 2010). If the link above becomes obsolete, the SDK can be found from the Microsoft Download Site (search with "DirectX SDK June 2010"). The location of this SDK will normally be set with the environment variable DXSDK_DIR at installation time. The default location is normally "C:/Program Files/Microsoft DirectX SDK (June 2010)/". If DXSDK_DIR is not set, the build process may look for it in the default location or "C:/DXSDK/".
- Microsoft Visual Studio 10 SP1 (express edition works). The compiler and other tools are expected to reside in the location defined by the variableVS100COMNTOOLS which is set by the Microsoft Visual Studio installer.

Mac

To configure your Mac, make sure you have at least version 10.7 installed. Install the latest version of Xcode and that you have the developer tools installed. You can install them by using the menus within Xcode: XCode -> Preferences -> Downloads -> Components. Install the latest JDK 8 build. In order to build WebKit, you will also need to install QT 5.2 (because WebKit uses QMake).

IMPORTANT: If you have a different version of X code (say one that is compatible with OS X 10.9), you will need to add the following line to your ~/.gradle/gradle.properties file:

```
MACOSX_MIN_VERSION=10.9
```

Depending on the version of X code that you have, the value of MACOSX_MIN_VERSION may need to be different (ie. 10.8). If you do not set this variable correctly, the C code will not build.

Linux

Setting up a Linux build configuration is fairly straightforward. These build instructions were used for the "official" build platform of Ubuntu 10.04, but also on the latest Ubuntu 12.10. First, run the following command to install all the required development packages:

Ubuntu 14.04, 15.10, 16

```
sudo apt-get update
sudo apt-get install ksh bison flex gperf libasound2-dev libgl1-mesa-dev \
    libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev libjpeg-dev \
    libpng-dev libx11-dev libxml2-dev libxslt1-dev libxt-dev \
    libxxf86vm-dev pkg-config x11proto-core-dev \
    x11proto-xf86vidmode-dev libavcodec-dev mercurial \

   libgtk2.0-dev libgtk-3-dev \
    libxtst-dev libudev-dev libavformat-dev
```

You will also need to install QT 5.2 in order to build WebKit (because WebKit uses Qmake). With Ubuntu 16, this will satisfy the requirements:

```
sudo apt-get install cmake ruby
```

Ubuntu 14.10

currently not recommended for building ARM because of packaging conflicts with libgl1-mesa-dev and the compatibility libraries needed for ARM.

Oracle Enterprise Linux 7 and Fedora 21

```
yum install mercurial bison flex gperf ksh pkgconfig \

            libpng12-devel libjpeg-devel libxml2-devel \

   libxslt-devel systemd-devel glib2-devel  gtk2-devel \

   libXtst-devel pango-devel freetype-devel
```

Linux ARM

Building OpenJFX for Linux ARM has only been tested on as a cross build from Linux and MacOSX. The process is only regularly used on Linux. Follow the steps for a Linux build setup first, and then refer to the steps for Cross Building for ARM Hard Float.
[1]

User should be able to compile this program in all IDE but Intellij is strongly recommended because all the setting has been preset well by the Intellij.

[1] Anon, (2017). [online] Available at: https://wiki.openjdk.java.net/display/OpenJFX/Building+OpenJFX. [Accessed 5 Oct. 2017].

Input the initial Sickness population, Probability of infection , minimum and maximum sickness days, the probability of death and how many initial sickness people have found before simulation. Than press the start button in the end of the window. Click inside the black boundary where you want the initial sickness people spawn. The software will automatically start simulation after the total amount of initial sickness people has successfully spawned. The final data can be found at final_data.txt file. The file has been set a good formating and record the result per day. Daily results can also been found in the output window at the left cornor.

Location of the source code

KTH AFS:

The 2 file here are the same, but belong to the each team member.

Qi Li

\\AFS\kth.se\home\q\i\qi5\untitled2

Hussam hassanein

\\AFS\kth.se\home\h\u\hussamh\untitled2

Git

SSH:git@github.com:GiantPanda0090/EngineeringSkill_Disease.git

Appendices

Requirements Analysis

The general requirement is to create a system that simulates the process of the spread of an infection in a population.
We build a system to help solving the problem, according to some requirements, which will be specified in the following table.

Description of requirement	**What is required to satisfy the requirement**	**To what degree the requirement is fulfilled**
The population is modeled as an NxN matrix	Population is represented in a table with equal number of rows and columns	Completely fulfilled
The disease to be simulated is spread by direct contact between individuals	The infection process is to occur only when there is a direct contact between individuals, meaning that you can only infect your neighbors.	Completely fulfilled
An individual is ill for a random number of days during which time there is a probability that the individual infects each of its direct neighbors (normally 8) in the population	The duration of infection is randomized. There is a probability of infection to 8 neighbors in 3x3 grid.	Completely fulfilled
Every day an individual is ill, there is also a probability that the individual dies.	There should a probability assumed for the ill individual to be dead, which gets updated everyday.	Completely fulfilled
An individual who dies is not contagious	If an individual dies, they can't infect others anymore.	Completely fulfilled
A sick person cannot be infected	If an individual has been infected they can't be infected again. In other words you either recover or die.	Completely fulfilled

An individual who has been ill but recovered, is immune and cannot be infected again.	If an individual is recovered they can’t change their state at all. They will stay recovered for ever.	Completely fulfilled
Some input values are to be decided by the user	An input is to be required in the start of the process	Completely fulfilled

Summary

We have created a system, that simulates how an infection spreads in population. The system had to follow the requirements set by the client, which outlined the basics of this project. On the other hand, the way to solve the problem has been improving during the whole process of building the system. Building the system from scratch was a challenge in the start, because of us using modelling, which provides a different perspective to the problem. The validation and verification stage has helped a lot in deepen the understanding of the problem and made sure that everything works as intended. In general, The problem we have handled could have a lot more depth into it in real life, meaning that there are a lot more attributes and requirements that the problem could be based on. The model we have built deals with infection problems, according to the requirements specified by the client.

Source code

JavaFX Main Class:

```
package sample;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Main extends Application {

   @Override
   public void start(Stage primaryStage) throws Exception{
       Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));
       primaryStage.setTitle("Hello World");
       Scene scene =new Scene(root, 1124, 845);
       scene.getStylesheets().add("canvas-with-border.css");
       primaryStage.setScene(scene);
       primaryStage.show();
   }


   public static void main(String[] args) {
       launch(args);
   }
}
```

