

Gabarito da AD2 de Programação I

Rio de Janeiro, 3 de novembro de 2011

1. Funcionamento geral

O sistema proposto na AD2 deve realizar o controle de passaportes através de três formulários: um formulário principal contendo campos de consulta um outro que apresentará os resultados dela e um terceiro que deverá receber os dados de viagem do passageiro. Para as explicações desse gabarito, foi escolhido operar com arquivos, mas comentários sobre como operar com banco de dados serão feitos. O primeiro formulário do sistema deve ser algo semelhante ao esboço abaixo:

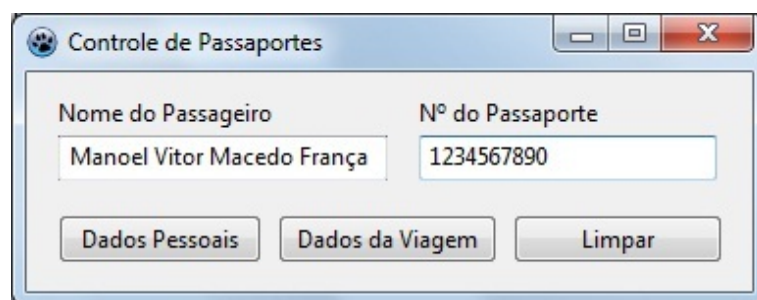
A imagem mostra uma janela de software intitulada "Controle de Passaportes". No topo, há uma barra de título com o ícone de uma pata e botões de minimizar, maximizar e fechar. O corpo da janela contém dois campos de entrada de texto. O primeiro campo, rotulado "Nome do Passageiro", contém o texto "Manoel Vitor Macedo França". O segundo campo, rotulado "Nº do Passaporte", contém o número "1234567890". Abaixo dos campos, há três botões de ação: "Dados Pessoais", "Dados da Viagem" e "Limpar".

Figura 1: Controle de Passaportes

Esse formulário deve conter:

- Dois **TLabeledEdit** para a inserção do nome e do passaporte do passageiro. **TLabeledEdit** é um elemento da aba *Additional* do Lazarus que é uma combinação de **TLabel** e **TEdit**;
- Três **TButtons**: um para abrir o formulário de dados do passageiro (Figura 2), outro para abrir o formulário de dados da viagem (Figura 3) e um terceiro para limpar os dois campos citados acima.

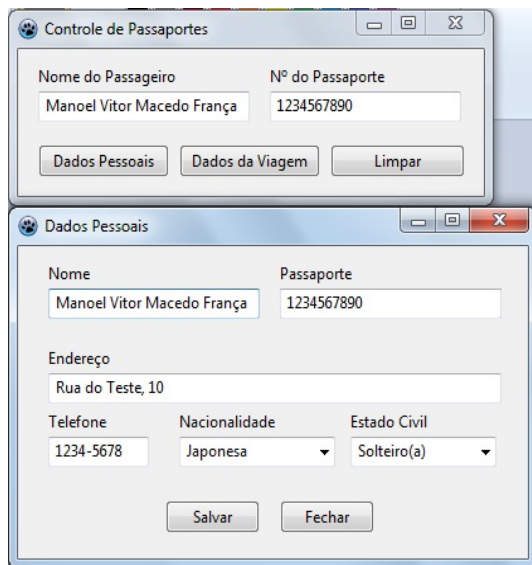


Figura 2: Dados do Passageiro

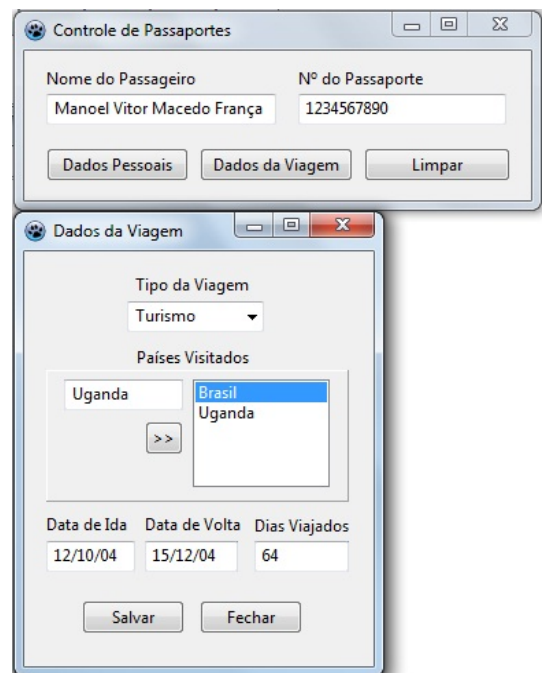


Figura 3: Dados da Viagem

O formulário de dados do passageiro é aberto após o clique no botão correspondente do formulário principal. Uma consulta aos arquivos de armazenamento de dados ou ao banco de dados implementado é realizada e os dados encontrados para o passageiro buscado são carregados. Esse formulário é composto por:

