CENTRO UNIVERSITÁRIO AUGUSTO MOTTA – UNISUAM
CURSO CIÊNCIA DA COMPUTAÇÃO

Projeto da Disciplina: Laboratório de Desenvolvimento de Sistemas I

Times CBF

Professor: André Rezende

Aluno:

Leonardo José Garcia Maia

Rio de Janeiro, 09 de Novembro de 2008.

Sumário

1. Codificação Completa (*Unit's*)

4.1. Projeto

```
program ProjectTimes;

uses
  Forms,
  UnitPrincipal in 'UnitPrincipal.pas' {FormPrincipal},
  UnitCadastrarTime in 'UnitCadastrarTime.pas' {FormCadastrarTime},
  UnitAutorInformacoes in 'UnitAutorInformacoes.pas' {FormAutorInformacoes},
  UnitValorCache in 'UnitValorCache.pas' {FormValorCache},
  UnitVerificarConfirmacoes in 'UnitVerificarConfirmacoes.pas' {FormVerificarConfirmacoes},
  UnitModificarTime in 'UnitModificarTime.pas' {FormModificarTime},
  UnitApagarTime in 'UnitApagarTime.pas' {FormApagarTime},
  UnitBuscaNome in 'UnitBuscaNome.pas' {FormBuscaNome},
  UnitBuscaCodigo in 'UnitBuscaCodigo.pas' {FormBuscaCodigo},
  UnitRelatorio in 'UnitRelatorio.pas' {FormRelatorio};
{$R *.res}
begin
  Application.Initialize;
  Application.Title := 'Times CBF';
  Application.CreateForm(TFormPrincipal, FormPrincipal);
  Application.CreateForm(TFormCadastrarTime, FormCadastrarTime);
  Application.CreateForm(TFormAutorInformacoes, FormAutorInformacoes);
  Application.CreateForm(TFormValorCache, FormValorCache);
  Application.CreateForm(TFormVerificarConfirmacoes, FormVerificarConfirmacoes);
  Application.CreateForm(TFormModificarTime, FormModificarTime);
  Application.CreateForm(TFormApagarTime, FormApagarTime);
  Application.CreateForm(TFormBuscaNome, FormBuscaNome);
  Application.CreateForm(TFormBuscaCodigo, FormBuscaCodigo);
```

```
  Application.CreateForm(TFormRelatorio, FormRelatorio);
  Application.Run;
end.
```

4.2. Principal


```
unit UnitPrincipal;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, Menus, jpeg;

type
  TFormPrincipal = class(TForm)
    MainMenu1: TMainMenu;
    ime1: TMenuItem;
    Modificar1: TMenuItem;
    Apagar1: TMenuItem;
    Cadastrar1: TMenuItem;
    Sair1: TMenuItem;
    Busca1: TMenuItem;
    Nome1: TMenuItem;
    Codigo1: TMenuItem;
```

Cach1: TMenuItem;

Valor1: TMenuItem;

Confirmao1: TMenuItem;

Verificar1: TMenuItem;

Informaes1: TMenuItem;

Autor1: TMenuItem;

PopupMenu1: TPopupMenu;

ime2: TMenuItem;

Busca2: TMenuItem;

Cach2: TMenuItem;

Confirmao2: TMenuItem;

Informaes2: TMenuItem;

Cadastrar2: TMenuItem;

Modificar2: TMenuItem;

Apagar2: TMenuItem;

Sair2: TMenuItem;

Autor2: TMenuItem;

Verificar2: TMenuItem;

Valor2: TMenuItem;

Nome2: TMenuItem;

Cdigo1: TMenuItem;

Image1: TImage;

Sair3: TMenuItem;

Sair4: TMenuItem;

N1: TMenuItem;

Relatrio1: TMenuItem;

PrVisualizao1: TMenuItem;

Impresso1: TMenuItem;

