Virus Spread Disease

We decided to do a simulation where we could see how viruses are spread. This was driven by the idea of COVID-19 which we all suffer from and where many simulations were done to prove different things, like how the virus may spread (speed), how social distance impact the development of the disease, and how in some countries the disease ended up being more lethal than in others. Therefore this simulation doesn't intend to create a new way to analyze infection diseases but to show us with graphics how other people analyzed it in that time. In this simulation we tried to leave some important values in the main in order to model different viruses and their impact.

Boards and Profiles:

-Board: In our simulation the board is represented by the list of people and their interactions. We can model it by using a 2D array where the element at position [i][j] is '1' person i is acquainted with person j, and '0' otherwise.

-Profiles: these are represented by the status of each person, being these "healthy", "sick", "cured", or "dead".

SGS dynamics:

The dynamics of our social game can be modeled through feedback (ϕ) and spawn (σ) functions.

-Feedback function(ϕ): this function defines how a person's state changes based on interactions. In the simulation this is handled in the "handleinfection" method.

-Spawn function(σ): In this context, spawning new players might not be relevant as we are working with a fixed set of people.

Game configurations and Transitions:

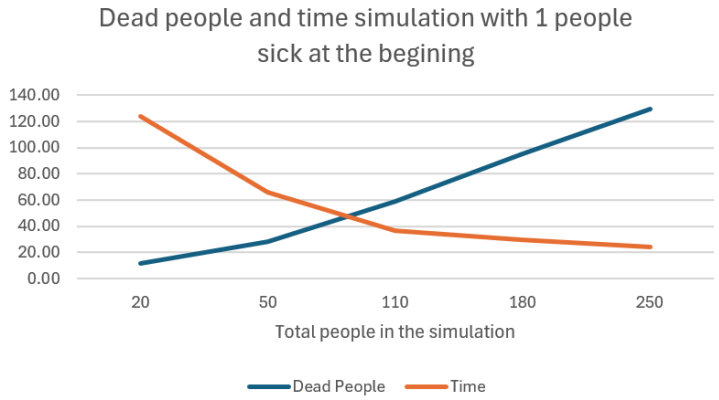
Each configuration is a pair (β,π), where β is the board and π is the profile. The transitions are performed by the simulation loop.

Results:

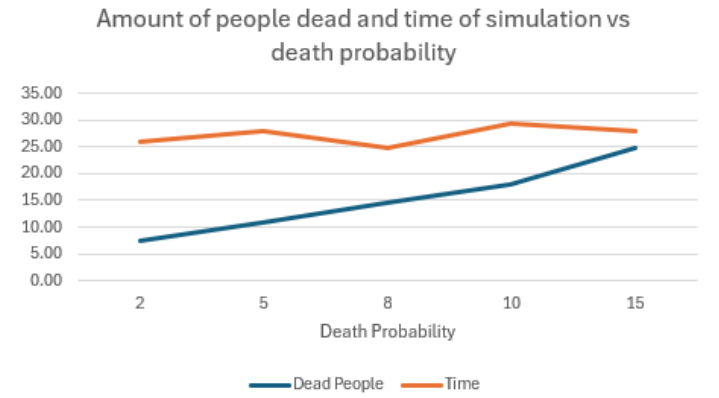
In order to analyze this project we found it interesting asking some questions. For starters if the amount of people with which we run the program impacted in some way the spreadness of the disease. In theory, we gave the hypothesis that it did, if there are more persons in one country, and the deadly probability is the same for all, then in the countries with more population, more people is going to end up dying. But we wanted to prove this, so here we have the graph that proves that hypothesis.

In this graph we can see two lines, the first one with the amount of people that die in the simulation. The second one with the time that took the simulation to end (taking into account that the simulation stops when there are no people left infected, so there can only be people healthy, cured or dead). We ran the simulation 30 times with each amount of people, the death probability was equal for all of them and also the initial probability of sickness and the probability of being cured. Here we can see that if we increase the amount of people in the simulation, then the amount of people that dies increases and the time the simulation takes to end decreases. The interesting part here is not the one we were looking for (the amount of

people dying) but the time. This would mean that in countries with more population, the infection spreads faster and therefore it should end faster.



