CSE 443

Object Oriented Analysis And Design

Final Report

151044014

Ferhat ÇELİK

Since Javafx is newer than java swing and it uses the mvc structure, I chose it and started my homework. It provides the opportunity to create an interface using the Javafx XML language. I used a scene builder with intellij while creating the interface of the project. All elements created and designed in Xml can be accessed in the Controller class. Therefore, every update to be done on the interface must be done by the controller. In order to use the controller in other classes, I would pass the controller I created in Main to the classes that need to use it as a parameter.

According to my structure, each individual in the community should have 6 basic characteristics:

To wear a mask or not

Speed

Social Distance

Social factor

Healthy sick or dead

Uniq id

Since every person must have a unique id, I have kept a static integer value named largestid and during each person object creation, I increase this value by one and assign it to the id value.

I used the Builder design pattern because the Person class has too many variables and it will be difficult to assign to these variables during each object creation. In this way, the user can select these features in the order they want and as many as they want in the gui. For example, the speed variable of each person is initially 1 by default, and if the user does not want to change it, a constructor will have to be written in which the speed variable is not taken as a parameter. This causes the same problem for every variable. In addition, other properties can be added to each person in the future and a constructor must be written for these properties.

For these reasons, I wrote the PersonBuilder class and this class should be used for every person to be created.

Personbuilder takes a mediator object as a parameter to the build method.

Because every person in the community must have a mediator.

Person's update method calls the update method of the mediator object with itself as a parameter.

People are created according to user inputs in the Main class and each created person is registered in the mediator using the add person method of the mediator.

Every person who wants to act on the canvas should know the positions of other people and act accordingly. Since it would be quite costly for every human object to communicate with each other, every person object communicates only with the mediator and the mediator performs all updates.

There should be square shapes on the canvas to represent each person.

These square shapes must have coordinate and velocity vectors to determine their current position and speed.

Since it would be illogical to store these values in the person object, I wrote an interface named Shape.

Each shape's coordinate, velocity , etc. There are getter and setter methods to retrieve and change this informations, and draw methods to draw it on the canvas. Since it is desired to use square shapes to represent each person, I implemented the class named square from the shape interface.

The add person class of the mediator class creates a square object for each added individual and returns this person object as a parameter to the shape object it creates. Shape object takes the id variable of the person object it receives as the parameter and assigns this variable to its own id variable.

In this way, the person and the shape have the same id. This shape is added to the shapeList kept in the mediator.

Every time a person object calls the update method, the mediator checks for any collisions and finds the shape object with the same id as the person object and updates its coordinates.

The mediator also controls whether the people represented by the two colliding objects infect each other with disease, and important information such as crash times, death times, and hospital admission.

According to the structure I have established, 2 healthy people or 2 sick people cannot transmit disease to each other, so when they collide, they continue on their way in opposite directions without waiting.

If one of the 2 people colliding is healthy and the other is sick, they wait for the specified time and the rate of illness is calculated and it is determined whether the healthy person will be sick.

I wrote the StopWatch class to determine whether people who are sick are hospitalized or time left to die.

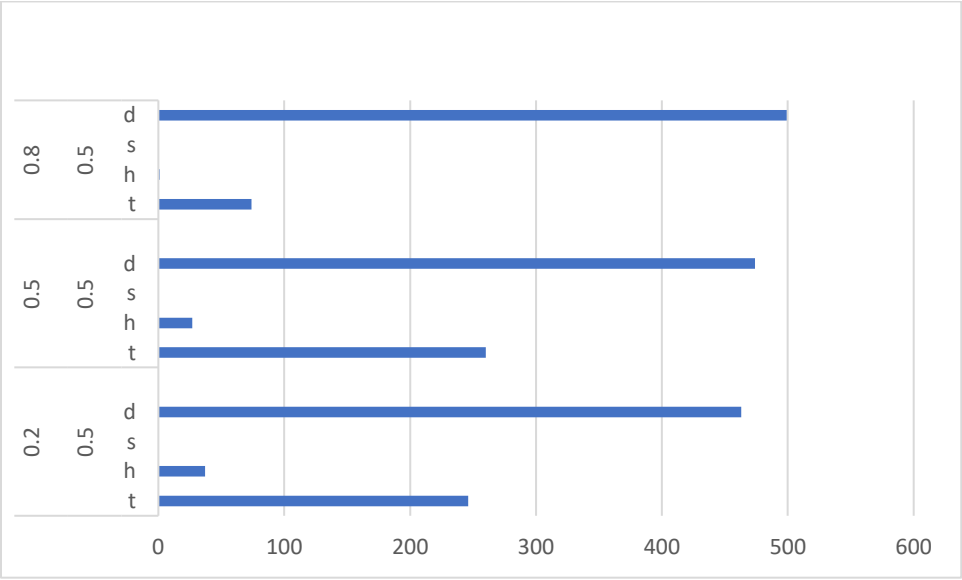
Finally, I draw real time graphs in order to better see the data of the disease spreading in the society.

