

Gabarito AD2 - 2009/2

23 de novembro de 2009

The image shows a screenshot of a software application window titled "Oficina". The window has a menu bar with "Arquivo" and "Operações". Below the menu bar, there are several input fields and a dropdown menu. The "Marca" field contains "Fiat" with a small dropdown arrow. The "Modelo" field is a dropdown menu with "Uno" selected and "Brasilia" visible below it. The "Placa" field contains "CDF1010". The "Proprietario" field contains "ana". The "Data de Entrada" field is a date picker showing "31/05/2007". The "Descrição do Defeito" field contains "perda total".

QUESTÃO 1:

Figura 1: Formulário Oficina

Funcionamento geral do programa:

- O projeto deve abrir arquivos (.txt ou outro tipo) ao clicar no menu “Abrir-Arquivo” usando um componente TOpenDialog.

O arquivo que contém as informações sobre os carros podem ter uma formatação específica que ajude a reter as informações; por exemplo, as informações sobre cada carro no arquivo podem ser guardadas em uma ordem específica (modelo, placa, proprietário, data de entrada e descrição do defeito).

As informações podem ser salvas usando 5 TStringLists: Modelos, placas, proprietários, datas e descrições. Assim, obtêm-se a propriedade de que o carro cujo modelo está guardado na posição i da stringlist *Modelos*, possui também as demais informações guardadas nas posições i das stringlists *Placas*, *proprietários*, *datas* e *descrições*.

Também para facilitar o armazenamento das informações, o nome da marca referente ao arquivo que está sendo aberto pode ser guardado, dentre outras formas, através das seguintes sugestões:

- Exigir que os arquivos possuam o nome da marca como última linha do arquivo; guardar a última linha em uma string *NomeMarcaAberta*;
- Extrair o nome da marca através do *filename* do arquivo aberto.

O filename também deve ser guardado em uma variável, a fim de que as informações sejam posteriormente salvas no mesmo arquivo. Para guardar as informações dos carros nas StringLists, usa-se o código abaixo, após executar o TOpenDialog:

```
if (dlgAbrir.execute) then
begin
    filename := dlgAbrir.filename;
    LinhasArquivo.LoadFromFile(filename);
    for i := 0 to LinhasArquivo.Count - 2 do
    begin
        if ((i mod 5) = 0) then
        begin
            modelos.add(linhasarquivo[i]);
        end;
        if ((i mod 5) = 1) then
        begin
            placas.add(linhasarquivo[i]);
        end;
        if ((i mod 5) = 2) then
        begin
            proprietarios.add(linhasarquivo[i]);
        end;
        if ((i mod 5) = 3) then
        begin
            datas.add(linhasarquivo[i]);
        end;
        if ((i mod 5) = 4) then
        begin
            descricoes.add(linhasarquivo[i]);
        end;
    end;
    //guarda caminho do arquivo
```

```

NomeArquivoAberto := filename;
//guarda marca aberta
MarcaAberta := linhasarquivo[LinhasArquivo.Count - 1];
end;

```

- Para salvar o arquivo, deve ser feita a operação contrária: remontar as stringLists que guardam as informações dos carros em uma única StringList e usar o método *SaveToFile*. Para isto, exige-se que saibamos qual o arquivo aberto (manter o caminho do arquivo guardado ao abri-lo).

```

linhasArquivo.Clear;
for i := 0 to modelos.Count - 1 do
begin
    linhasarquivo.Add(modelos[i]);
    linhasarquivo.Add(placas[i]);
    linhasarquivo.Add(proprietarios[i]);
    linhasarquivo.Add(datas[i]);
    linhasarquivo.Add(descricoes[i]);
end;
linhasarquivo.add(MarcaAberta); //mantem formatacao do arquivo
linhasarquivo.SaveToFile(NomeArquivoAberto);

```

- Para inserir um novo carro, o aluno poderia programar de forma a exigir que somente carros da marca cujo arquivo está aberto possam ser inseridos, indicando esta exigência ao usuário através do atributo Enabled = false no campo em que a marca aparece no formulário.

The image shows a Windows-style dialog box titled "Incluir Novo". It contains a form with the following fields and values:

- Marca:** Ford
- Modelo:** Ranger
- Placa:** RAN1234
- Proprietario:** Zé das Couves
- Data de Entrada:** 12 / 11 / 2009
- Descrição do Defeito:** pneu furado

At the bottom of the form are two buttons: "Cancelar" and "Salvar".