- Dois **TLabelledEdit** com a propriedade 'readonly' recebendo o valor **true**, para receber o nome e passaporte do passageiro consultado;
- Dois **TLabelledEdit** para receber os valores de endereço e telefone, que virão preenchidos caso existam armazenados;
- Um **TLabel** e um **TComboBox**, o primeiro com a propriedade 'caption' recebendo o valor 'Nacionalidade' e o segundo contendo uma lista fixa de nacionalidades para que seja escolhida a do passageiro. O formulário será aberto com a nacionalidade armazenada previamente selecionada, caso exista;
- Um **TLabel** e um **TComboBox**, o primeiro com a propriedade 'caption' recebendo o valor 'Estado Civil' e o segundo contendo uma lista fixa de possíveis estados civis para que seja escolhida a do passageiro. O formulário será aberto com o estado civil armazenado previamente selecionado, caso exista;
- Dois **TButton**, o primeiro salvará todos os dados presentes nesse formulário em arquivo ou em banco e o segundo limpará todos os campos exceto os provenientes do formulário de consulta.

Por fim, o formulário de dados da viagem é aberto após o clique no botão correspondente do formulário principal. Esse formulário é composto por:

- Um **TLabel** e um **TComboBox**, o primeiro com a propriedade ‘caption’ recebendo o valor ‘Tipo da Viagem’ e o segundo contendo uma lista fixa de possíveis tipos de viagem para serem escolhidos;
- Um **TPanel** para agrupar as componentes relacionadas aos países, descritas abaixo;
- Um **TEdit**, um **TButton** e um **TListBox** para, respectivamente: escrever um novo país a ser inserido, incluir o país digitado na lista de países e para guardar e manter visível essa lista;
- Dois **TLabel** e dois **TMaskEdit**, para a inserção das datas de ida e de volta: os **TLabel** possuirão as propriedades ‘caption’ recebendo os valores ‘Data de Ida’ e ‘Data de Volta’ e os **TMaskEdit** serão responsáveis pelo armazenamento de cada uma das datas no formato DD/MM/AA;
- Um **TLabel** e um **TEdit** com a propriedade ‘readonly’ recebendo o valor **true**, o primeiro para rotular o segundo com a “caption” recebendo o valor ‘Dias Viajados’ e o segundo para a visualização da quantidade de dias viajados. O cálculo desse campo será feito a cada vez que houver uma mudança nas datas de ida ou de volta e portanto, serão criadas *procedures* para os eventos *onChange* de cada uma das duas **TMaskEdit** responsáveis pelas datas;
- Dois **TButton**, o primeiro salvará todos os dados presentes nesse formulário em arquivo ou em banco e o segundo limpará todos os campos.

2. Armazenamento dos dados

Como dito anteriormente, a AD2 deve ser capaz de lidar com persistência de dados, em arquivo ou usando um banco de dados a escolher. Para isso, primeiramente, devem ser estipuladas as entidades envolvidas no sistema e as relações entre elas. A estrutura de dados projetada para o sistema descrito no gabarito é mostrada na figura abaixo.

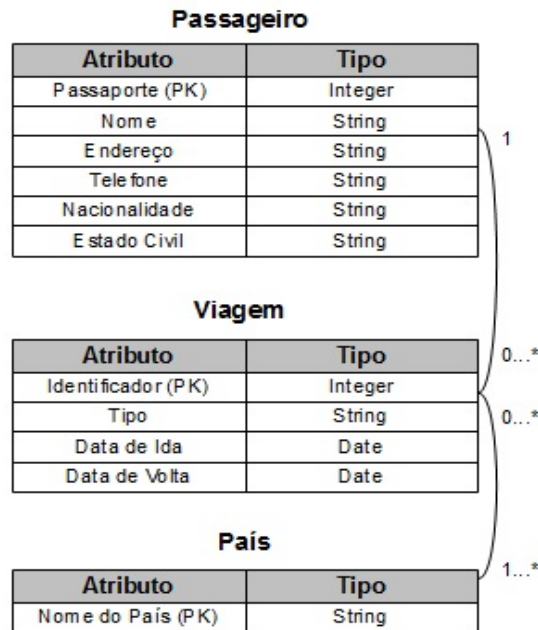


Figura 4: Estrutura dos dados

Observações sobre o esboço acima:

- **PK** refere-se ao identificador da entidade, conhecido como **chave primária**. Note que nenhum dos campos pedidos para representar a viagem identifica-a unicamente e por isso, foi necessário criar um identificador de viagem;
- Cada relacionamento possui símbolos em suas extremidades que representam as quantificações. Uma entidade pode se relacionar com 0, 1 ou muitas instâncias de uma outra entidade relacionada com ela. Os símbolos possíveis são descritos abaixo:
 - *1*: se no final de um relacionamento de uma entidade A com uma entidade B esse símbolo aparecer, uma instância de A pode se relacionar com apenas uma instância de B;
 - *0...1*: se no final de um relacionamento de uma entidade A com uma entidade B esse símbolo aparecer, uma instância de A pode se relacionar com apenas uma instância de B, mas esse relacionamento é opcional;
 - *0...**: se no final de um relacionamento de uma entidade A com uma entidade B esse símbolo aparecer, uma instância de A pode se relacionar com nenhuma, uma ou mais de uma instância de B;
 - *1...**: se no final de um relacionamento de uma entidade A com uma entidade B esse símbolo aparecer, uma instância de A pode se relacionar com uma ou mais de uma instância de B;

- Cada relação que envolve uma instância relacionada com múltiplas outras (ou seja, uma extremidade possui **1** ou **0..1** e a outra, **0...*** ou **1...***), a entidade com a extremidade de multiplicidade maior do que 1 possuirá um atributo que identificará a instância do outro lado da relação à qual suas instâncias se relacionam. Esse atributo é conhecido com **chave estrangeira**. Quando a base usada for apresentada, isso ficará mais claro.
- Relações de muitas instâncias para muitas instâncias (ou seja, de **0...*** ou **1...*** para **0...*** ou **1...***) possuirão uma entidade intermediária que representará as múltiplas relações entre as entidades através de instâncias de suas chaves primárias (no caso do sistema da AD2, isso ocorre com as entidades ‘Viagem’ e ‘País’). Quando a base for apresentada, isso também ficará mais claro.

Como foi dito no início, a forma de armazenamento escolhida foi através de arquivos. Para se fazer isso, é preciso representar a estrutura do esboço acima nos arquivos de alguma forma. A maneira mais intuitiva é representar cada entidade como um arquivo, cada instância como uma linha e cada atributo como um trecho da linha. Por fim, para dividir um atributo do outro, um caracter separador, que não pode ocorrer dentro de um atributo, precisa ser usado. Nessa aplicação, foi escolhido o ponto-e-vírgula(;). Cada arquivo será apresentado abaixo, junto com comentários explicativos:

```

1 1234567890;Manoel Vitor Macedo França;Rua Do Teste, 10;1234-5678;Japonesa;Solteiro(a)
2 0000000000;Fulano De Tal;Rua Do Teste, 11;8765-4321;Alemã;Divorciado(a)

```

Figura 5: Arquivo da entidade ‘Passageiro’

Como pode ser visto, existem duas instâncias nessa entidade. A ordem dos atributos é a mesma do esboço apresentado anteriormente.

```

1 1;Turismo;10/11/2010;21/02/2011;1234567890
2 2;Estudos;21/02/2001;25/05/2001;1234567890
3 3;Asilo;31/01/2005;05/10/2007;0000000000

```

Figura 6: Arquivo da entidade ‘Viagem’

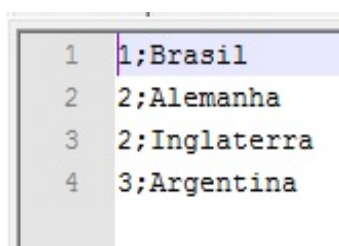
Nessa entidade, existem três instâncias. Note que a falta de um atributo que identifique unicamente uma viagem fez necessária a criação de um número identificador, que é o primeiro atributo. Pode-se notar também um atributo extra ao final de cada instância, o passaporte do passageiro relacionado. Isso ocorre devido ao que foi explicado acima: como a relação entre ‘Passageiro’ e ‘Viagem’ é de (1) para (0...*), a entidade que possui extremidade de multiplicidade maior do que 1, que é a ‘Viagem’, possuirá uma chave estrangeira referenciando uma instância de ‘Passageiro’. Como a chave identificadora dessa entidade é o passaporte, ele foi usado. Com relação à relação com a entidade ‘País’, como dito acima, um arquivo separado será usado.



1	Brasil
2	Alemanha
3	Itália
4	Grécia
5	Estados Unidos
6	Inglaterra
7	Argentina

Figura 7: Arquivo da entidade ‘País’

Nessa entidade, existem sete instâncias. Como dito acima, um arquivo separado representará a relação dessa entidade com ‘Viagem’. Ele encontra-se abaixo.



1	1;Brasil
2	2;Alemanha
3	2;Inglaterra
4	3;Argentina

Figura 8: Arquivo de relacionamento ‘Viagem’ – ‘País’

Essa entidade não representa diretamente nenhum elemento do sistema. Ela serve apenas para mapear viagens com os seus países. Para isso, instâncias foram criadas contendo como atributos as chaves primárias de cada entidade em questão. Em caso de se estar usando um banco de dados, essa entidade seria uma tabela, como as outras descritas anteriormente, mas contendo apenas chaves estrangeiras de ‘Viagem’ e ‘País’. Sua chave primária seria a união dessas duas chaves.