Relatrio2: TMenuItem;

```
    PrVisualizao2: TMenuItem;
    Impresso2: TMenuItem;
    procedure Sair1Click(Sender: TObject);
    procedure Cadastrar1Click(Sender: TObject);
    procedure Autor1Click(Sender: TObject);
    procedure Valor1Click(Sender: TObject);
    procedure Verificar1Click(Sender: TObject);
    procedure Modificar1Click(Sender: TObject);
    procedure Apagar1Click(Sender: TObject);
    procedure Nome1Click(Sender: TObject);
    procedure Codigo1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure PrVisualizao1Click(Sender: TObject);
    procedure Impresso1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  FormPrincipal: TFormPrincipal;
implementation
uses UnitCadastrarTime, UnitAutorInformacoes, UnitValorCache,
  UnitVerificarConfirmacoes, UnitModificarTime, UnitApagarTime,
  UnitBuscaNome, UnitBuscaCodigo, UnitRelatorio;
{$R *.dfm}
procedure TFormPrincipal.Sair1Click(Sender: TObject);
begin
```

```
  Close; //Fecha o formulário

end;

procedure TFormPrincipal.Cadastrar1Click(Sender: TObject);

begin

      FormCadastrarTime.showmodal; //Chama o formulário Cadastrar Time de forma a bloquear o
acesso ao formulário principal

end;

procedure TFormPrincipal.Autor1Click(Sender: TObject);

begin

      FormAutorInformacoes.showmodal; //Chama o formulário Autor Informações de forma a
bloquear o acesso ao formulário principal

end;

procedure TFormPrincipal.Valor1Click(Sender: TObject);

begin

      FormValorCache.showmodal; //Chama o formulário Valor Cachê de forma a bloquear o
acesso ao formulário principal

end;

procedure TFormPrincipal.Verificar1Click(Sender: TObject);

begin

      FormVerificarConfirmacoes.showmodal; //Chama o formulário Verificar Confirmações de
forma a bloquear o acesso ao formulário principal

end;

procedure TFormPrincipal.Modificar1Click(Sender: TObject);

begin

      FormModificarTime.showmodal; //Chama o formulário Modificar Time de forma a bloquear o
acesso ao formulário principal

end;

procedure TFormPrincipal.Apagar1Click(Sender: TObject);

begin

  FormApagarTime.showmodal; //Chama o formulário Apagar Time de forma a bloquear o acesso
ao formulário principal
```

```
end;

procedure TFormPrincipal.Nome1Click(Sender: TObject);

begin

  FormBuscaNome.showmodal; //Chama o formulário Busca Nome de forma a bloquear o acesso ao formulário principal

end;

procedure TFormPrincipal.Codigo1Click(Sender: TObject);

begin

  FormBuscaCodigo.showmodal; //Chama o formulário Busca Código de forma a bloquear o acesso ao formulário principal

end;

procedure TFormPrincipal.FormCreate(Sender: TObject);

begin

    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que seja movido o formulário

end;

procedure TFormPrincipal.PrVisualizao1Click(Sender: TObject);

begin

  FormRelatorio.QuickRepTimes.Preview; //Chama o formulário Relatório para ser visualizadas as informações

end;

  procedure TFormPrincipal.Impresso1Click(Sender: TObject);

begin

  FormRelatorio.QuickRepTimes.Print; //Chama o formulário Relatório para ser enviado as informações diretamente para impressora

end;

end.
```

4.3. Cadastrar Time


```
unit UnitCadastrarTime;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, jpeg, ExtCtrls, StdCtrls, Buttons, Mask, DBCtrls, DB, ADODB;

type
  TFormCadastrarTime = class(TForm)
    BitBtnCadastrar: TBitBtn;
    BitBtnLimpar: TBitBtn;
    BitBtnSair: TBitBtn;
    Label3: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
```

```
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Image1: TImage;
    TblTimes: TADOTable;
    DBEdit2: TDBEdit;
    DataSource1: TDataSource;
    DBEdit1: TDBEdit;
    DBEdit4: TDBEdit;
    DBEdit5: TDBEdit;
    DBEdit6: TDBEdit;
    DBEdit7: TDBEdit;
    DBComboBox1: TDBComboBox;
    TblTimesCodigo: TWideStringField;
    TblTimesNomeDoTime: TWideStringField;
    TblTimesUltimoTitulo: TWideStringField;
    TblTimesCorCamisa: TWideStringField;
    TblTimesCidadeOrigem: TWideStringField;
    TblTimesEstadoOrigem: TWideStringField;
    TblTimesPricipalTorcida: TWideStringField;
    TblTimesPossuiEstadio: TWideStringField;
    TblTimesDataDeInscricao: TDateTimeField;
    DBEdit3: TDBEdit;
    DBRadioGroup1: TDBRadioGroup;
    procedure BitBtnSairClick(Sender: TObject);
    procedure BitBtnCadastrarClick(Sender: TObject);
    procedure BitBtnLimparClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
```

