Problema: Dentro de um mapa, são definidos vários indivíduos, necessário encontrar o local, neste mapa, onde todas as pessoas percorrem o menor caminho possível para se encontrarem.

Requisitos Funcionais:

RF1 – O sistema deve representar indivíduos e arestas para o usuário.

RF2 – O sistema deve permitir a entrada de informações por meio de um documento de texto.

RF3 – O sistema deve permitir a entrada de coordenadas e o comprimento referentes ao posicionamento da linha.

RF4 – O sistema deve permitir a entrada da posição de um indivíduo no mapa.

RF5 – O usuário deve ter a permissão para alterar e excluir linhas e indivíduos.

RF6 – O sistema deve fornecer o ponto de encontro entre todos os indivíduos onde, cada um percorra pelas linhas o menor caminho possível.

RF7 – O sistema deve fornecer a visualização dos indivíduos percorrendo as linhas do mapa em direção do seu ponto de encontro.

RF8 – O sistema deve fornecer um tutorial para facilitar a utilização.

RF9 – O sistema deve representar o ponto de encontro no mapa de maneira chamativa.

RF10 – O usuário deve definir se a linha é mão única ou mão dupla. Caso for mão única, o usuário deve fornecer a direção da linha.

RF11 – O sistema deve exibir, em tempo real, a movimentação dos indivíduos até o ponto de encontro.

RF12 – O sistema deve permitir o desenho de arestas paralelas.

RF13 – O sistema deve conter uma janela interna com o nome dos desenvolvedores.

RF14 – O sistema deve conter um botão para o início da execução onde os indivíduos começam a caminhar para o ponto de encontro.

RF15 – O sistema deve conter uma janela interna somente para a exibição do mapa completo.

RF16 – O sistema deve conter um botão para a finalização do programa.

RF17 – O sistema deve permitir o desenho de arestas curvas.

RF18 – O sistema deve permitir que o usuário navegue nos diretórios do seu computador para encontrar um arquivo texto pré-configurado.

RF19 – O sistema deve permitir a abertura de múltiplas janelas internas.

RF20 – O sistema deve permitir que o usuário escolha medidas parciais e inteiras no tamanho da aresta.

Requisitos Não-Funcionais:

RNF1 – Toda linha deve ser convertida em vértices e arestas.

RNF2 – Utilizar linguagem de programação Java.

RNF3 – O mapa será desenhado utilizando recursos do Java 2D.

RNF4 – O ponto de encontro deve ser calculado utilizando o algoritmo de Floyd Warshall e escolhendo a vértice com menor custo dentre todos os vértices representados na Matriz de retorno do algoritmo.

RNF5 – O caminho entre cada indivíduo e o ponto de encontro deve ser calculado pelo algoritmo Dijkstra.

RNF6 – O arquivo onde será salvo as configurações deve ser no formato .txt

RNF7 – O sistema irá verificar se todos os dados estão corretamente inseridos no arquivo texto para a execução do mesmo.

RNF8 – O sistema utiliza uma lista simplesmente encadeada para o melhor caminho.

RNF9 – A classe de OperadorArquivos apenas deve conter dois métodos: o método para ler os dados e o método para gravar os dados.

RNF10 – Para desenhar e percorrer uma curva, deve-se utilizar o mesmo método.

RNF11 – Deve-se utilizar o padrão de projeto Observer para a comunicação entre tela (view) e a classe controler.

RNF12 – Para a escolha do melhor caminho, deve ser utilizado uma lista encadeada que será necessária para caminhar entre os vértices e arestas.

RNF13 – Para a leitura das informações de um arquivo, este precisa estar em um formato .txt.

RNF14 – As classes Individuo, Aresta e Vértice precisam implementar a interface Desenhavel.

RNF15 – Existe apenas um Frame pai na aplicação.

Regras de Negócio:

RN1 – O ponto de encontro deve sempre ser um vértice do mapa.

RN2 – O indivíduo sempre deve estar em cima de um vértice.

RN3 – A aresta obrigatoriamente deve ter uma direção (Origem -> Destino ou Bidirecional).

RN4 – A aresta obrigatoriamente deve ter um comprimento.

RN5 – Se o comprimento da aresta for maior que o comprimento entre o vértice de origem e o vértice de destino, a aresta é representada como uma curva.

RN6 – Indivíduos serão representados por pontos e arestas por traços, no mapa.

RN7 – O sistema precisa conter um método que realiza a leitura das informações de um arquivo texto e carrega-las em uma lista de desenháveis.

RN8 – No arquivo texto, um indivíduo precisa conter um nome.

RN9 – Mesmo com o arquivo uma vez carregado, o usuário pode realizar alterações no mesmo e carrega-lo novamente.

RN10 – O algoritmo matemático denominado de FloydWarshall se encarregará de definir o ponto de encontro no mapa.

RN11 – Ao iniciar o caminho de cada indivíduo no mapa, a aplicação irá movimentá-los todos simultaneamente até o ponto de encontro “repintando” o frame interno do mapa.

RN12 – O Tutorial de uso estará acessível em qualquer parte da aplicação através do menu superior que pertencerá ao frame pai.

RN13 – O ponto de encontro será destacado com um tamanho e cor diferente dos indivíduos, para o usuário notar sua presença dentro da aplicação.

RN14 – Dentro do arquivo texto, na sessão dos dados das arestas, o quarto atributo será destinado a um valor necessário para o algoritmo de Dijkstra verificar se esta aresta é ou não bidirecional.

RN15 – Arestas paralelas apenas precisarão ter seu comprimento igual ao valor entre o vertice1 e o vertice2 para que seja uma reta.

RN16 – Ao iniciar a aplicação, o sistema irá incluir a primeira sub-tela visível ao usuário que será a que contém o nome dos desenvolvedores do sistema.

RN17 – O botão responsável para iniciar a caminhada dos indivíduos ao ponto de encontro, se localizará dentro do menu superior que pertencerá a sub-tela Mapa. O mesmo irá percorrer a lista de indivíduos e o mesmo irá movimentá-los simultaneamente.

RN18 – A janela do Mapa irá ser acessível no menu superior pertencente ao frame pai, onde o mesmo irá chamar a sub-tela mapa pertencente ao sistema.

RN19 – O botão para finalização do programa se localizará no menu superior pertencente ao frame pai e sua utilidade é justamente finalizar o software.

RN20 – Para o desenho de arestas curvas, o usuário deve pôr um comprimento de arestas maior do que a distância entre os dois vértices da mesma.

RN21 – Para navegar entre os diretórios do computador do usuário, é necessário a utilização do componente JFileChooser pertencente a biblioteca java.swing.

RN22 – Para abrir múltiplas janelas na aplicação, o usuário deve apenas navegar entre o menu superior pai e ir selecionando as opções de menu que lá contém sem fechar as janelas abertas.

RN23 – Para definir o tamanho da aresta, o usuário deve ir até o terceiro atributo da aresta e definir sua medida, podendo ser um valor parcial ou inteiro.