

Fundação CECIERJ - Vice Presidência de Educação Superior a Distância  
Curso de Tecnologia em Sistemas de Computação  
Disciplina: Programação I  
AD2 - 2º semestre de 2012.

GABARITO

### 1. Solução

A Figura 1 é composta por três botões. O botão “Buscar”, verifica se a placa informada está cadastrada e, caso positivo, preenche o Formulário Secundário (Figura 2) com os valores armazenados, caso negativo, envia a placa informada no Formulário Principal para o Formulário Secundário, deixando os demais campos em branco. O botão “Limpar”, apenas limpa o campo da placa. O botão “Pesquisar por Fabricante e Ano”, abre o Terceiro formulário, indicado na Figura 3.



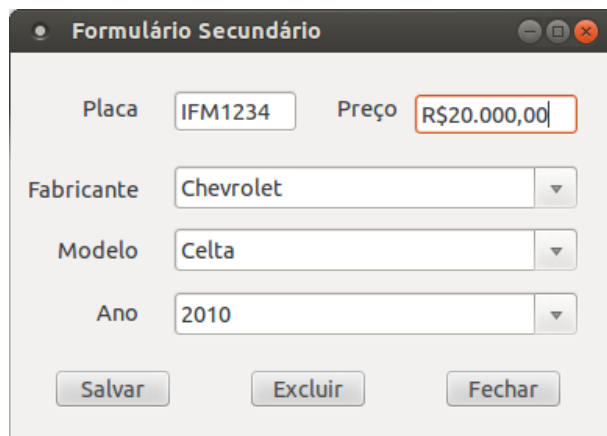
Agência de Automóveis

Placa IFM1234

Buscar Limpar Pesquisar por Fabricante e Ano

Figura 1: Formulário Principal.

O Formulário Secundário, contém três TComboBox que foram previamente preenchidos com seus respectivos itens. O objetivo desse formulário é incluir (salvar informações associadas a uma nova placa), editar (quando a placa já havia sido cadastrada) e excluir os valores associados a uma placa.



Formulário Secundário

Placa IFM1234 Preço R\$20.000,00

Fabricante Chevrolet

Modelo Celta

Ano 2010

Salvar Excluir Fechar

Figura 2: Formulário Secundário.

No Terceiro Formulário, é possível realizar pesquisas por fabricante e ano de fabricação. Nesse formulário existe um TListBox utilizado para listar todos os registros retornados do clique do botão “Pesquisar”. Ao clicar sobre um registro do TListBox, o Formulário Secundário será aberto com os valores do registro que foi selecionado.

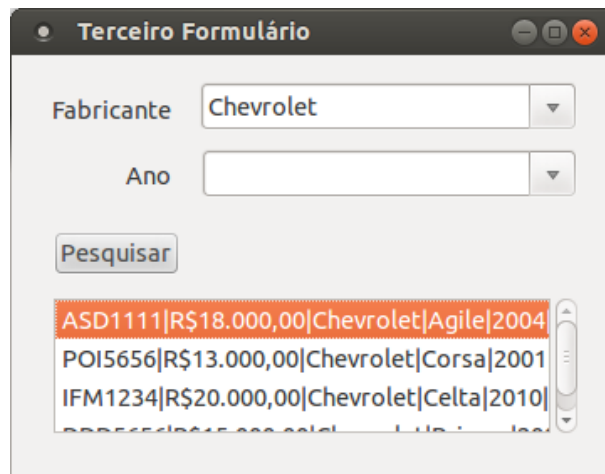


Figura 3: Terceiro Formulário.

Para acessar a linha inteira do TListBox com os valores associados ao registro selecionado, pode-se utilizar o seguinte comando: `ListBox.Items.Strings[ListBox.ItemIndex]`; A partir dessa string é possível separar os valores dos campos com uma pequena variação do `getToken` apresentado no gabarito da AD1. A função `GetToken`, apresentada abaixo, recebe três parâmetros, o primeiro é a string, o segundo é o caractere que separa os valores e o terceiro é um inteiro que indica qual o campo desejado, onde 0 retorna o primeiro campo da string (Placa), 1 retorna o preço, 2 retorna o Fabricante, 3 retorna o Modelo e 4 retorna o Ano.

Implementação da função `GetToken`.

```

1  function TFormPrinc.GetToken (a:String; Sep: Char; dev: integer):String;
2
3  var
4      Token : String;
5      TEnd : Byte;
6      contador : integer;
7  begin
8      contador:=0;
9      while(contador<=dev) do
10         begin
11             TEnd := Length(a);
12             Result := '';
13             TEnd := Pos(Sep, a);
14             if TEnd <> 0 then
15                 begin
16                     Token := Copy( a, 1, TEnd-1);
17                     Delete( a, 1, TEnd);
18                 end
19             else
20                 begin
21                     Token := a;

```

```

22         a := '';
23     end;
24     Result := Token;
25     contador:= contador +1;
26 end;
27 end;

```

## Armazenamento

Os dados deverão ser armazenados em um ou mais arquivos. Uma solução é utilizar um único arquivo de texto para armazenar todos os registros. Para isso, cada linha representa um único registro e os campos desse registro podem ser separados por um símbolo especial, por exemplo, em meu caso, utilizei o caractere “|” para separar os campos. Portanto, os campos de cada registro podem facilmente ser obtidos com o uso da função `GetToken`.

Uma estratégia para trabalhar com arquivo e não ficar o tempo todo acessando o disco é carregar o arquivo texto para uma variável do tipo *TStringList*.

Exemplo de como declarar, carregar e salvar arquivos para *TStringList*:

```

1  var
2      List : TStringList;
3  begin
4      List := TStringList.Create;
5      List.LoadFromFile('dados.txt');
6
7      List.SaveToFile('dados.txt');
8  end

```

Outros três comandos importantes para o desenvolvimento são:

- Apagar uma linha (registro), por exemplo, apagar a primeira linha:  
List.Delete(0);
- Inserir um registro na linha 5:  
List.Insert(4, novoRegistro);
- Obter todo o conteúdo da linha 4:  
registro := List.Strings[3];