Every person who gets sick is taken to the hospital if there is a place in the hospital. The Produce method checks this for each patient. The Consume method checks whether the individuals who are hospitalized in each cycle are expired, and those that have expired return to the canvas in a healthy way.

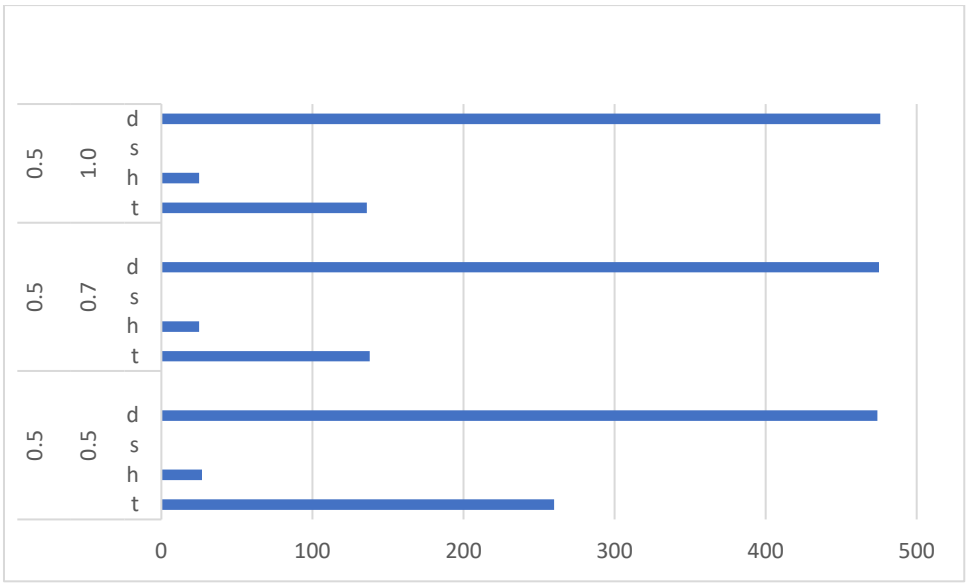
Note: Death and hospital timers will not start until the first patient infects a person.

My charts :

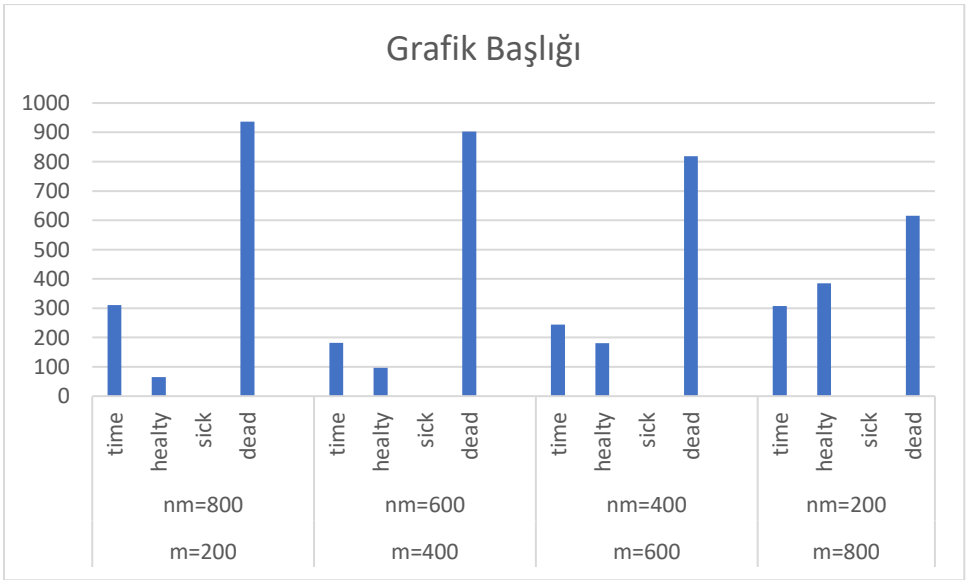
Mortality Rate: 0.2 – 0.5 – 0.8 Spreading Rate: 0.5 t : elapsed time