JavaFX Controller Class:

```
package sample;

/*
Controller Class
*/

import javafx.animation.AnimationTimer;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.control.ToggleButton;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;

import java.io.PrintWriter;
import java.net.URL;
import java.util.Random;
import java.util.ResourceBundle;

/**
* Created by lqsch on 2017-10-01.
*/
public class Controller2 implements Initializable {
   @FXML
   public TextField init;

   @FXML
   public TextField death;

   @FXML
   public TextField Min;
   @FXML
   public TextField Max;

   @FXML
   public TextField infection;

   @FXML
   public TextField population;

   @FXML
   public TextArea outputBox;

   @FXML
   public BorderPane background;

   @FXML
   public ToggleButton infectSW;
   @FXML
   public ToggleButton deadSW;
   @FXML
   public ToggleButton recoverySW;
```

```
    @FXML
    public ToggleButton illSW;
    @FXML
    public ToggleButton tInfectSW;
    @FXML
    public ToggleButton tDeathSW;

    /* non fxml init*/
    PrintWriter writer;
    PrintWriter tableWriter;


    //output
    private int days;
    private int deathCounter;
    private int infectedCounter;
    private int recoverCounter;
    private int illCounter;
    private int accumInfecC;
    private int accumDeathcC;


    public GraphicsContext gc;
    public int maxY;
    public int maxX;
    public boolean Xover;
    public boolean Yover;

    public int initInt;
    public int counter;

    private long lastUpdate;
    boolean[][] original;
    int[][] originalT;
    Canvas display;
    int[][] mouseRec;
    public int stopcounter;
    @FXML
    public void start(ActionEvent actionEvent) {
        //initialization for start button
        int populationInt= Integer.parseInt(population.getText());
        outputBox.setText("Please click on the empty space on the right to choose where has the
initial patient been detected. ");
        //generate canvas depends on total population
        if (populationInt < 74 * 80) {
            maxX = (int) Math.floor(Math.sqrt(populationInt));
            maxY = populationInt / maxX;
            display=new Canvas(maxX*10,maxY*10);
            display.setId("display");
            gc = display.getGraphicsContext2D();
            VBox vbox= new VBox();
            HBox pbox=new HBox();
            StackPane cavasContainer = new StackPane(display);
            cavasContainer.prefWidthProperty().bind(display.widthProperty());
            cavasContainer.prefHeightProperty().bind(display.heightProperty());
            cavasContainer.setAlignment(Pos.CENTER);
            cavasContainer.getStyleClass().add("canvas");
            pbox.getChildren().add(cavasContainer);
            pbox.setAlignment(Pos.CENTER);
            pbox.prefWidthProperty().bind(cavasContainer.widthProperty());
            pbox.prefHeightProperty().bind(cavasContainer.heightProperty());
            vbox.getChildren().add(pbox);
            vbox.setAlignment(Pos.CENTER);
            vbox.prefWidth(maxX*10);
            vbox.prefHeight(maxY*10);
            background.setCenter(vbox);
            original =new boolean[maxX][maxY];
            originalT=new int[maxX][maxY];
        } else {
            outputBox.setText("Population size is too big. it should be less that 70*77 ");
            return;
        }
        int initInt= Integer.parseInt(init.getText());
        mouseRec= new int[initInt][2];
        //initialize people get sick and where are they
        final int[] xAxis = new int[1];
        final int[] yAxis = new int[1];
        int[][] record=new int[initInt][2];

        display.setOnMouseClicked(event -> {
             xAxis[0] =(int) event.getX()/10;
             yAxis[0] =(int) event.getY()/10;
             xAxis[0]=xAxis[0]*10;
             yAxis[0]=yAxis[0]*10;
            boolean[][] copy =original;
            infectedCounter++;
            updateCanvas(xAxis[0], yAxis[0],copy);
            original=copy;
            counter++;
        });
        //end of initialization
        //log input value into log file
        writer.print("Result for Testing value: ");
        writer.println("Total Population: "+ population.getText() +" Infected Probability: "+
infection.getText() + " Days of sickness duration: From "+ Min.getText()+" till "+
Max.getText() +" days.");
        writer.println("Death Probability: "+ death.getText()+" Initial Sickness Porpulation:
"+ init.getText());
        //start simulation
        AnimationTimer timer = new AnimationTimer() {
            @Override
            public void handle(long now) {
                display.setOnMouseEntered(event -> {
                    outputBox.setText("Mouse X: "+(int) event.getX()/10+"Mouse Y: "+(int)
event.getY()/10);
                });
                //if total number of initilized people are successfully been input and accepted
                if (counter>=initInt) {
                    display.setOnMouseClicked(null);
                    display.setOnMouseEntered(null);
                    //start a new day per second
                    if (now - lastUpdate >= 1000_000_000) {
                        resetAllDCounter();
                        for (int y = 0; y < maxY; y++) {
                        for (int x = 0; x < maxX; x++) {
                            if (original[x][y] == true) {

                                //death
                                if(death(x,y)){
                                    break;
                                }

                                //healing
                               if (healing(x,y)){
                                    break;
                               }
                                //probability go though neighbor cell(8 of them)
                                if (x > 0 + 1 && y > 0 + 1 && x < maxX - 1 && y < maxY - 1) {
                                    //8 neighbors
                                    boolean[][] copy = original;//copy
                                    for (int diffy = -1; diffy <= 1; diffy++) {
                                        int Ny = y + diffy;
                                        for (int diffx = -1; diffx <= 1; diffx++) {
                                            int Nx = x + diffx;
                                            if (originalT[Nx][Ny] >= 0) {
                                                //infected
                                                stopcounter=0;
                                                infected(Nx,Ny,copy);
                                            }
                                        }
                                    }
                                    original = copy;
                                }
                            }
                        }
                    }
                        lastUpdate = now ;
                        illCounter=illCounter+infectedCounter-deathCounter-recoverCounter;
                        accumInfecC=accumInfecC+infectedCounter;
                        accumDeathcC=accumDeathcC+deathCounter;
                        //output
                        outFX();
                        days++;

if(infectedCounter==0&&deathCounter==0&&recoverCounter==0&&illCounter==0) {
                            stopcounter++;//quickfix
                        }
                        if (stopcounter> 10){
                           writer.close();
                           tableWriter.close();
                           resetAllDCounter();
                           resetAllOth();
                           this.stop();
                           outputBox.setText(outputBox.getText()+"\n Simulation has done!!");

                        }
                }
                    }
                }


        };

            timer.start();

    }
    public void outFX(){
        outputBox.setText("Day "+ days+" - \n");
        tableWriter.println(" - ");
        tableWriter.print(" | ");
        tableWriter.println(days+" | ");

        if(infectSW.isSelected()){
            outputBox.setText(outputBox.getText()+infectedCounter+" people has been infected
today; \n");
            tableWriter.println(" - ");
            tableWriter.println(infectedCounter+" | ");
        }
        if(deadSW.isSelected()){
            outputBox.setText(outputBox.getText()+deathCounter+" people has dead today; \n");
            tableWriter.println(" - ");
            tableWriter.println(deathCounter+" | ");
        }
        if(recoverySW.isSelected()){
            outputBox.setText(outputBox.getText()+recoverCounter+" people has been recovered
today; \n");
            tableWriter.println(" - ");
            tableWriter.println(recoverCounter+" | ");

        }
        if(illSW.isSelected()){
            outputBox.setText(outputBox.getText()+illCounter+" people are still ill today; \n"
);
            tableWriter.println(" - ");
            tableWriter.println(illCounter+" | ");


        }
        if(tInfectSW.isSelected()){
            outputBox.setText(outputBox.getText()+accumInfecC+" total amount of people has
infected until today \n" );
            tableWriter.println(" - ");
            tableWriter.println(accumInfecC+" | ");

        }
        if(tDeathSW.isSelected()){
            outputBox.setText(outputBox.getText()+accumDeathcC+" total amount of people has
dead until today \n");
            tableWriter.println(" - ");
            tableWriter.println(accumDeathcC+" | ");

        }
        outputBox.setText(outputBox.getText()+"Red = Infected,Yello = Death, Green = Recovered
\n");
        //log output data
        writer.println(outputBox.getText());

writer.println("==============================================================================
========");
    }
    //infected method
    public boolean infected(int Nx,int Ny,boolean[][]copy){
        Random rand = new Random();
        double test = rand.nextDouble();
        if (test < Double.parseDouble(infection.getText())) {
            infectedCounter++;
            updateCanvas(Nx * 10, Ny * 10, copy);
            return true;
        }
        return false;
    }

    //death method
    public boolean death(int x,int y){
        //death
        Random dRand = new Random();
        double dTest = dRand.nextDouble();
        if (dTest < Double.parseDouble(death.getText())) {
            originalT[x][y] = -2;
            original[x][y] = false;
            deathCounter++;
            gc.setFill(Color.YELLOW);
            gc.fillRect(x * 10, y * 10, 10, 10);
            return true;
        }
        return false;
    }

    //healing method
    public boolean healing(int x, int y){
        originalT[x][y] = originalT[x][y] - 1;
        if (originalT[x][y] == 0) {
            original[x][y] = false;
            originalT[x][y] = -1;
            recoverCounter++;
            gc.setFill(Color.GREEN);
            gc.fillRect(x * 10, y * 10, 10, 10);
            //gc.clearRect(x* 10,y* 10,10,10);
            return true;
        }
        return false;
    }

    //before pressing the start button
    //initial state
    @Override
    public void initialize(URL location, ResourceBundle resources) {
        //geneneral initialization for overall application
        stopcounter=0;
        days=0;
        deathCounter=0;
        infectedCounter=0;
        recoverCounter=0;
        illCounter=0;
        accumInfecC=0;
        accumDeathcC=0;
        lastUpdate = 0;
        infectSW.setSelected(true);
        deadSW.setSelected(true);
        recoverySW.setSelected(true);
        illSW.setSelected(true);
        tInfectSW.setSelected(true);
        tDeathSW.setSelected(true);
        try {
            writer = new PrintWriter("final_data.txt", "UTF-8");
            tableWriter= new PrintWriter("final_data_table.txt", "UTF-8");
        }catch(Exception e ){
            System.err.println(e);
        }
        //default value
        population.setText("5000");
        infection.setText("0.5");
        Min.setText("2");
        Max.setText("4");
        death.setText("0.2");
        init.setText("1");
        //user mistake elimination
        outputBox.setText("Please input the initial value for each box. \n"+"For probability 
please use decimal number instead of percentage.\nFor example 0.75 instead of 75 percent. ");
    }
    public void resetAllOth(){
        original=null;
        originalT=null;
        mouseRec=null;
        maxX=0;
        maxY=0;
        lastUpdate=0;
        counter=0;
        stopcounter=0;
        accumDeathcC=0;
        accumInfecC=0;


    }
    //reset the counters for yesterday
    public void resetAllDCounter(){
        deathCounter=0;
        infectedCounter=0;
        recoverCounter=0;
        illCounter=0;


    }

    //gui update per change
    public void updateCanvas(int x,int y,boolean[][]copy){
            final int xReal = x;
            final int yReal = y;
            gc.setFill(Color.RED);
            gc.fillRect(xReal, yReal, 10, 10);
            copy[x / 10][y / 10] = true;
       int minT= Integer.parseInt(Min.getText());
       int maxT=Integer.parseInt(Max.getText());
       Random durationR= new Random();
        originalT[x/10][y/10]=durationR.nextInt(maxT-minT) + minT;

    }

}
```