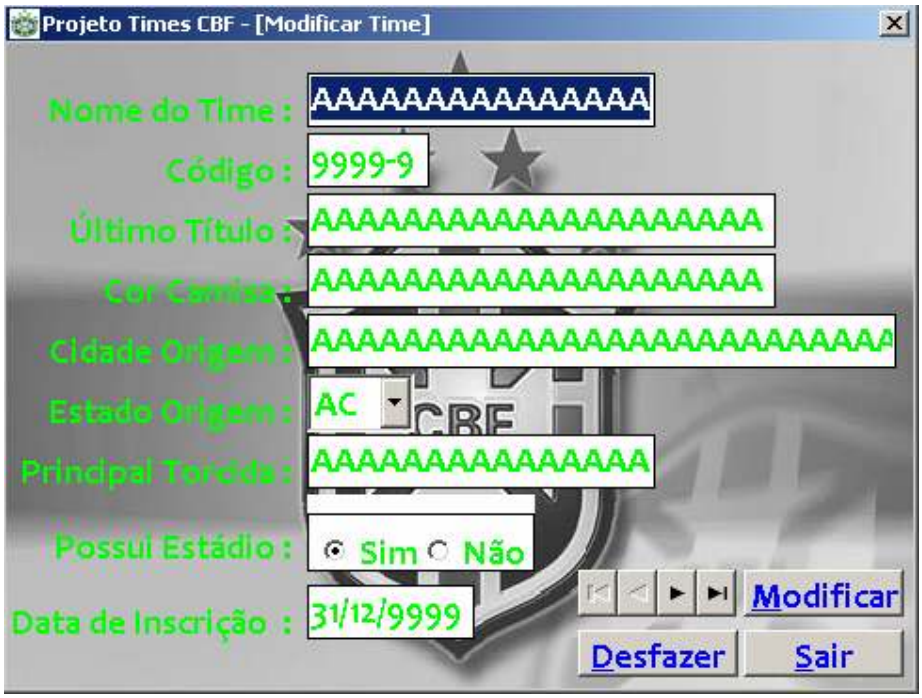
```
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  FormCadastrarTime: TFormCadastrarTime;
implementation
{$R *.dfm}
procedure TFormCadastrarTime.BitBtnSairClick(Sender: TObject);
begin
  Close; //Fecha o formulário
end;
procedure TFormCadastrarTime.BitBtnCadastrarClick(Sender: TObject);
begin
  If (DBEdit1.Text='') Then //Verifica se os Edit de Nome do Time esta vazio
    Begin
      ShowMessage('Favor verificar se o Nome do time foi digitado!'); //Apresenta um mensagem
para que se verifique se foi digitado o Nome do time
      DBEdit1.SetFocus; //Coloca a seleção no Edit
     End
  Else if (DBEdit2.Text='    - ') Then //Verifica se os MaskEdit do Código esta vazio
     Begin
      ShowMessage('Favor verificar se o Código foi digitado!'); //Apresenta um mensagem para que
se verifique se foi digitado o Código
      DBEdit2.SetFocus; //Coloca a seleção no Edit
     End
```

```
  Else

    Begin

          TBLTimes.Post; //Salva o registro atual na base de dados do ADOTable

      ShowMessage('Cadastrado com sucesso!'); //Apresenta um mensagem de que o registro foi
Cadastrado com sucesso

      TBLTimes.Append; //Cria um novo Registro em Branco na base de dados do ADOTable

    End;

end;

procedure TFormCadastrarTime.BitBtnLimparClick(Sender: TObject);

begin

  If (DBEdit1.Text<>'') or (DBEdit2.Text<>'    - ') Then //Verifica se o Edit e o MaskEdit de Nome
do Time e Código estão vazios

  If MessageDlg ('Salvar modificações?', mtConfirmation, mbOkCancel,0) = mrOk //Mensagem na
tela que efetua uma pergunta se deseja salvar as modificações se pressionar ok

  Then

    Begin

       If (DBEdit1.Text='') Then //Verifica se os Edit de Nome do Time esta vazio

         Begin

           ShowMessage('Favor verificar se o Nome do time foi digitado!'); //Apresenta um
mensagem para que se verifique se foi digitado o Nome do time

           DBEdit1.SetFocus; //Coloca a seleção no Edit

         End

       Else if (DBEdit2.Text='    - ') Then //Verifica se os MaskEdit do Código esta vazio

         Begin

           ShowMessage('Favor verificar se o Código foi digitado!'); //Apresenta um mensagem para
que se verifique se foi digitado o Código

           DBEdit2.SetFocus; //Coloca a seleção no Edit

         End

       Else

         Begin

           TBLTimes.Post; //Salva o registro atual na base de dados do ADOTable
```

```
          ShowMessage('Cadastrado com sucesso!'); //Apresenta um mensagem de que o registro
