|  |  |
| --- | --- |
| Logo ESGT_Jpeg | Escola Superior de Gestão e Tecnologia |

Ano letivo 2019/2020, 2º semestre

Linguagens de Programação

Simulador de Pandemias

**Discentes**:

André Páscoa 190100366

André Carvalho 10100122

**Docentes**:

Prof. Ricardo Rodrigues

Prof. António Roberto

*Santarém, 13 de junho de 2020*

**Índice**

Conteúdo

[Introdução 1](#_Toc42980323)

[Descrição do Sistema 2](#_Toc42980324)

[Considerações Finais e Conclusão 4](#_Toc42980325)

[Bibliotecas Extra 5](#_Toc42980326)

# Introdução

O projeto com o tema de simulação de pandemias, tem como objetivo criar um modelo de certos vírus, e o seu efeito numa população concentrada.

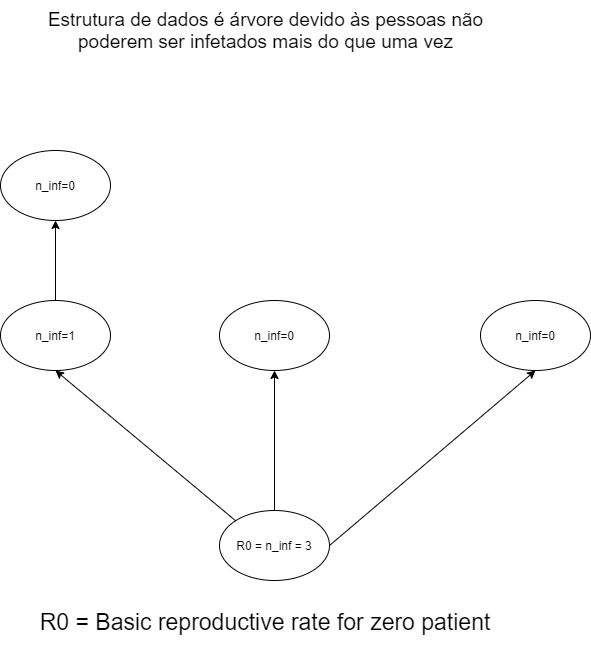
Para alcançar este objetivo decidimos utilizar o JavaFX para dar ao trabalho uma boa ferramenta de visualização do modelo e outras características, inicialmente o nosso objetivo era só fazer o modelo das pessoas a circularem, mas ao longo do tempo apercebemo-nos que era possível concretizar outras tarefas com o nosso conhecimento em interfaces gráficas.

As pessoas possuem 4 estados, suscetível, ou seja pode ser infetado mas ainda não foi, infetado sem sintomas, onde a pessoa pode infetar mas não é detetada ou seja não vai para o hospital, infetado com sintomas, caso o hospital tenha camas vai diretamente para o hospital onde não será colocado em quarentena e finalmente recuperado ou morto dependendo de uma probabilidade que será ajustada pelo utilizador.

# Descrição do Sistema

O nosso projeto foi desenvolvido em 3 versões, 1.0.0, 2.0.0 e 2.1.0, a primeira versão tem uma representação do modelo, onde irá dar para visualizar as pessoas a navegar pela cidade, na segunda versão, o utilizador tem a capacidade de analisar um gráfico com as pessoas infetadas e mortas, nesta versão também será implementada uma versão crude de um visualizador de grafos. Finalmente na versão 2.1.0 o utilizador tem a capacidade não só de visualizar o modelo e um gráfico com as pessoas infetadas e mortas, mas também uma versão completa do visualizador da árvore de pessoas infetadas.

No futuro, este projeto poderá ser evoluído para incluir não só uma cidade, mas um conjunto de cidades que possam formar um país que por si mesmos irão formar continentes, planeta terra, sistema solar etc, etc.



A estrutura de dados utilizada é então uma árvore geral, onde o nó de raiz é o paciente zero, o primeiro a ser infetado. A nossa estrutura de dados vai dinamicamente aumentando baseando-se no simulador implementado, cada node (Pessoa) possui atributos como o número de infetados, nome, id de fotografia (imagem para simular um programa real), estado de infeção entre outros. O R0 é o número básico de reprodução do primeiro paciente, ou seja, quantas pessoas o primeiro paciente infetou.

O simulador foi implementado com colisões simples, ou seja é inicializada a cidade, com n pessoas, cada pessoa vai ter um raio de infeção, uma posição, velocidade e um tamanho(em pixéis), é então gerada aleatoriamente uma posição e velocidade(inclui direção), as pessoas começam a navegar até colidirem com um infetado dentro do raio de infeção, após esta colisão ficaram elas mesmo infetadas, se colidirem com as paredes a velocidade é invertida, o que irá provocar uma inversão de direção.

# Considerações Finais e Conclusão

No desenvolvimento deste projeto encontramos alguns problemas que teríamos de resolver. Como por exemplo inicialmente queríamos que cada pessoa tivesse uma variável independente para o raio de infeção entre outros, mas devido a complicar o código e não termos em mente uma implementação para esta funcionalidade, decidimos não a implementar pois só iria afetar o performance da aplicação, sem trazer alguma coisa de nova, apesar de ser melhor para “future proof”. Este tipo de problemas que nos foram apresentados ao longo do projeto, ajudaram-nos a melhorar bastante as nossas habilidades, desde algoritmos, estruturas de dados e configuração de projetos (maven, java 9 modular) até programação orientada a objetos.

# Bibliotecas Extra

Para fazer a visualização da árvore foram misturados o código do post de stack overflow e da biblioteca por sirolf2009, sendo esta mistura presente no pacote fxgraph. Para ser utilizado o java 9 modular atualizamos a seguinte biblioteca para ser compatível com o nosso projeto. Todo o código exceto o pacote fxgraph e treelayout foi implementado pelo nosso grupo.

<https://stackoverflow.com/questions/30679025/graph-visualisation-like-yfiles-in-javafx>

<https://github.com/sirolf2009/fxgraph>

<https://github.com/abego/treelayout>

Mais as bibliotecas incluídas no ficheiro pom.xml