REDE DE TRENS ENTRE CIDADES

A RENFE - Rede Nacional Ferroviária de Espanha - deseja fornecer aos seus usuários um aplicativo que permita verificar os caminhos entre cidades, através de viagens de trem.

Para tanto, um arquivo texto contendo nomes de duas cidades, distância entre elas e preço da passagem é fornecido.

Esse arquivo se chama GrafoTremEspanhaPortugal.txt e representa cidades e ligações entre elas, usando o mapa abaixo como fonte.

Também temos o arquivo cidades.txt, cujo conteúdo é: nome da cidade, coordenada x (longitude) em porcentagem da largura da imagem do mapa e coordenada y (latitude) em porcentagem da altura da imagem do mapa. Use as porcentagens para definir as coordenadas físicas (pixels) na tela do aplicativo, em relação ao tamanho real (pixels) da imagem do mapa que está sendo apresentado na tela. Isso é feito para que, caso o formulário aumente ou diminua de tamanho, a imagem do mapa seja ajustada (ancorada e configurada para adaptar-se ao tamanho de seu container) e as posições das cidades sejam recalculadas proporcionalmente à mudança do tamanho físico da imagem.

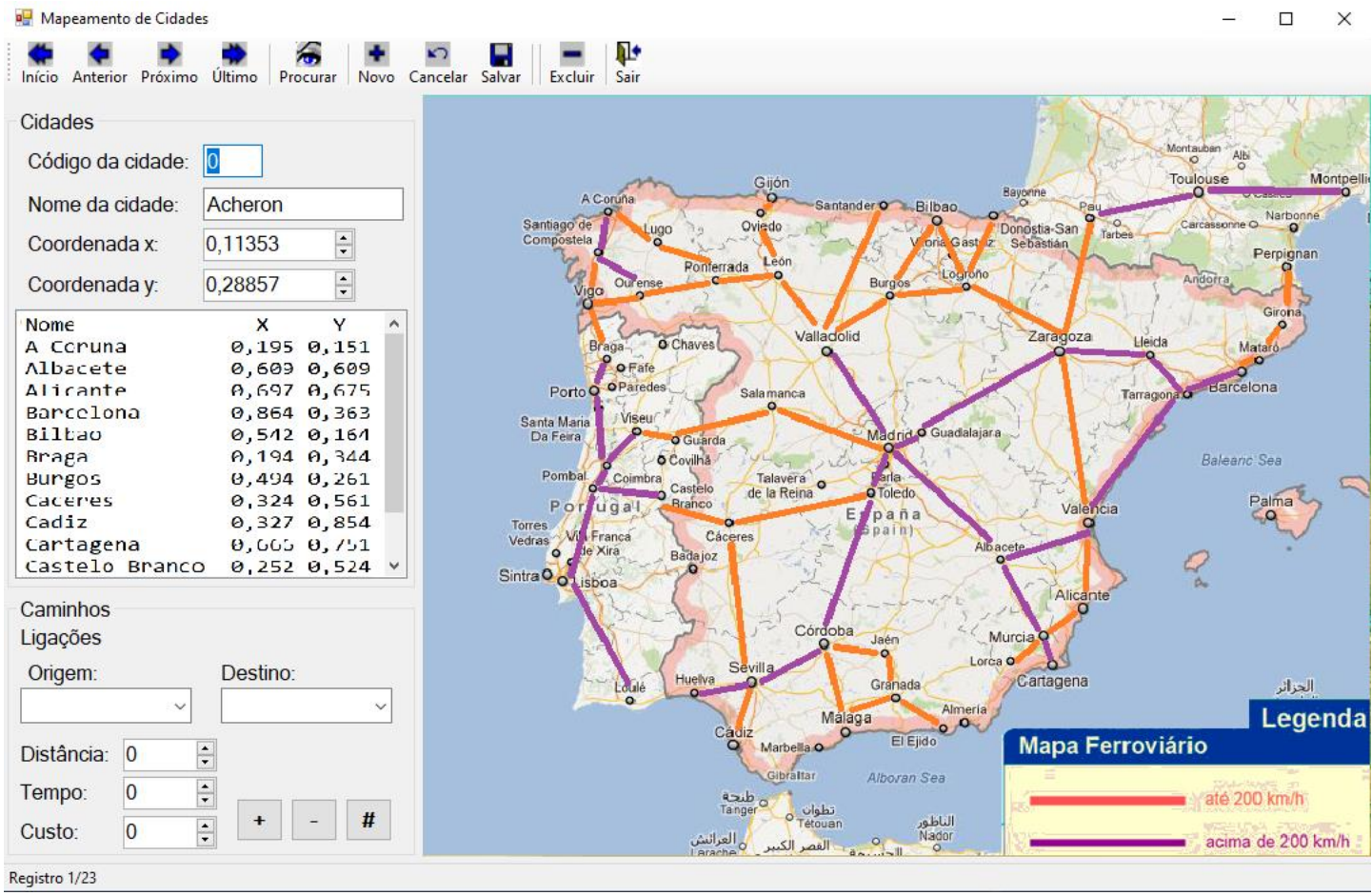
Obviamente, o mapa no seu tamanho original não caberá na tela. Portanto, permita que o mapa seja armazenado num componente PictureBox que se ajuste ao tamanho da tela e lembre-se que isso mudará as coordenadas de exibição de cada cidade no mapa **proporcionalmente** à mudança da altura y e largura x do mapa apresentado na tela, numa proporção entre a largura e a altura da tela com a coordenada (X, Y) original da cidade.