foi Cadastrado com sucesso

          TBLTimes.Append; //Cria um novo Registro em Branco na base de dados do ADOTable

        End;

   End

  Else

   Begin

        TBLTimes.Cancel; //Cancela as modificações feitas na base de dados do ADOTable através
do formulário e limpa os campos

        TBLTimes.Append; //Cria um novo Registro em Branco na base de dados do ADOTable

   End;

end;

procedure TFormCadastrarTime.FormShow(Sender: TObject);

begin

  TBLTimes.Active:=True; //Ativa a base de dados do ADOTable

  TBLTimes.Append; //Cria um novo Registro em Branco na base de dados do ADOTable

  DBEdit1.SetFocus; //Coloca a seleção no Edit

end;

procedure TFormCadastrarTime.FormClose(Sender: TObject;

  var Action: TCloseAction);

begin

  If (DBEdit1.Text<>'') or (DBEdit2.Text<>'    - ') Then //Verifica se o Edit e o MaskEdit de Nome
do Time e Código estão vazios

  If MessageDlg ('Salvar modificações?', mtConfirmation, mbOkCancel,0) = mrOk //Mensagem na
tela que efetua uma pergunta se deseja salvar as modificações se pressionar ok

  Then

   Begin

      If (DBEdit1.Text='') Then //Verifica se os Edit de Nome do Time esta vazio

        Begin

          ShowMessage('Favor verificar se o Nome do time foi digitado!'); //Apresenta um
mensagem para que se verifique se foi digitado o Nome do time
```

```
            DBEdit1.SetFocus; //Coloca a seleção no Edit

            Abort; //Aborta o fechamento do formulário

          End

        Else if (DBEdit2.Text='    - ') Then //Verifica se os MaskEdit do Código esta vazio

          Begin

            ShowMessage('Favor verificar se o Código foi digitado!'); //Apresenta um mensagem para
que se verifique se foi digitado o Código

            DBEdit2.SetFocus; //Coloca a seleção no Edit

            Abort; //Aborta o fechamento do formulário

          End

        Else

          Begin

            TBLTimes.Post; //Salva o registro atual na base de dados do ADOTable

            ShowMessage('Cadastrado com sucesso!'); //Apresenta um mensagem de que o registro
foi Cadastrado com sucesso

          End;

    End;

  TBLTimes.Cancel; //Cancela as modificações feitas na base de dados do ADOTable através do
formulário e limpa os campos

  TBLTimes.Active:=False; //Desativa a base de dados do ADOTable

end;

procedure TFormCadastrarTime.FormCreate(Sender: TObject);

begin

    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND);); //Impede que
seja movido o formulário

end;

end.
```