Figura 2: Formulário de Inserção de Novo Carro

O carro inserido é adicionado à última posição de todas as StringLists que guardam os carros, usando o método *add* das Stringlists.

- Para modificar um proprietário ou uma descrição de defeito, o aluno poderia usar a propriedade de que há o mesmo índice nas StringList para atributos de um mesmo carro. O aluno poderia exigir que um carro esteja selecionado na ListBox, para então modificar somente o tal índice selecionado na StringList adequada. O aluno poderia usar um dialog como uma inputbox, por exemplo:
- Para procurar um carro, o aluno poderia programar de forma a exigir que somente carros da marca cujo arquivo está aberto possam ser procurados, indicando esta exigência ao usuário através do atributo `Enabled = false` no campo em que a marca aparece no formulário.

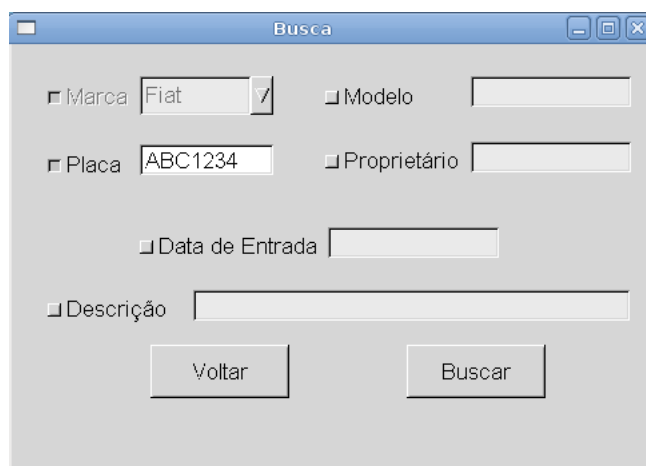


Figura 3: Formulário de Busca

Para a busca, novamente usa-se a propriedade de que há o mesmo índice nas StringList para atributos de um mesmo carro. Com isto, é necessário somente procurar os campos indicados pelo usuário e guardar os índices das entradas que casam com os parâmetros de busca.

O formulário foi feito de forma a que o usuário tenha que clicar em uma checkbox relativa ao campo pelo qual deseja buscar a entrada. O TEdit referente ao campo somente é liberado para a edição (`enabled = true`) quando esta checkbox está assinalada (`checked = true`). Usando este modelo de formulário, para verificar se uma

entrada de um carro (guardada na posição i das StringLists) é igual ao dados inseridos, usa-se a seguinte função:

```
function TFrmBusca.match(i : integer) : boolean;
begin
    result := true;
    if ((cbxModelo.Checked) and (txtModelo.Text <> unit1.modelos[i])) then
        result := false;
    if ((cbxPlaca.Checked) and (txtPlaca.Text <> unit1.placas[i])) then
        result := false;
    if ((cbxData.Checked) and (txtData.Text <> unit1.datas[i])) then
        result := false;
    if ((cbxProprietario.Checked) and (txtProprietario.Text <> unit1.proprietarios[i])) then
        result := false;
    if ((cbxDescricao.Checked) and (txtDescricao.Text <> unit1.descricoes[i])) then
        result := false;
    end;
end;
```

Usando esta função para todos os carros (ou seja, com parâmetro variando entre 0 e $modelos[i].Count - 1$), obtêm-se todas as ocorrências das propriedades inseridas como chave de procura.

The screenshot shows a window titled 'Oficina 2' with a menu bar containing 'Arquivo' and 'Editar'. The main area contains several input fields and a list box. The 'Marca' field contains 'Ford'. The 'Modelo' field is open, showing a list with 'Focus', 'Ka', 'Fiesta', and 'Escort'. Below these is a toolbar with icons for navigation (left, right, first, last) and editing (add, delete, edit, undo, redo). The 'Placa' field contains 'GAY2424'. The 'Proprietário' field contains 'Clodovil'. The 'Data de entrada' field contains '24/12/1924'. The 'Descrição do Defeito' field contains 'pintura arranhada'.