JavaFX FXML xml file:

```
<?xml version="1.0" encoding="UTF-8"?>

<!--Authors lqsch Qi Li, Hussam Hassanein-->
<!--Front end InterfaceXML file-->
<!--Download JavaFX 8.0 plus SceneBuilder before edit it-->
<!--suggested to edit it in Intellij with SceneBuilder-->
<!--Right click the fxml file on the project list in the right and choose Open in Scene
Builder-->

<?import javafx.scene.canvas.*?>
<?import javafx.scene.shape.*?>
<?import javafx.geometry.*?>
<?import javafx.scene.text.*?>
<?import java.lang.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>

<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="845.0" prefWidth="1124.0" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller2">
  <left>
     <VBox prefHeight="845.0" prefWidth="321.0" BorderPane.alignment="CENTER">
        <children>
           <HBox prefHeight="77.0" prefWidth="321.0">
              <children>
                 <VBox prefHeight="75.0" prefWidth="151.0">
                    <children>
                       <Label prefHeight="24.0" prefWidth="133.0" text="Population Size">
                          <VBox.margin>
                             <Insets left="10.0" top="35.0" />
                          </VBox.margin>
                       </Label>
                    </children>
                    <HBox.margin>
                       <Insets />
                    </HBox.margin>
                 </VBox>
                 <VBox prefHeight="75.0" prefWidth="152.0">
                    <children>
                       <TextField fx:id="population" onAction="#start">
                          <VBox.margin>
                             <Insets left="3.0" top="30.0" />
                          </VBox.margin>
                       </TextField>
                    </children>
                 </VBox>
              </children>
           </HBox>
           <HBox layoutX="10.0" layoutY="10.0" prefHeight="100.0" prefWidth="200.0">
              <children>
                 <VBox prefHeight="100.0" prefWidth="152.0">
                    <children>
                       <Label prefHeight="16.0" prefWidth="142.0" text="Infection
Probability">
                          <VBox.margin>
                             <Insets left="10.0" top="35.0" />
                          </VBox.margin>
                       </Label>
                    </children>
                    <HBox.margin>
                       <Insets />
                    </HBox.margin>
                 </VBox>
                 <VBox prefHeight="100.0" prefWidth="149.0">
                    <children>
                       <TextField fx:id="infection" prefHeight="31.0" prefWidth="167.0">
                          <VBox.margin>
                             <Insets left="3.0" top="30.0" />
                          </VBox.margin>
                       </TextField>
                    </children>
                 </VBox>
              </children>
           </HBox>
           <HBox layoutX="10.0" layoutY="110.0" prefHeight="100.0" prefWidth="200.0">
              <children>
                 <VBox prefHeight="94.0" prefWidth="138.0">
                    <children>
                       <Label prefHeight="21.0" prefWidth="135.0" text="Sickness Duration">
                          <VBox.margin>
                             <Insets left="10.0" top="35.0" />
                          </VBox.margin>
                       </Label>
                    </children>
                    <HBox.margin>
                       <Insets />
                    </HBox.margin>
                 </VBox>
                 <VBox layoutX="10.0" layoutY="10.0" prefHeight="94.0" prefWidth="184.0">
                    <children>
                       <HBox prefHeight="54.0" prefWidth="184.0">
                          <children>
                             <Label prefHeight="16.0" prefWidth="142.0" text="  Min" />
                             <TextField fx:id="Min" prefHeight="31.0" prefWidth="167.0" />
                          </children>
                       </HBox>
                       <HBox prefHeight="41.0" prefWidth="57.0">
                          <children>
                             <Label prefHeight="16.0" prefWidth="142.0" text="  Max" />
                             <TextField fx:id="Max" prefHeight="31.0" prefWidth="167.0" />
                          </children>
                       </HBox>
                    </children>
                 </VBox>
              </children>
           </HBox>
           <HBox layoutX="10.0" layoutY="210.0" prefHeight="100.0" prefWidth="200.0">
              <children>
                 <VBox prefHeight="100.0" prefWidth="152.0">
                    <children>
                       <Label prefHeight="16.0" prefWidth="142.0" text="Death Probability">
                          <VBox.margin>
                             <Insets left="10.0" top="35.0" />
                          </VBox.margin>
                       </Label>
                    </children>
                    <HBox.margin>
                       <Insets />
                    </HBox.margin>
                 </VBox>
                 <VBox prefHeight="100.0" prefWidth="149.0">
                    <children>
                       <TextField fx:id="death" prefHeight="31.0" prefWidth="167.0">
                          <VBox.margin>
                             <Insets left="3.0" top="30.0" />
                          </VBox.margin>
                       </TextField>
                    </children>
                 </VBox>
              </children>
           </HBox>
           <HBox layoutX="10.0" layoutY="310.0" prefHeight="100.0" prefWidth="200.0">
              <children>
                 <VBox prefHeight="94.0" prefWidth="151.0">
                    <children>
                       <Label prefHeight="16.0" prefWidth="142.0" text="Initial Sickness">
                          <VBox.margin>
                             <Insets left="10.0" top="35.0" />
                          </VBox.margin>
                       </Label>
                    </children>
                    <HBox.margin>
                       <Insets />
                    </HBox.margin>
```

```
                    </VBox>
                    <VBox prefHeight="94.0" prefWidth="158.0">
                       <children>
                          <TextField fx:id="init" prefHeight="31.0" prefWidth="131.0">
                             <VBox.margin>
                                <Insets left="3.0" top="30.0" />
                             </VBox.margin>
                          </TextField>
                       </children>
                    </VBox>
                 </children>
              </HBox>
              <VBox prefHeight="249.0" prefWidth="321.0">
                 <children>
                    <HBox alignment="CENTER" prefHeight="29.0" prefWidth="321.0">
                       <children>
                          <Label text="Output Value" />
                       </children>
                    </HBox>
                    <HBox alignment="CENTER" prefHeight="225.0" prefWidth="321.0">
                       <children>
                          <TextArea fx:id="outputBox" prefHeight="223.0" prefWidth="324.0"
wrapText="true" />
                       </children>
                    </HBox>
                 </children>
              </VBox>
              <HBox alignment="CENTER" layoutX="10.0" layoutY="406.0" prefHeight="29.0"
prefWidth="321.0">
                 <children>
                    <Button mnemonicParsing="false" onAction="#start" text="Start!" />
                 </children>
              </HBox>
           </children>
           <BorderPane.margin>
              <Insets />
           </BorderPane.margin>
        </VBox>
     </left>
     <top>
        <HBox prefHeight="100.0" prefWidth="200.0" BorderPane.alignment="CENTER">
           <children>
              <HBox prefHeight="100.0" prefWidth="1126.0">
                 <children>
                    <HBox alignment="CENTER" prefHeight="51.0" prefWidth="385.0">
                       <children>
                          <Label prefHeight="48.0" prefWidth="384.0" text="Disease Effection
Simulator " textAlignment="CENTER">
                             <font>
                                <Font size="31.0" />
```

```
                           </font>
                        </Label>
                     </children>
                  </HBox>
                  <HBox alignment="BOTTOM_RIGHT" prefHeight="100.0" prefWidth="738.0">
                     <children>
                        <BorderPane prefHeight="100.0" prefWidth="749.0">
                           <bottom>
                              <HBox alignment="CENTER" prefHeight="24.0" prefWidth="738.0"
BorderPane.alignment="CENTER">
                                 <children>
                                    <HBox prefHeight="31.0" prefWidth="115.0">
                                       <children>
                                          <ToggleButton fx:id="infectSW"
mnemonicParsing="false" text="infected /day" />
                                       </children>
                                    </HBox>
                                    <HBox prefHeight="31.0" prefWidth="90.0">
                                       <children>
                                          <ToggleButton fx:id="deadSW" mnemonicParsing="false"
text="dead /day" />
                                       </children>
                                    </HBox>
                                    <HBox prefHeight="31.0" prefWidth="115.0">
                                       <children>
                                          <ToggleButton fx:id="recoverySW"
mnemonicParsing="false" text="recoverd/day" />
                                       </children>
                                    </HBox>
                                    <HBox prefHeight="31.0" prefWidth="73.0">
                                       <children>
                                          <ToggleButton fx:id="illSW" mnemonicParsing="false"
text="ill /day" />
                                       </children>
                                    </HBox>
                                    <HBox prefHeight="31.0" prefWidth="142.0">
                                       <children>
                                          <ToggleButton fx:id="tInfectSW"
mnemonicParsing="false" prefHeight="31.0" prefWidth="134.0" text="Total infection" />
                                       </children>
                                    </HBox>
                                    <HBox prefHeight="31.0" prefWidth="115.0">
                                       <children>
                                          <ToggleButton fx:id="tDeathSW"
mnemonicParsing="false" text="Total death" />
                                       </children>
                                    </HBox>
                                 </children>
                              </HBox>
                           </bottom>
```