4.4. Modificar Time



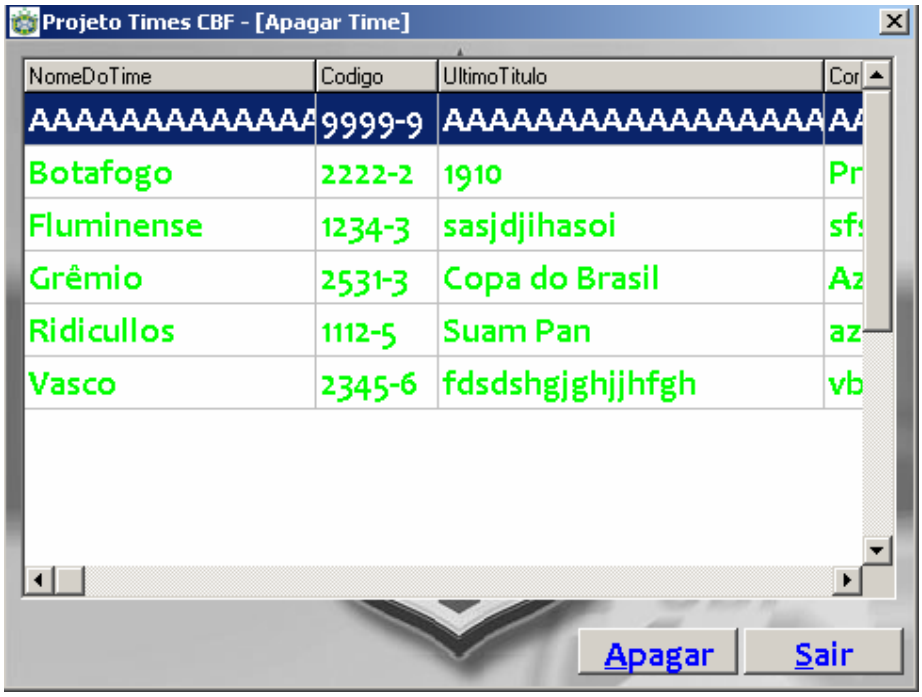

```
unit UnitModificarTime;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, UnitCadastrarTime, DB, ADODB, StdCtrls, ExtCtrls, DBCtrls, Mask,
  Buttons, jpeg;
type
  TFormModificarTime = class(TFormCadastrarTime)
    DBNavigator1: TDBNavigator;
    procedure BitBtnCadastrarClick(Sender: TObject);
    procedure BitBtnSairClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure BitBtnLimparClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
```

```
    { Private declarations }
  public
    { Public declarations }
  end;
var
  FormModificarTime: TFormModificarTime;
implementation
{$R *.dfm}
procedure TFormModificarTime.BitBtnCadastrarClick(Sender: TObject);
begin
  If TBLTimes.State = dsEdit // Se ouve alguma modificação nos Edits
  Then If (DBEdit1.Text='') Then //Verifica se os Edit de Nome do Time esta vazio
    Begin
      ShowMessage('Favor verificar se o Nome do time foi digitado!'); //Apresenta um mensagem
para que se verifique se foi digitado o Nome do time
      DBEdit1.SetFocus; //Coloca a seleção no Edit
    End
  Else if (DBEdit2.Text='    - ') Then //Verifica se os MaskEdit do Código esta vazio
    Begin
      ShowMessage('Favor verificar se o Código foi digitado!'); //Apresenta um mensagem para que
se verifique se foi digitado o Código
      DBEdit2.SetFocus; //Coloca a seleção no Edit
    End
  Else
    Begin
      TBLTimes.Post; //Salva o registro atual na base de dados do ADOTable
      ShowMessage('Moficado com sucesso!');//Apresenta um mensagem de que o registro foi
modificado com sucesso
      TBLTimes.Active:=False; //Desativa a base de dados do ADOTable
      TBLTimes.Active:=True; //Ativa a base de dados do ADOTable
```

```
    End;

end;

procedure TFormModificarTime.BitBtnSairClick(Sender: TObject);

begin

  close; //Fecha o formulário

end;

procedure TFormModificarTime.FormClose(Sender: TObject;

  var Action: TCloseAction);

begin

  If TBLTimes.State = dsEdit // Se ouve alguma modificação nos Edits

    Then If MessageDlg ('Salvar modificações?', mtConfirmation, mbOkCancel,0) = mrOk
//Mensagem na tela que efetua uma pergunta se deseja salvar as modificações se pressionar ok

  Then If (DBEdit1.Text='') Then //Verifica se os Edit de Nome do Time esta vazio

    Begin

      ShowMessage('Favor verificar se o Nome do time foi digitado!'); //Apresenta um mensagem
para que se verifique se foi digitado o Nome do time

      DBEdit1.SetFocus; //Coloca a seleção no Edit

      Abort; //Aborta o fechamento do formulário

    End

  Else if (DBEdit2.Text='    - ') Then //Verifica se os MaskEdit do Código esta vazio

    Begin

      ShowMessage('Favor verificar se o Código foi digitado!'); //Apresenta um mensagem para que
se verifique se foi digitado o Código

      DBEdit2.SetFocus; //Coloca a seleção no Edit

      Abort; //Aborta o fechamento do formulário

    End

  Else

    Begin

      TBLTimes.Post; //Salva o registro atual na base de dados do ADOTable
```

```
      ShowMessage('Modificado com sucesso!');//Apresenta um mensagem de que o registro foi
modificado com sucesso

    End;

  TBLTimes.Cancel; //Cancela as modificações feitas na base de dados do ADOTable através do
formulário e limpa os campos

  TBLTimes.Active:=False; //Desativa a base de dados do ADOTable

end;

procedure TFormModificarTime.BitBtnLimparClick(Sender: TObject);

begin

  If TBLTimes.State = dsEdit Then //Verifica se algo foi alterado no formulário

        If MessageDlg ('Desfazer modificações não salvas?', mtConfirmation, mbOkCancel,0) =
mrOk //Mensagem na tela que efetua uma pergunta se deseja desfazer as modificações não salvas
se pressionar ok

        Then

    Begin

      TBLTimes.Cancel; //Cancela as modificações feitas na base de dados do ADOTable através do
formulário e limpa os campos

      ShowMessage('Moficações desfeitas!'); //Apresenta um mensagem de que o registro não foi
modificado

      DBEdit1.SetFocus; //Coloca a seleção no Edit

    End;

end;

procedure TFormModificarTime.FormShow(Sender: TObject);

begin

  TBLTimes.Active:=True; //Ativa a base de dados do ADOTable

end;

procedure TFormModificarTime.FormCreate(Sender: TObject);

begin

    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que
seja movido o formulário

end; end.
```