O programa deverá implementar os métodos da Classe **ListaDupla** solicitados pela Interface **IDados**, usando uma lista duplamente ligada como estrutura de armazenamento e recuperação de

dados. Essa interface define métodos que deverão ser usados no formulário para navegar na lista duplamente ligada de cidades, incluir, excluir, exibir, pesquisar, listar dentre outras funcionalidades, de maneira semelhante à manutenção de dados que estudamos no 1º semestre do curso (vetores de objetos).

Esse programa é a primeira parte de um projeto mais completo que, ao final deste semestre, buscará caminhos entre as cidades que forem cadastradas nos arquivos, usando a técnica de Backtracing que estudaremos.

Dessa forma, a parte de “Caminhos” ainda não será feita, apenas a de “Cidades”.



No início da execução do programa, um OpenFileDialog deve ser exibido para que o nome do arquivo com os dados seja definido pelo usuário e o programa possa ler esse arquivo e armazenar seus registos na lista duplamente ligada.

Após isso, o programa deverá mostrar cada cidade no mapa, usando um círculo preenchido e o nome da cidade para mostrar onde a cidade se encontra.

O mapa poderá variar de tamanho caso você aumente ou diminua o tamanho do formulário. Assim, como as coordenadas X e Y são proporcionais, para determinar a o pixel (x, y) exato da cidade no mapa que está sendo exibido, a largura do mapa deve ser multiplicada pela coordenada X da cidade e a altura do mapa deve ser multiplicada pela coordenada Y da cidade. Por exemplo, se o mapa estiver sendo exibido com tamanho de 1200 x 600 pixels (largura x altura), para a cidade de Salamanca do exemplo acima as coordenadas onde o círculo deve ser exibido nesse mapa será (441, 244). Você pode escrever o nome da cidade acima do círculo.

A atualização do PictureBox para que os nomes de cidades sejam exibidos é feita no evento Paint do PictureBox.

Após a exibição dos círculos e nomes das cidades no mapa, o programa deve entrar em modo de navegação (SituacaoAtual = navegando) e o ponteiro Atual da lista deve ser posicionado no primeiro nó. O nó exibido deve ser o nó atual, se houver.

Conforme os botões de navegação forem pressionados, o ponteiro Atual deve ser movido dentro da lista duplamente ligada, através dos métodos de navegação que você codificará ao implementar os métodos já especificados na classe ListaDupla.

Use o nome da cidade como chave de pesquisa e ordenação da lista duplamente ligada. Novas cidades devem ser incluídas na lista de forma a mantê-la ordenada. Para montar a lista, leia o arquivo cidades.txt, que deverá estar ordenado previamente à leitura.

Ao incluir uma cidade, codifique o evento Click do PictureBox de modo que seu parâmetro **e** (EventArgs) informe a coordenada (x,y) em pixels onde o mapa foi clicado, e converta essa coordenada em porcentagem da largura e da altura do mapa, para fornecer os valores dos numericUpDown referentes aos campos Coordenada X e Coordenada Y.

Ao excluir uma cidade, remova-a da lista duplamente ligada.

Quando o programa for finalizado, deve-se gravar os dados armazenados na lista duplamente ligada. Essa gravação será feita no arquivo selecionado pelo usuário no início da execução do programa.