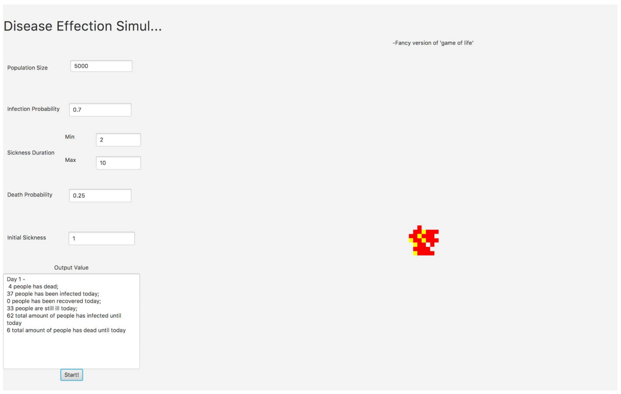
```
                          <center>
                             <HBox alignment="BOTTOM_CENTER" prefHeight="1.0"
prefWidth="738.0" BorderPane.alignment="CENTER">
                                <children>
                                   <Label alignment="CENTER" contentDisplay="CENTER"
prefHeight="24.0" prefWidth="169.0" text="Output switch" />
                                </children>
                             </HBox>
                          </center>
                       </BorderPane>
                    </children>
                 </HBox>
              </children>
           </HBox>
        </children>
     </HBox>
  </top>
  <center>
     <BorderPane fx:id="background" prefHeight="200.0" prefWidth="200.0"
BorderPane.alignment="CENTER" />
  </center>
</BorderPane>
```

Methods

- We start the experiment with one infected person and as we see from the following pictures the spread of the infection until there are no active infected people anymore; they are either dead, recovered or not involved at all. We assume here that there is a possibility of dying after being ill.

Colors representation

- ❏ Red is infected.
- ❏ Green is immuned.
- ❏ Yellow is dead.
- ❏ Grey(default) is not involved.

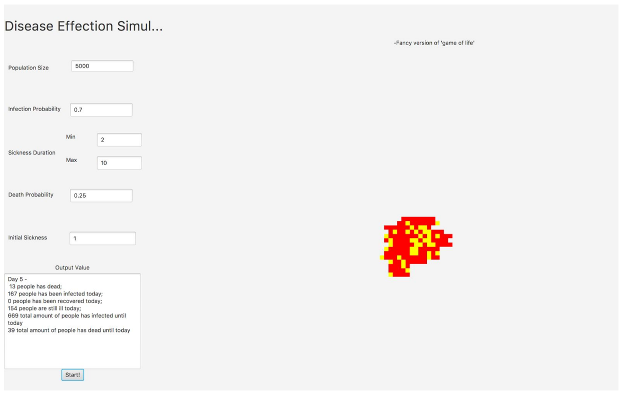

Disease Effection Simul...

-Fancy version of 'game of life'

Population Size: 5000
Infection Probability: 0.7
Sickness Duration: Min 2, Max 10
Death Probability: 0.25
Initial Sickness: 1

Output Value

```
Day 1 -
 4 people has dead;
37 people has been infected today;
0 people has been recovered today;
33 people are still ill today;
62 total amount of people has infected until
today
6 total amount of people has dead until today
```

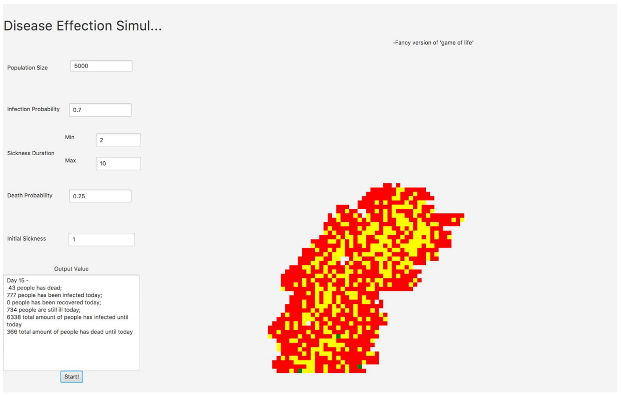

Disease Effection Simul...

-Fancy version of 'game of life'

Population Size: 5000
Infection Probability: 0.7
Sickness Duration: Min 2, Max 10
Death Probability: 0.25
Initial Sickness: 1

Output Value

```
Day 5 -
 13 people has dead;
167 people has been infected today;
0 people has been recovered today;
154 people are still ill today;
669 total amount of people has infected until
today
39 total amount of people has dead until today
```

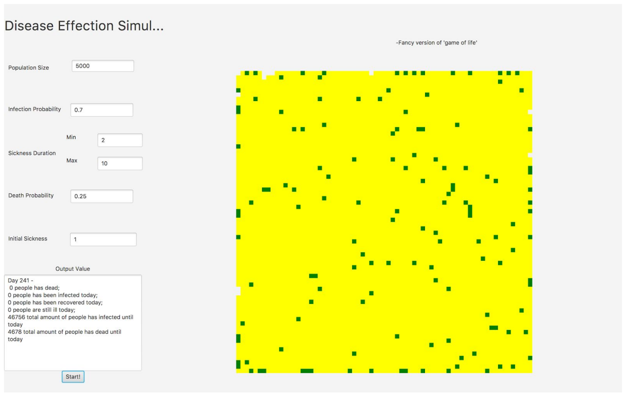

Disease Effection Simul...

-Fancy version of 'game of life'

Population Size: 5000
Infection Probability: 0.7
Sickness Duration: Min 2, Max 10
Death Probability: 0.25
Initial Sickness: 1

Output Value

```
Day 15 -
 43 people has dead;
777 people has been infected today;
0 people has been recovered today;
734 people are still ill today;
6338 total amount of people has infected until
today
366 total amount of people has dead until today
```


Disease Effection Simul...

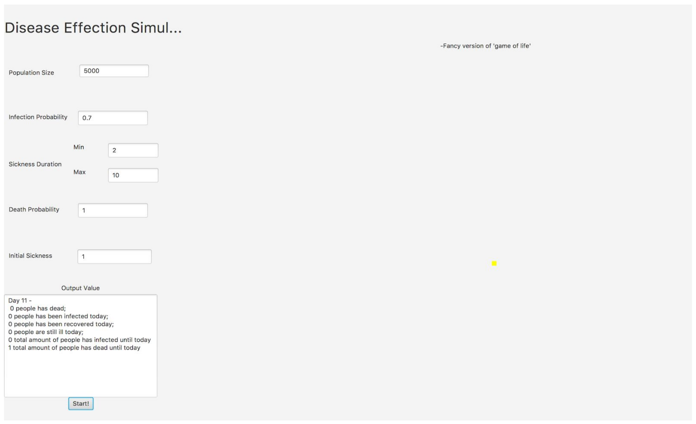
-Fancy version of 'game of life'

Population Size: 5000
Infection Probability: 0.7
Sickness Duration: Min 2, Max 10
Death Probability: 0.25
Initial Sickness: 1

Output Value

```
Day 241 -
 0 people has dead;
0 people has been infected today;
0 people has been recovered today;
0 people are still ill today;
46756 total amount of people has infected until
today
4678 total amount of people has dead until
today
```

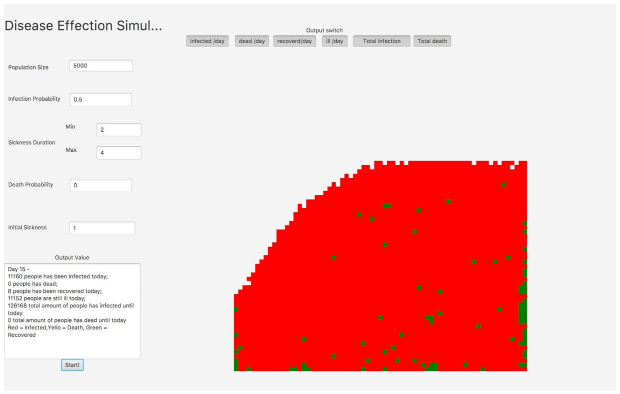
- If we choose the death probability to be 1 and we start with one infected person, then after each day we would still have one dead person.