4.5. Apagar Time

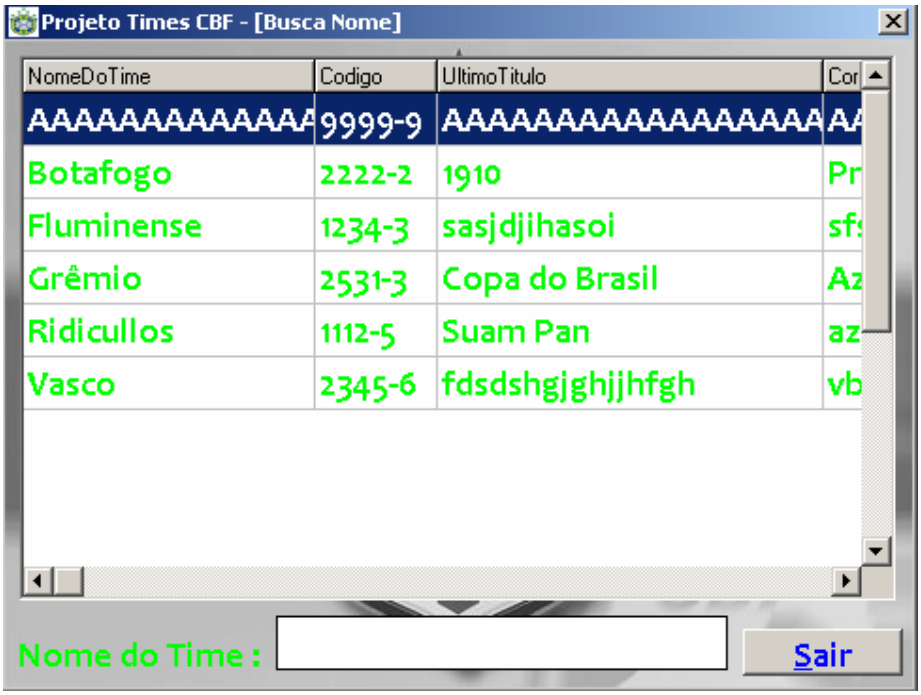



```
unit UnitApagarTime;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, DB, ADODB, StdCtrls, Buttons, jpeg, ExtCtrls;

type
  TFormApagarTime = class(TForm)
    Image1: TImage;
    Image2: TImage;
    BitBtnSair: TBitBtn;
    BitBtnApagar: TBitBtn;
    TblTimes: TADOTable;
    TblTimesNomeDoTime: TWideStringField;
    TblTimesCodigo: TWideStringField;
    TblTimesUltimoTitulo: TWideStringField;
    TblTimesCorCamisa: TWideStringField;
```

```
    TblTimesCidadeOrigem: TWideStringField;
    TblTimesEstadoOrigem: TWideStringField;
    TblTimesPricipalTorcida: TWideStringField;
    TblTimesPossuiEstadio: TWideStringField;
    TblTimesDataDeInscricao: TDateTimeField;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    procedure BitBtnSairClick(Sender: TObject);
    procedure BitBtnApagarClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  FormApagarTime: TFormApagarTime;
implementation
{$R *.dfm}
procedure TFormApagarTime.BitBtnSairClick(Sender: TObject);
begin
  Close; //Fecha o formulário
end;
procedure TFormApagarTime.BitBtnApagarClick(Sender: TObject);
begin
```

```
  If MessageDlg ('Excluir registro?', mtConfirmation, mbOkCancel,0) = mrOk //Mensagem na tela

  que efetua uma pergunta se deseja excluir se pressionar ok

        Then

    Begin

      TBLTimes.Delete; // O registro selecionado na DBgrid é deletado da base de dados do
ADOTable

      ShowMessage('Excluido com sucesso!'); //Apresenta um mensagem de que o registro foi
excluído com sucesso

    End;

end;

procedure TFormApagarTime.FormClose(Sender: TObject;

  var Action: TCloseAction);

begin

  TBLTimes.Active:=False; //Desativa a base de dados do ADOTable

end;

rocedure TFormApagarTime.FormShow(Sender: TObject);

begin

  TBLTimes.Active:=True; //Ativa a base de dados do ADOTable

end;

procedure TFormApagarTime.FormCreate(Sender: TObject);

begin

    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que
seja movido o formulário

end;

end.
```