QUESTÃO 2:

Figura 4: Formulário Oficina

Funcionamento geral do programa:

O programa possui o mesmo funcionamento do anterior, porém usando um banco de dados para guardar as informações sobre os carros. Segue um exemplo das tabelas de um banco de dados para guardar as informações dos carros:

- Marca(**codmarca**, nome)
- Modelo(**codmodelo**, *codmarca*, nome)

Figura 5: Formulário Novo

- Veiculo(**codveiculo**, *codmodelo*, proprietario, placa, data, defeito)

As chaves primárias estão em negrito, as chaves estrangeiras em itálico e o campo “placa” da tabela Veículo é único para todos os carros (“unique” em Mysql).

Para listar as marcas na TCombobox, utiliza-se a query “`select nome from marca`”.

Para fazer a conexão entre a seleção de uma marca na Combobox e a listagem de modelos com esta marca na TListBox, a query a ser executada é:

```
'select distinct modelo.nome as nome_marca from veiculo, modelo, marca
where marca.nome = ' + ''' + cbxMarcas.Text + ''' + ' and modelo.codmarca = marca.codmarca ' +
' and veiculo.codmod = modelo.codmod';
```

Ao selecionar um modelo na TListBox, através do DBNavigator é possível navegar por todos os carros que contém mesma marca e modelo através desta query:

```
'select data, placa, proprietario, defeito from veiculo, modelo
where modelo.codmod = veiculo.codmod' + ' and modelo.nome = ' + '''
+ LstModelos.Items[LstModelos.ItemIndex] + ''';
```

Para alterar o proprietário ou a descrição do defeito de um carro cujos atributos estão sendo exibidos nos TEdits, é usado o comando update:

```
'update veiculo set proprietario = ' + ''' + novoproprietario + ''' + ' where placa = ' + ''' +
placa + ''';
'update veiculo set defeito = ' + ''' + defeito + ''' + ' where placa = ' + ''' +
placa + ''';
```

Figura 6: Formulário Busca

Para buscar um carro através de parâmetros escolhidos por checkboxes no formulário, os seguintes comandos montam a query a ser executada:

```

query := 'select marca.nome as marca, modelo.nome as modelo, ' +
'veiculo.proprietario as proprietario, veiculo.placa as placa, veiculo.data as data, ' +
'veiculo.defeito as defeito from veiculo, marca, modelo where veiculo.codmod = modelo.codmod' +
' and modelo.codmarca = marca.codmarca';

if (cbxMarca.Checked) then
  query := query + ' and marca.nome = ' + ''' + cbbMarca.Text + ''';
if (cbxModelo.Checked) then
  query := query + ' and modelo.nome = ' + ''' + LstModelo.Items[LstModelo.ItemIndex] + ''';
if (cbxPlaca.Checked) then
  query := query + ' and veiculo.placa = ' + ''' + txtPlaca.text + ''';
if (cbxProprietario.Checked) then
  query := query + ' and veiculo.proprietario = ' + ''' + txtProprietario.text + ''';
if (cbxData.Checked) then
  query := query + ' and veiculo.data = ' + ''' + txtData.text + ''';
if (cbxDefeito.Checked) then
  query := query + ' and veiculo.defeito = ' + ''' + txtDefeito.text + ''';

```

Para inserir um novo carro, usa-se o comando 'insert'. Porém, para

inserir um novo item na tabela Veículo, é necessário saber o código do modelo do carro. Para descobri-lo, sabendo que seu nome aparece selecionado na TListBox, é usada a seguinte função:

```
function DescobreCodigoModelo (NomeModelo : string; NomeMarca : string) : string;
begin
    queryModelo.Close;
    queryModelo.sql.text := 'select modelo.codmod from veiculo, modelo, marca where modelo.nome = '
+ ''' + NomeModelo + ''' + ' and marca.nome = ' + '''
+ NomeMarca + ''' + ' and modelo.codmarca = marca.codmarca';
    queryModelo.Open;
    result := queryModelo.fieldbyname('codmod').asString;
end;
```

Usando esta função, o novo carro é inserido na tabela usando a query:

```
'insert into veiculo(codmod, placa, data, proprietario, defeito)
values(' + codmod + ', ''' + txtPlaca.text + ''', ''' + txtData.text + ''', '''
+ txtProprietario.text + ''', ''' + memDefeito.text + ''' + ')';
```

Onde

```
codmod := DescobreCodigoModelo(ListBox1.Items[ListBox1.ItemIndex], cbbModelo.Text);
```