Population Size 5000
Infection Probability 0.7
Sickness Duration Min 2 Max 10
Death Probability 1
Initial Sickness 1

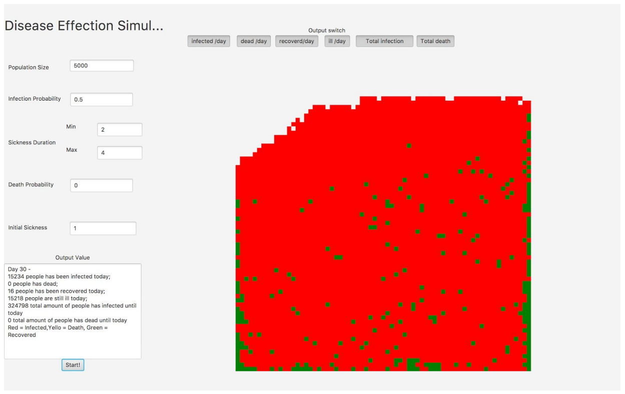
```
Day 11 -
 0 people has dead;
0 people has been infected today;
0 people has been recovered today;
0 people are still ill today;
0 total amount of people has infected until today
1 total amount of people has dead until today
```

- If we set the death probability when being ill to zero, we conclude that we either have people either recovered, infected or not affected yet. The recovered people don't get infected again.



Population Size 5000
Infection Probability 0.5
Sickness Duration Min 2 Max 4
Death Probability 0
Initial Sickness 1

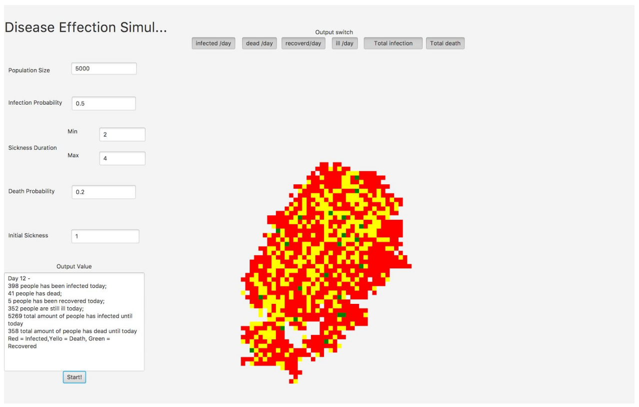
```
Day 15 -
11160 people has been infected today;
0 people has dead;
8 people has been recovered today;
11152 people are still ill today;
126168 total amount of people has infected until
today
0 total amount of people has dead until today
Red = Infected,Yello = Death, Green =
Recovered
```



Population Size 5000
Infection Probability 0.5
Sickness Duration Min 2 Max 4
Death Probability 0
Initial Sickness 1

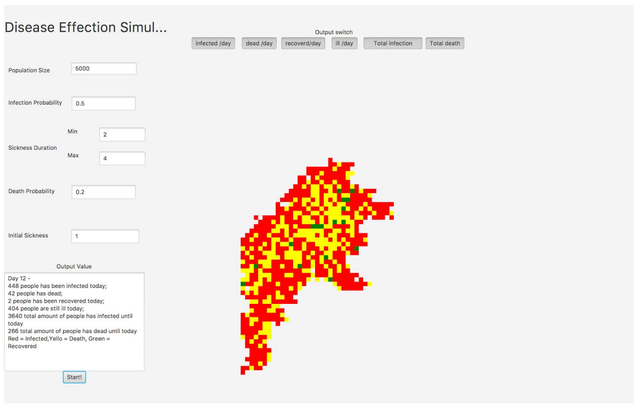
```
Day 30 -
15234 people has been infected today;
0 people has dead;
16 people has been recovered today;
15218 people are still ill today;
324798 total amount of people has infected until
today
0 total amount of people has dead until today
Red = Infected,Yello = Death, Green =
Recovered
```

- Under the same conditions, period of time, and input the output differs.



Population Size 5000
Infection Probability 0.5
Sickness Duration Min 2 Max 4
Death Probability 0.2
Initial Sickness 1

```
Day 12 -
398 people has been infected today;
41 people has dead;
5 people has been recovered today;
352 people are still ill today;
5269 total amount of people has infected until
today
358 total amount of people has dead until today
Red = Infected,Yello = Death, Green =
Recovered
```



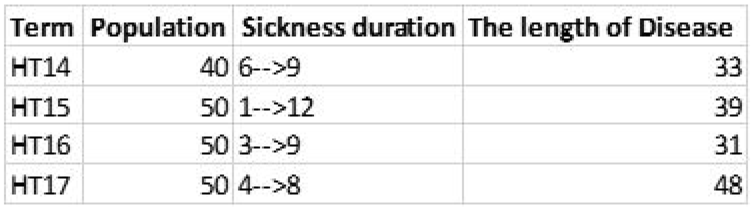
Population Size 5000
Infection Probability 0.5
Sickness Duration Min 2 Max 4
Death Probability 0.2
Initial Sickness 1

```
Day 12 -
448 people has been infected today;
42 people has dead;
2 people has been recovered today;
404 people are still ill today;
3640 total amount of people has infected until
today
266 total amount of people has dead until today
Red = Infected,Yello = Death, Green =
Recovered
```

Input parameters

Semester	N	minDays	maxDays	L	Initial number of ill individuals	Placement
ht17	**40**	**6**	**9**	**0**	**1**	**20,20**
ht16	**50**	**1**	**12**	**0**	**1**	**25,25**
ht15	**50**	**3**	**9**	**0**	**1**	**25,25**
ht14	**50**	**4**	**8**	**0**	**1**	**25,25**