4.6. Busca Nome

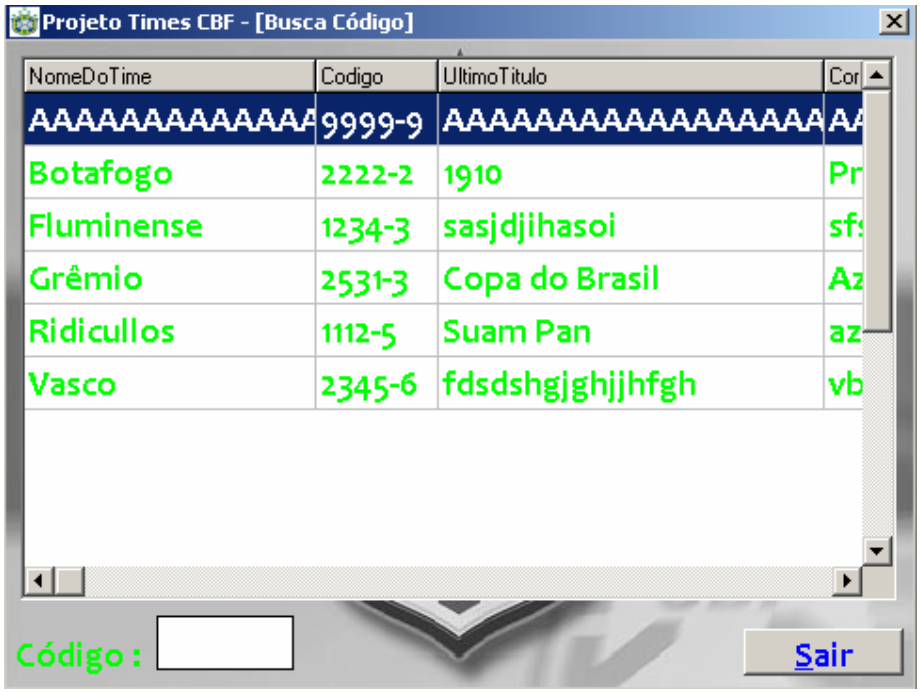



```
unit UnitBuscaNome;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, ADODB, jpeg, ExtCtrls, StdCtrls, Buttons, Grids, DBGrids;

type
  TFormBuscaNome = class(TForm)
    DBGrid1: TDBGrid;
    BitBtnSair: TBitBtn;
    Image2: TImage;
    TblTimes: TADOTable;
    TblTimesNomeDoTime: TWideStringField;
    TblTimesCodigo: TWideStringField;
    TblTimesUltimoTitulo: TWideStringField;
    TblTimesCorCamisa: TWideStringField;
    TblTimesCidadeOrigem: TWideStringField;
```

```
    TblTimesEstadoOrigem: TWideStringField;
    TblTimesPricipalTorcida: TWideStringField;
    TblTimesPossuiEstadio: TWideStringField;
    TblTimesDataDeInscricao: TDateTimeField;
    DataSource1: TDataSource;
    Label3: TLabel;
    EditNome: TEdit;
    procedure BitBtnSairClick(Sender: TObject);
    procedure EditNomeChange(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  FormBuscaNome: TFormBuscaNome;
implementation
{$R *.dfm}
procedure TFormBuscaNome.BitBtnSairClick(Sender: TObject);
begin
  Close; //Fecha o formulário
end;
procedure TFormBuscaNome.EditNomeChange(Sender: TObject);
begin
```

```
  TBLTimes.Locate ('NomeDoTime', EditNome.Text, [loPartialKey]); //Procura na base de dados do
ADOTable o nome do time digitado no Edit, de forma parcial, toda vez que uma letra é digitada

end;

procedure TFormBuscaNome.FormClose(Sender: TObject;

  var Action: TCloseAction);

begin

  EditNome.Clear;

  TBLTimes.Active:=False; //Desativa a base de dados do ADOTable

end;

procedure TFormBuscaNome.FormShow(Sender: TObject);

begin

  TBLTimes.Active:=True; //Ativa a base de dados do ADOTable

end;

procedure TFormBuscaNome.FormCreate(Sender: TObject);

begin

    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que
seja movido o formulário

end;

end.
```