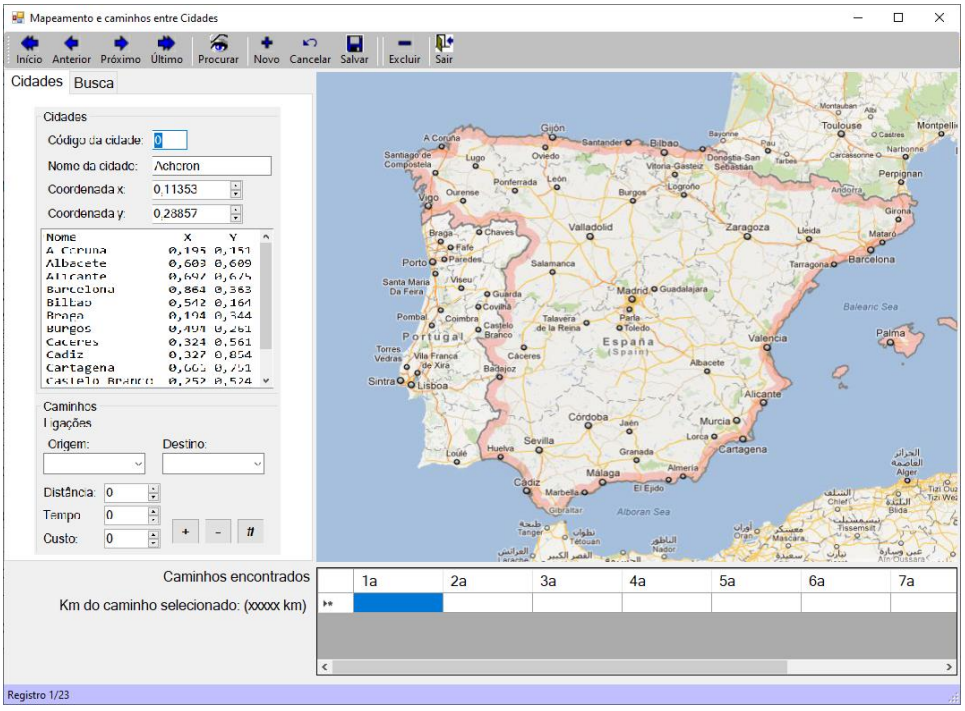
Descrição dos arquivos

CidadesMarte.txt
NomeCidade – string, 15 posições
CoordenadaX – real, 6 posições
CoordenadaY – real, 6 posições

CaminhoEntreCidadesMarte.txt
CidadeOrigem – string, 15 posições
CidadeDestino – string, 15 posições
distancia – inteiro, 4 posições
custo – inteiro, 4 posições