Total amount or sickness and length of disease

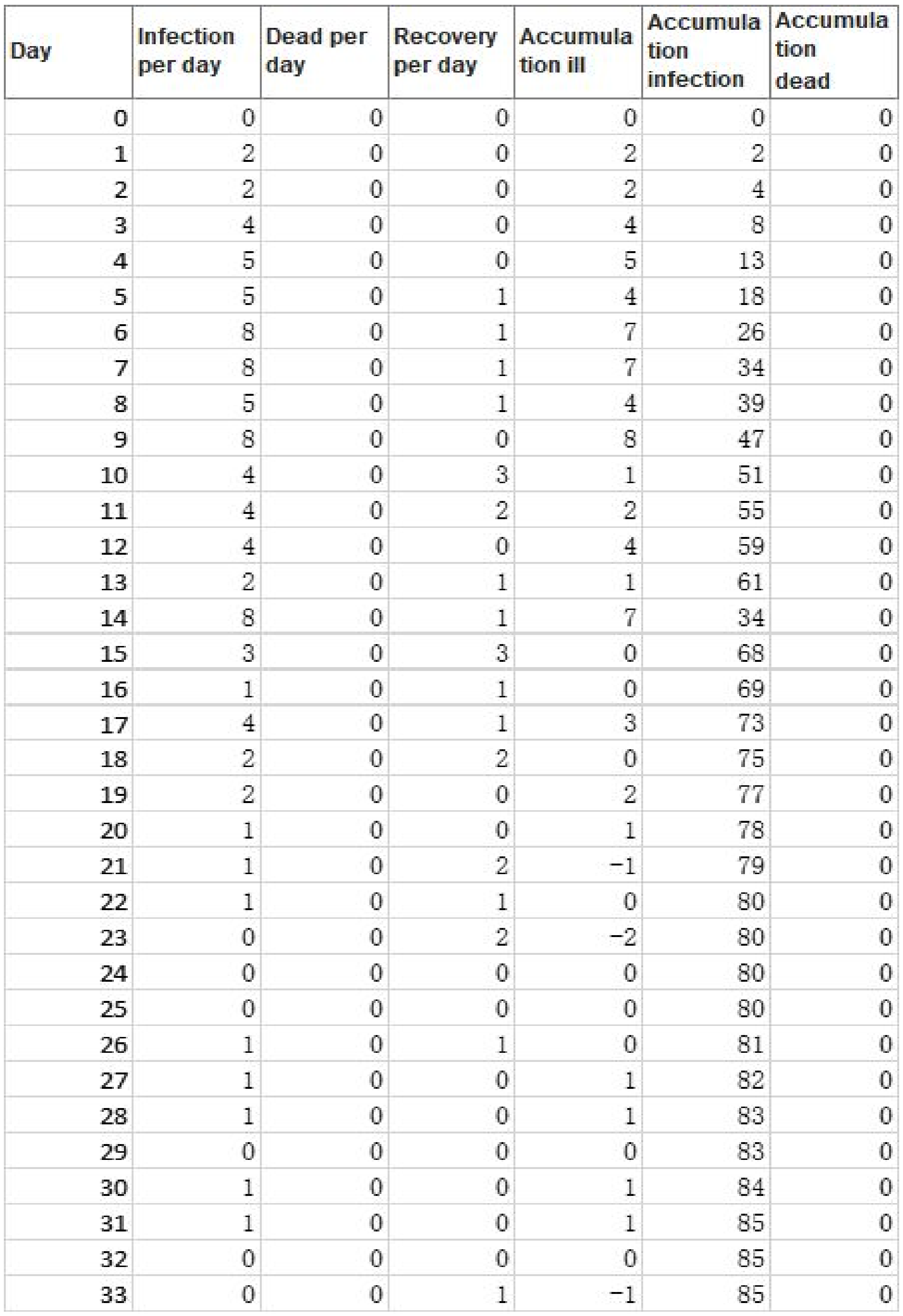


Term	Population	Sickness duration	The length of Disease
HT14	40	6-->9	33
HT15	50	1-->12	39
HT16	50	3-->9	31
HT17	50	4-->8	48