Then, we decided to explore more the first graph but with one change, instead of giving an initial probability of sickness, we started with a 0 patient. To see how many people could one person infect. Again we changed the total population, and for each of them we ran the simulation 30 times. As you can see, the the tendency was really similar as the first one, but we can see two important changes, the first one is that the least time that took the simulation to end was 22 seconds approximately while in the first one was almost 10, the biggest time as well was bigger in the second case with a little more than 120 seconds while in the first one was 80 seconds. The second thing we can see is the increase in the number of dead people, with a simulation of 250 people the first time approximately 40 people died, while in the second one 121 people died.



Finally we make this graph that shows what happens if we change the probability of people dying. This can represent the effect of different sickness (maybe ones that are more deathly than others). For this topic the result was what we expected, if you increase the death probability the amount of people that die increments. But in the other hand we thought that if we increment this probability then the simulation would end faster, and in the graph we can see that it didn't. This experiment was made with the same amount of people at the beginning, the same probability of starting sick and the same probability of getting cured.

Conclusions:

During this project we learn Java through exploring the elaboration of a social simulation to answer some interesting questions. We also learned how to plan a problem and model it in math and in java. Some of the things we implemented in this project that we learned in class were Generics, Arraylists, and threads. We use generics and arraylist to create a list that would contain all our participants, the use of generics is to guarantee that only the objects of type “person” would be added to the list, and therefore avoid errors at compilation time. We also use threads to run the gameLoop in parallel with the graphical user interface, and we did this to avoid the graphics freeze or stop while the gameLoop didn’t.
In terms of the graphics we presented above, the experiment shows that in countries that are more populated, more people die. Also that with more people the spread of the disease is faster and therefore it disappears faster. Finally, as with COVID-19 one person can end up infecting more than half of the population of one country. With all this information we can conclude that isolation is really necessary, because if we led people out in the streets one people may cause a pandemic, and that there is only two ways of end a pandemic, one is either you isolate people and wait so that the sick people cure themselves (that could take a while but you get sure the least amount of people die) or you led the disease spread (which is going to end up with lots of people dying but the sickness will disappear faster).

1. Population Model

The simulation manages a population of people, each of whom can be in one of the following states: healthy, sick, cured, or dead. Mathematically, we can define the set of states

S = healthy, sick, cured, dead

Each person in the population has a state s(i) E S and a position x(i), y(i) within the simulation area.

2. State Transitions

People can change their state in each iteration of the simulation. State changes are probabilistic and determined by the configured probabilities:

- Probability of getting sick: Represents the probability that a healthy person will get sick when interacting with a sick person. Denote it as P*getsick*

- Probability of getting cured: Represents the probability that a sick person will get cured. Denote it as P*cure*

- Probability of dying: Represents the probability that a sick person will die. Denote it as P*death.*

State transitions can be modeled using a Markov chain, where the states and transition probabilities are:

- P (healthy $\Longrightarrow$ sick) = P*getsick*
- P (sick $\Longrightarrow$ cured) = P*cure*

- P (sick $\Rightarrow$dead) = Pdeath

3. Population Initialization

The initial population is created according to a given configuration. If the initial sick probability is -1, a "Patient Zero" (a single initially sick individual) is introduced:

P*healthy* = 1,for all other individuals

If there is an initial probability of being sick P*initial_sick*, then that state is randomly assigned to a fraction of the population:

P*healthy* = 1 - P*initialsick*, P*sick* = P*initialsick*

4. Updating and Movement

In each iteration of the simulation loop, each person's state and position are updated:

1. Movement: People can move randomly within the simulation area. Assuming simple random movement, the new position (xi', yi') of a person “i” can be:

$xi' = xi + \Delta x, yi' = yi + \Delta y$

where Δx and Δy are random displacements in the “x” and “y” axes respectively.

2. Interactions and State Changes: If a healthy person is within an infection radius of a sick person, there is a probability P*getsick* of getting infected. This can be mathematically modeled as:

$d(i, j) \leq r \Rightarrow$ Person(i) has P*sick*

where d(i, j) is the Euclidean distance between person “i” and person “j”, and “r” is the infection radius.

5. Termination Conditions

The simulation ends when there are no sick people left. This is checked in each iteration:

SickCount = 0 $\Rightarrow$ stopSimulation

Summary

The simulation involves creating a population with defined initial states, updating states and positions probabilistically and randomly in each iteration, and checking termination conditions. Mathematically, it uses concepts of probability, Markov chains, and basic geometry to model the spread of the virus and the movement of people within the simulation area.