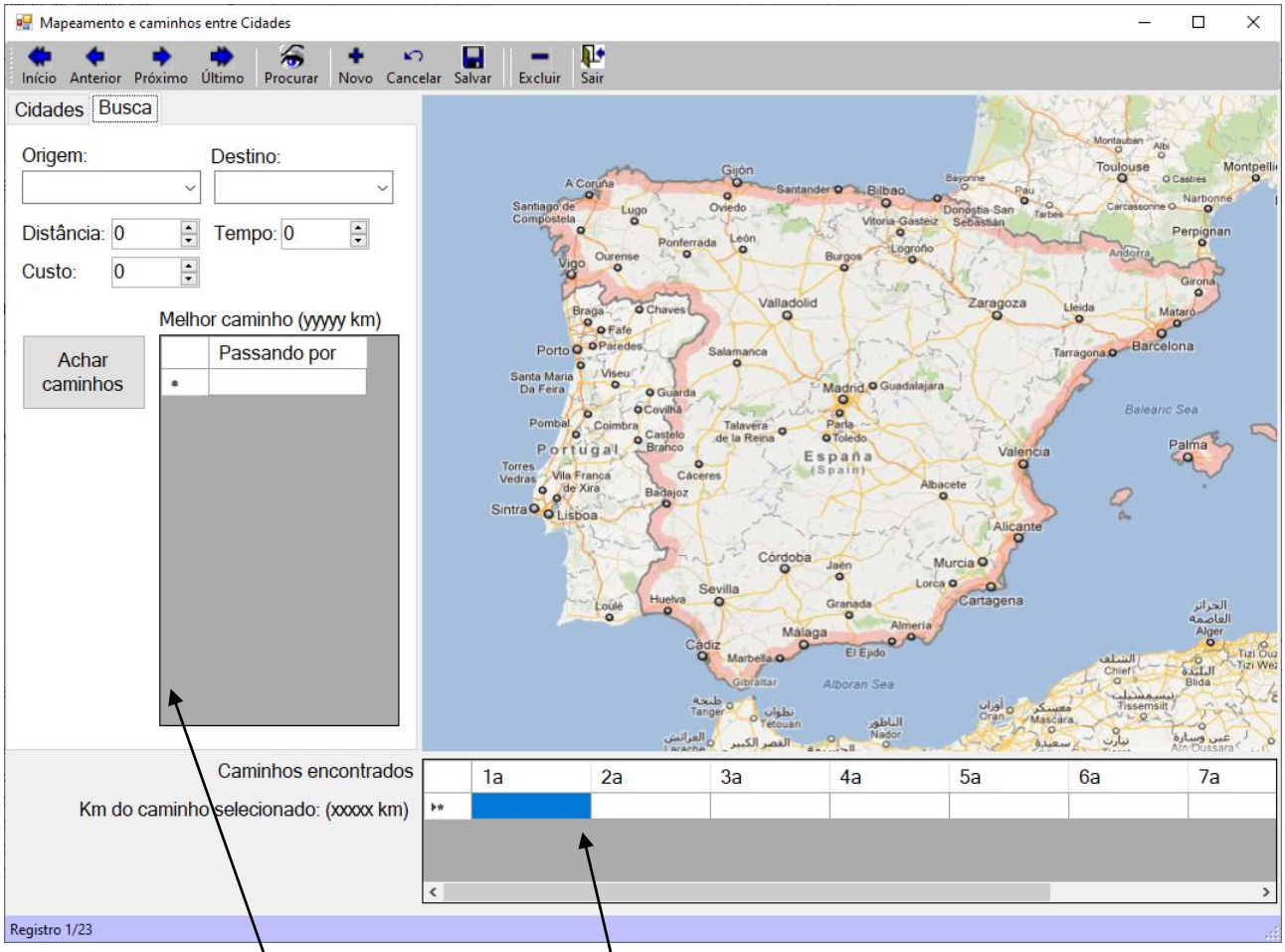
Parte II – Busca de Caminhos

Continuando a Parte I já desenvolvida, o aplicativo deverá encontrar todos os caminhos entre as cidades de origem e de destino escolhidas e relacionar o caminho mais vantajoso, de acordo com o critério de menor distância percorrida.



Modifique o formulário, colocando um TabControl com duas abas e movendo os controles de cadastro de cidades para a primeira aba, como vemos na figura anterior.

Na segunda aba, intitulada Busca, coloque os controles da figura abaixo, que permitirão fazer a escolha de uma cidade de origem e uma cidade de destino e, em seguida, buscar todos os caminhos entre elas e exibí-los, além de exibir separadamente o menor dos caminhos:



Relacione aqui as cidades do percurso com a menor distância

Relacione aqui todos os percursos encontrados (um percurso por linha). No label "Km do caminho selecionado" informe esse título e a distância percorrida no percurso que o usuário selecionar no DataGridView

Ignore os numericUpDown de distância, Tempo e Custo, pois não serão usados agora.

Use a técnica de Backtracking com Pilha para encontrar todos os caminhos entre a cidade de origem e a de destino, listando cada caminho no dataGridView horizonta inferior conforme é encontrado.

Ao final da busca de todos os caminhos, relacione o menor caminho no DataGridView vertical, à esquerda do formulário.

O arquivo cidades.txt deve ser lido e ter seus dados encapsulados no objeto da classe Cidade e esses objetos devem ser armazenados em um **vetor de cidades**. Esse vetor servirá para identificarmos o índice de cada cidade para indexar a matriz de adjacências, explicada abaixo.

O arquivo GrafoTremEspanhaPortugal.txt contém os dados de ligações diretas entre as cidades. Esse arquivo deve ser lido e a distância entre cada cidade deve ser armazenada em uma matriz

de adjacências, onde o índice da cidade de origem determinará o número da linha dessa matriz e o índice da cidade de destino determinará o número da coluna na matriz, como vemos na figura abaixo.

Nela, temos o vetor de cidades à esquerda, na vertical e a matriz de adjacências à direita. Obviamente a matriz de adjacências não possui uma coluna e uma linha com os nomes das cidades. Esses nomes estão mostrados apenas para explicar o mapeamento entre índice da cidade no vetor e seu índice de linha ou de coluna na matriz de adjacências.

		0	1	2	3	4	5
		Albacete	Alicante	Bilbao	Guarda	Leon	Nilce
0	Albacete						
1	Alicante						
2	Bilbao						
3	Guarda						
4	Leon						
5	Nilce						

IMPORTANTE

- Inicie o desenvolvimento da primeira parte do projeto (esta parte) rapidamente, não deixe para a última hora
- Instruções adicionais para a parte de busca de caminhos serão entregues posteriormente
- Tenha como meta que esta primeira parte fique pronta até 30/05/2023 (21 dias)
- Quando receber a descrição da segunda parte, inicie imediatamente seu desenvolvimento.
- Trabalho feito **em dupla;**
- Desenvolver em C# no Visual Studio, em Windows Forms;
- Comentar adequadamente o programa e o código programado;
- Nomear os identificadores de forma adequada;
- No início dos arquivos fonte, digitar comentário com os RAs e nomes dos alunos;
- **Relatório de desenvolvimento** deve ser feito num arquivo cujo nome é: RA1_RA2_RelProj2.PDF (exemplo: 22101_22192_RelProj2ED.pdf). Deve conter imagens da execução da busca de caminhos;
- O relatório **deve** ser entregue em formato **PDF**;
- Entrega do projeto completo (as duas partes): **20/06/2023 (43 dias)**, pelo Google Classroom
- Material a ser entregue: pasta **do projeto, arquivos de dados e PDF compactados em um único arquivo, cujo nome será** RA1_RA2_Proj2.rar (22101_22192_Proj2.rar, por exemplo).