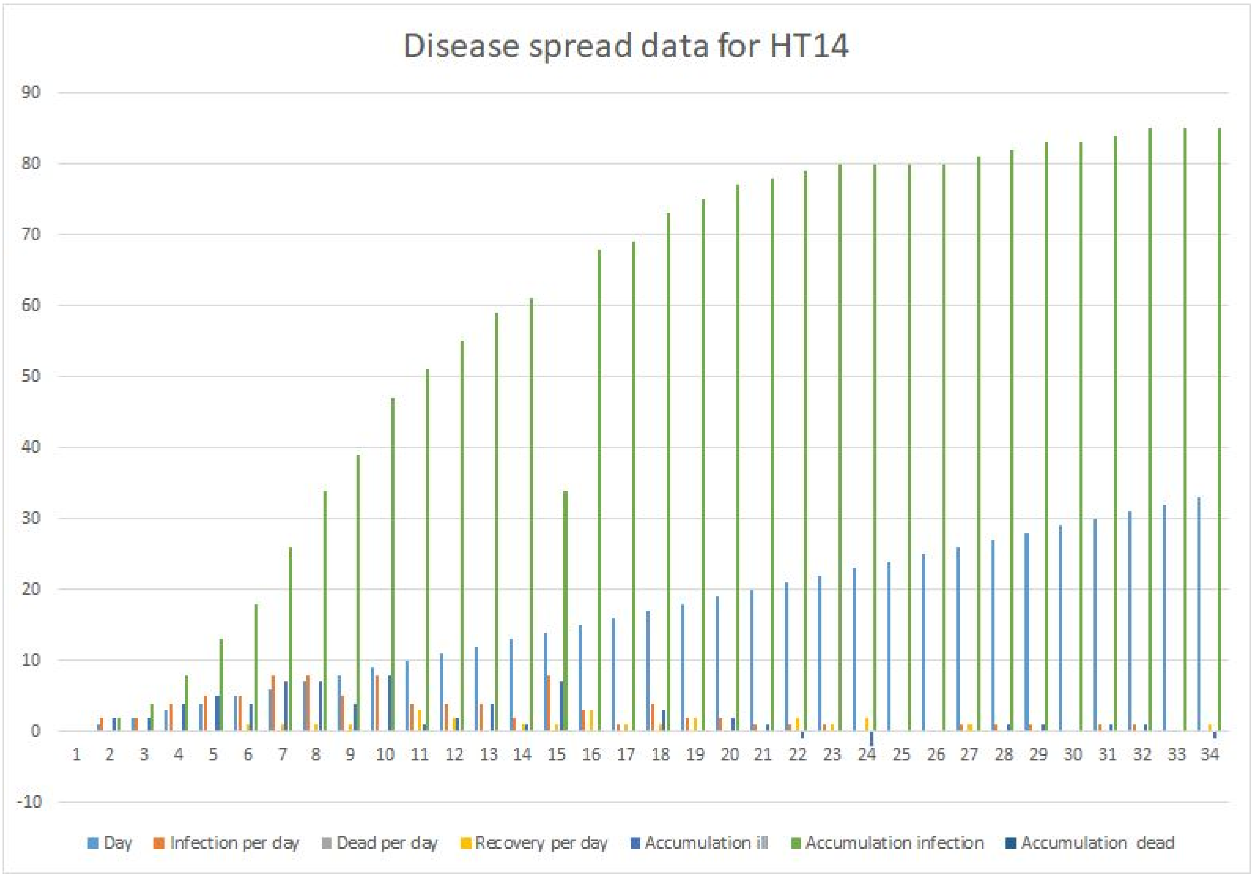
Raw data

Output data

Simulation data for HT14

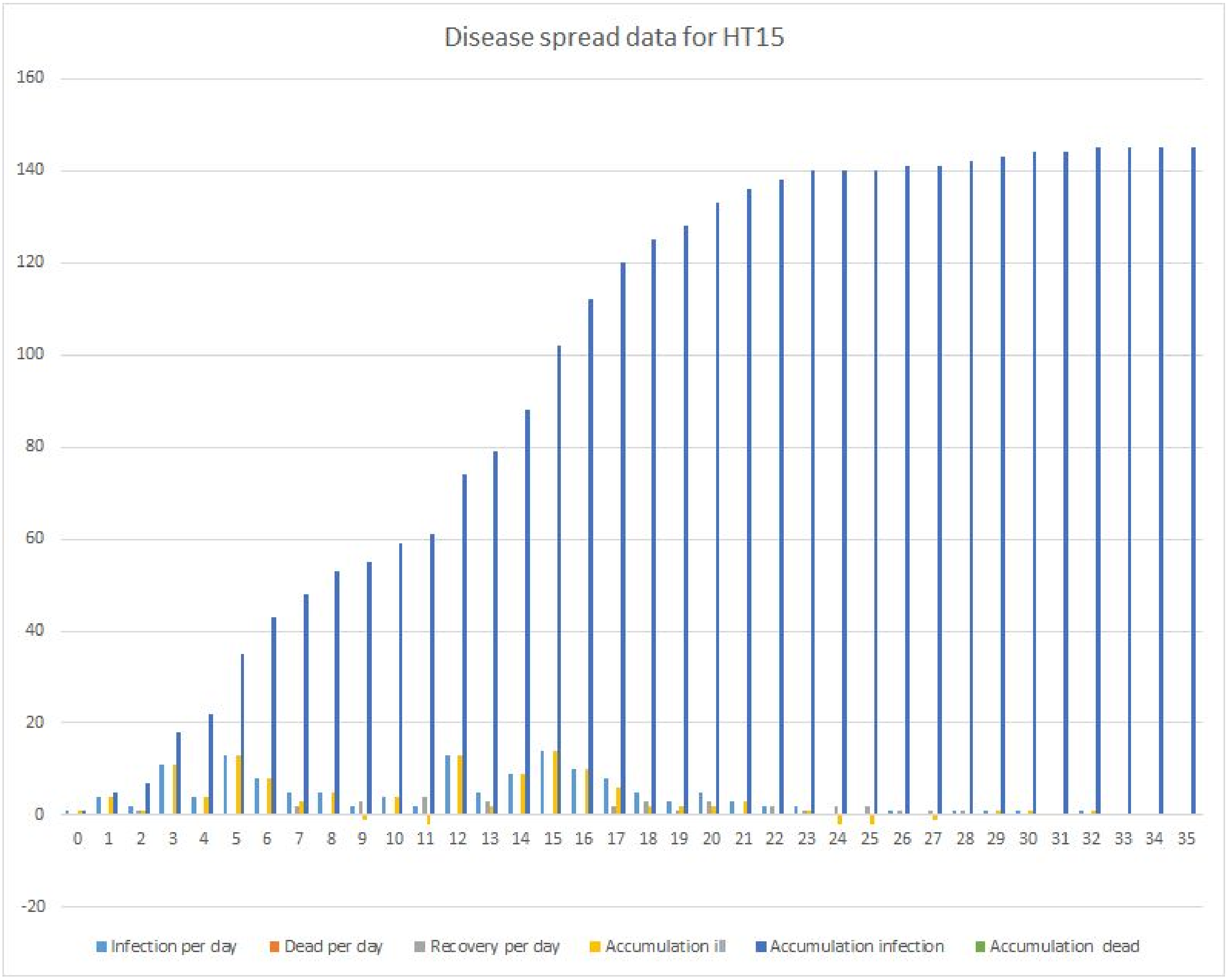


Day	Infection per day	Dead per day	Recovery per day	Accumulation ill	Accumulation infection	Accumulation dead
0	0	0	0	0	0	0
1	2	0	0	2	2	0
2	2	0	0	2	4	0
3	4	0	0	4	8	0
4	5	0	0	5	13	0
5	5	0	1	4	18	0
6	8	0	1	7	26	0
7	8	0	1	7	34	0
8	5	0	1	4	39	0
9	8	0	0	8	47	0
10	4	0	3	1	51	0
11	4	0	2	2	55	0
12	4	0	0	4	59	0
13	2	0	1	1	61	0
14	8	0	1	7	34	0
15	3	0	3	0	68	0
16	1	0	1	0	69	0
17	4	0	1	3	73	0
18	2	0	2	0	75	0
19	2	0	0	2	77	0
20	1	0	0	1	78	0
21	1	0	2	−1	79	0
22	1	0	1	0	80	0
23	0	0	2	−2	80	0
24	0	0	0	0	80	0
25	0	0	0	0	80	0
26	1	0	1	0	81	0
27	1	0	0	1	82	0
28	1	0	0	1	83	0
29	0	0	0	0	83	0
30	1	0	0	1	84	0
31	1	0	0	1	85	0
32	0	0	0	0	85	0
33	0	0	1	−1	85	0


Disease spread data for HT14
90
80
70
60
50
40
30
20
10
0
-10
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
Day
Infection per day
Dead per day
Recovery per day
Accumulation ill
Accumulation infection
Accumulation dead

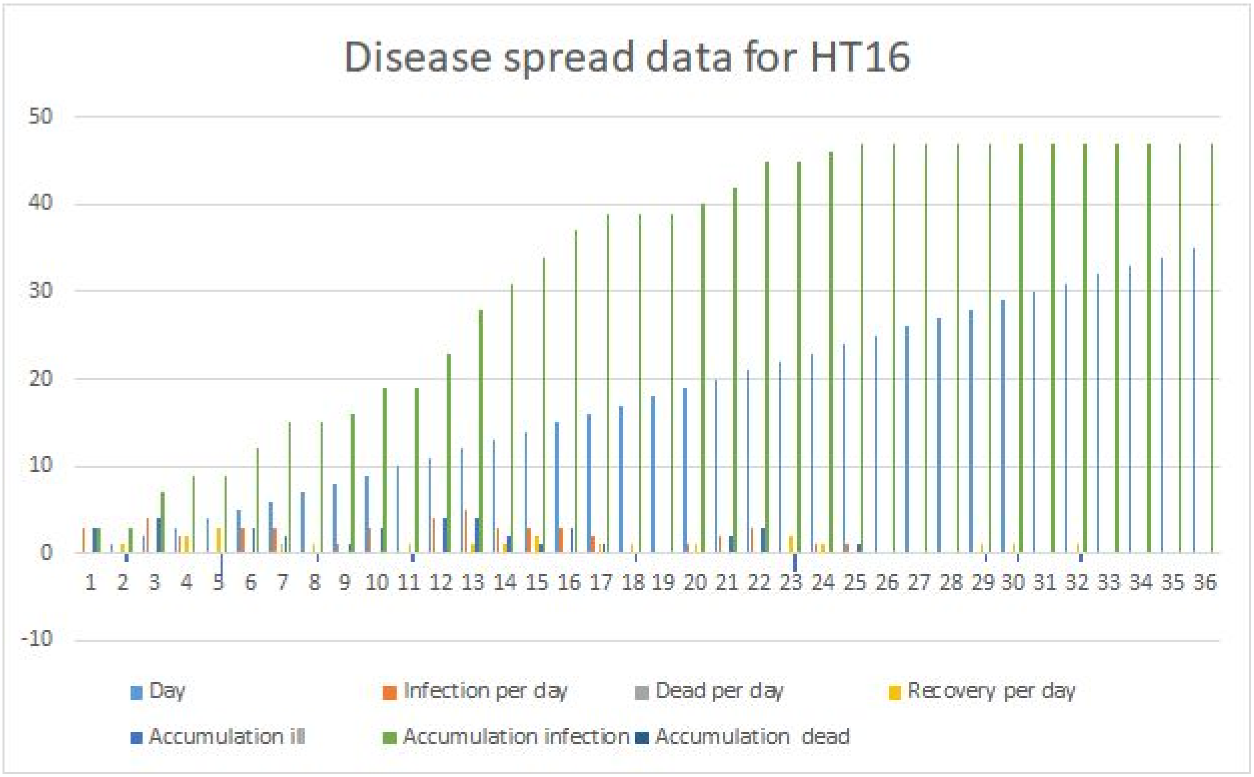
Simulation data for HT15

Day	Infection per day	Dead per day	Recovery per day	Accumulati on ill	Accumulati on infection	Accumulati on dead
0	1	0	0	1	1	0
1	4	0	0	4	5	0
2	2	0	1	1	7	0
3	11	0	0	11	18	0
4	4	0	0	4	22	0
5	13	0	0	13	35	0
6	8	0	0	8	43	0
7	5	0	2	3	48	0
8	5	0	0	5	53	0
9	2	0	3	−1	55	0
10	4	0	0	4	59	0
11	2	0	4	−2	61	0
12	13	0	0	13	74	0
13	5	0	3	2	79	0
14	9	0	0	9	88	0
15	14	0	0	14	102	0
16	10	0	0	10	112	0
17	8	0	2	6	120	0
18	5	0	3	2	125	0
19	3	0	1	2	128	0
20	5	0	3	2	133	0
21	3	0	0	3	136	0
22	2	0	2	0	138	0
23	2	0	1	1	140	0
24	0	0	2	−2	140	0
25	0	0	2	−2	140	0
26	1	0	1	0	141	0
27	0	0	1	−1	141	0
28	1	0	1	0	142	0
29	1	0	0	1	143	0
30	1	0	0	1	144	0
31	0	0	0	0	144	0
32	1	0	0	1	145	0
33	0	0	0	0	145	0
34	0	0	0	0	145	0
35	0	0	0	0	145	0


Disease spread data for HT15
160
140
120
100
80
60
40
20
0
-20
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
Infection per day
Dead per day
Recovery per day
Accumulation ill
Accumulation infection
Accumulation dead