Spreading Rate : 0.5-0.7-1.0 Mortality Rate: 0.5 t : elapsed time

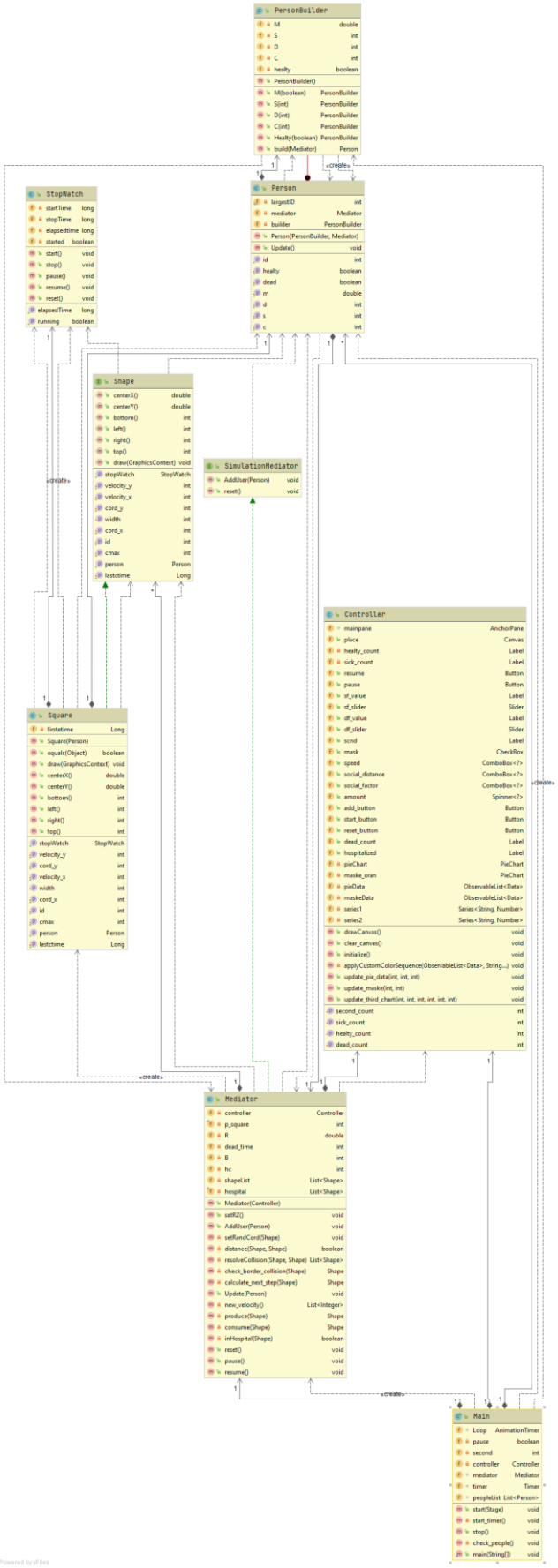


Spreading Faktor : 0.5 Mortality Factor: 0.6 m : wear mask nm : not wear mask



According to the graphics I have created, I think the rate of wearing a mask has an exponential relationship with the number of people who remain healthy.

My Class Diagram :



Sample output : [Click to watch simulation]

UI :

You can press add button one more time.

Spinner is editable.Firstly you should edit and press increase or decrease button one time.