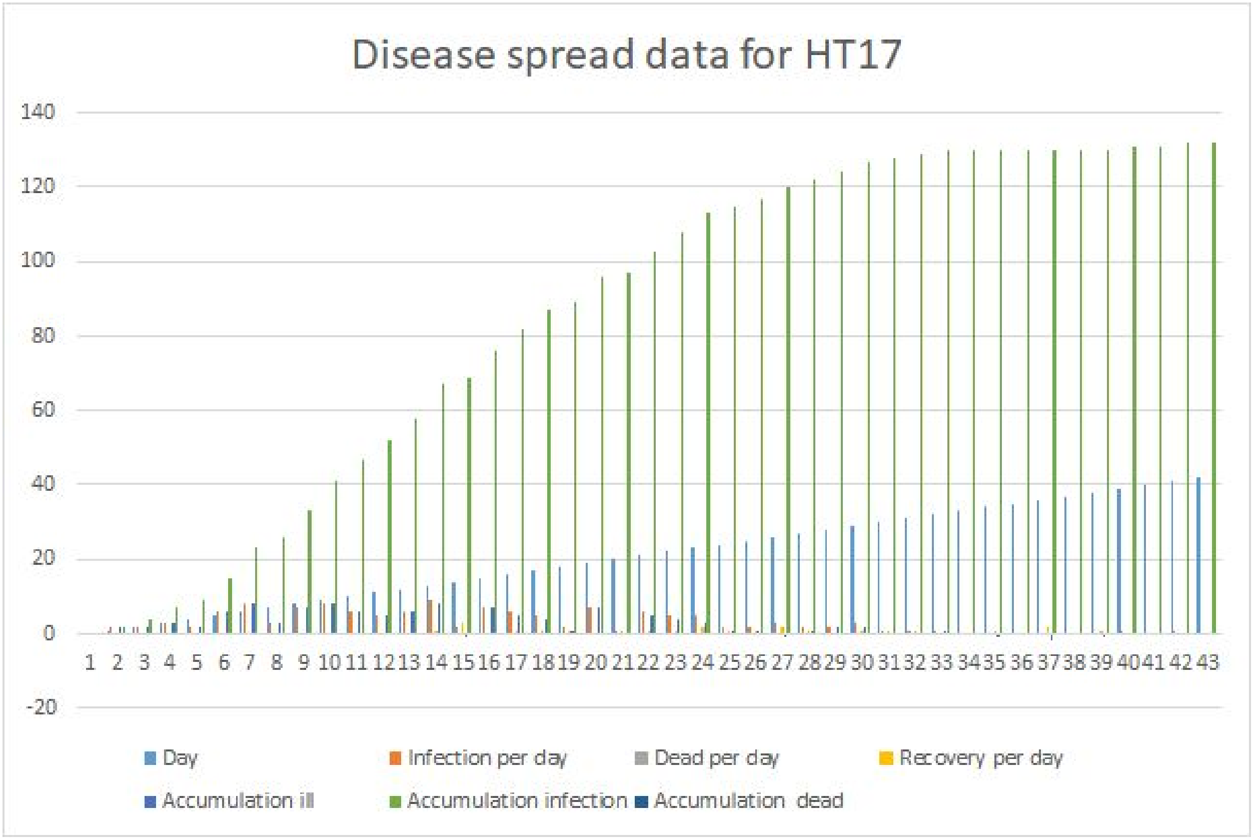
Simulation data for HT16

Day	Infection per day	Dead per day	Recovery per day	Accumulation ill	Accumulation infection	Accumulation dead
0	3	0	0	3	3	0
1	0	0	1	−1	3	0
2	4	0	0	4	7	0
3	2	0	2	0	9	0
4	0	0	3	−3	9	0
5	3	0	0	3	12	0
6	3	0	1	2	15	0
7	0	0	1	−1	15	0
8	1	0	0	1	16	0
9	3	0	0	3	19	0
10	0	0	1	−1	19	0
11	4	0	0	4	23	0
12	5	0	1	4	28	0
13	3	0	1	2	31	0
14	3	0	2	1	34	0
15	3	0	0	3	37	0
16	2	0	1	1	39	0
17	0	0	1	−1	39	0
18	0	0	0	0	39	0
19	1	0	1	0	40	0
20	2	0	0	2	42	0
21	3	0	0	3	45	0
22	0	0	2	−2	45	0
23	1	0	1	0	46	0
24	1	0	0	1	47	0
25	0	0	0	0	47	0
26	0	0	0	0	47	0
27	0	0	0	0	47	0
28	0	0	1	−1	47	0
29	0	0	1	−1	47	0
30	0	0	0	0	47	0
31	0	0	1	−1	47	0
32	0	0	0	0	47	0
33	0	0	0	0	47	0
34	0	0	0	0	47	0
35	0	0	0	0	47	0


Disease spread data for HT16
50
40
30
20
10
0
-10
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
Day
Infection per day
Dead per day
Recovery per day
Accumulation ill
Accumulation infection
Accumulation dead

Simulation data for HT17

Day	Infection per day	Dead per day	Recovery per day	Accumulation ill	Accumulation infection	Accumulation dead
0	0	0	0	0	0	0
1	2	0	0	2	2	0
2	2	0	0	2	4	0
3	3	0	0	3	7	0
4	2	0	0	2	9	0
5	6	0	0	6	15	0
6	8	0	0	8	23	0
7	3	0	0	3	26	0
8	7	0	0	7	33	0
9	8	0	0	8	41	0
10	6	0	0	6	47	0
11	5	0	0	5	52	0
12	6	0	0	6	58	0
13	9	0	1	8	67	0
14	2	0	3	−1	69	0
15	7	0	0	7	76	0
16	6	0	1	5	82	0
17	5	0	1	4	87	0
18	2	0	1	1	89	0
19	7	0	0	7	96	0
20	1	0	1	0	97	0
21	6	0	1	5	103	0
22	5	0	1	4	108	0
23	5	0	2	3	113	0
24	2	0	1	1	115	0
25	2	0	1	1	117	0
26	3	0	2	−1	120	0
27	2	0	1	1	122	0
28	2	0	0	2	124	0
29	3	0	1	2	127	0
30	1	0	1	0	128	0
31	1	0	1	0	129	0
32	1	0	0	1	130	0
33	0	0	0	0	130	0
34	0	0	1	−1	130	0
35	0	0	0	0	130	0
36	0	0	2	−2	130	0
37	0	0	0	0	130	0
38	0	0	1	−1	130	0
39	1	0	0	0	131	0
40	0	0	0	0	131	0
41	1	0	0	0	132	0
42	0	0	0	0	132	0


Disease spread data for HT17
140
120
100
80
60
40
20
0
-20
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
Day
Infection per day
Dead per day
Recovery per day
Accumulation ill
Accumulation infection
Accumulation dead

Individual Reflection

Individual reflection 1

-Hussam

As engineers, we must be able to solve problems. And these solutions must be correct, effective and good. We often look for the most optimal way to solve problems. Hence, we have to look at the bigger picture to be able to find the fastest, cheapest and most efficient solution. Engineers should be able to master these skills, because it is part of their daily routine in solving problems and getting involved in different activities and phases of designing products. It is great to be able to invent something that works, but it is important to be able to show that it works and serves the purpose and also that it is tested in a scientific way.

First of all, we need to be able to identify the assumption that this module is based on. After we have these assumption clear, we need to analyze this data and see in what circumstances it can give us different results. When we figure out different solutions to one problem, we need to select the most efficient one according to the resources that we have. And we do that, by comparing the advantages and the disadvantages of each of the solutions.

It is important to consider the validation and verification step, because we need to prove that our system is valid to be able to present a system that provides a solution to the problem.
It might be a bit difficult to go from a problem to model, because we are moving from a small step to generalize the problem and outline the requirement. The process takes longer, because we don't approach problems directly or don't go with the instinct solution that comes to mind, but we follow a scientific structure that requires a different mindset and some patience to be able to cope with what is going on and what is to be done.

I have learnt a lot about building a system from only knowing the problem, and I could see the changes during the whole process of designing the system. I can see the difference between the very early stages and the late ones. There is however a lot to be learned in the future about this topic. The more I work with such problems, the better I get.

Individual reflection 2

-Qi Li

In out future career life as a engineer to be able to identify what the customer need and identify what they want is the necessary step. This can help us clearly define our goal and the best way to reach it. During this assignment i realize i have issue towards the need and want which usually not the case for me. Long term study life has make myself focus too much on the theoretical level knowledge. At first of few days i start using multithreading to trying to achieve the maximum performance and forgot we can easily solve this issue with with simple multidimensional array. The mistake on define the wrong model has waste a lot of time. During the verification for each step, I have also realize that I am think stuff too straight and forget how computer thinks. Few of the function took me a while because of the returning result and process number is correct but i think it is wrong because it does not make sense in normal life. This is mostly due to the mental model of each function is not clear before i write it. A piece of paper and pen to model each step before i start coding is probably a good idea for future assignment. There are always possibility to find the shortcut if the mental model is correct before implementation. 50 LOC can always become 5 when different people are doing the same assignment. During the test, verification and validation phase, i feels like i start getting fuzzy for some complicated function. Especially when several loop happend together with different function calling. I should be more careful with those step and patient with them. Because this is the place that most likely result will go wrong. Even a little uncertainty here might lead to chain react and cause huge effect to the final result.