4.7. Busca Código

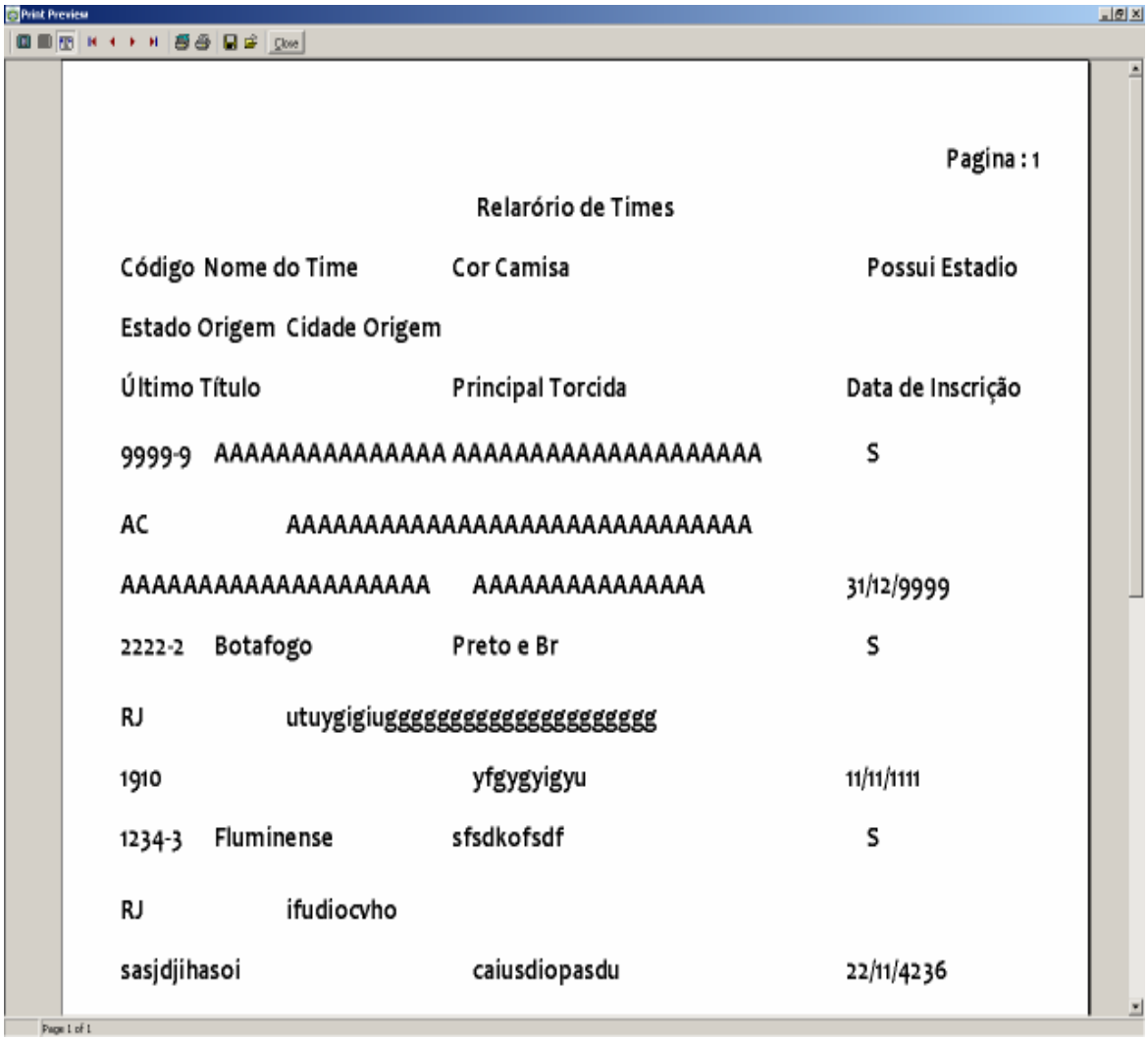



```
unit UnitBuscaCodigo;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, ADODB, jpeg, ExtCtrls, StdCtrls, Buttons, Grids, DBGrids,
  Mask;

type
  TFormBuscaCodigo = class(TForm)
    DBGrid1: TDBGrid;
    BitBtnSair: TBitBtn;
    Image2: TImage;
    TblTimes: TADOTable;
    TblTimesNomeDoTime: TWideStringField;
    TblTimesCodigo: TWideStringField;
    TblTimesUltimoTitulo: TWideStringField;
    TblTimesCorCamisa: TWideStringField;
```

```
    TblTimesCidadeOrigem: TWideStringField;
    TblTimesEstadoOrigem: TWideStringField;
    TblTimesPricipalTorcida: TWideStringField;
    TblTimesPossuiEstadio: TWideStringField;
    TblTimesDataDeInscricao: TDateTimeField;
    DataSource1: TDataSource;
    Label3: TLabel;
    EditCodigo: TEdit;
    procedure BitBtnSairClick(Sender: TObject);
    procedure EditCodigoChange(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  FormBuscaCodigo: TFormBuscaCodigo;
implementation
{$R *.dfm}
procedure TFormBuscaCodigo.BitBtnSairClick(Sender: TObject);
begin
  Close; //Fecha o formulário
end;
procedure TFormBuscaCodigo.EditCodigoChange(Sender: TObject);
begin
```

```
  TBLTimes.Locate ('Codigo', EditCodigo.Text, [loPartialKey]); //Procura na base de dados do
ADOTable o código digitado no Edit, de forma parcial, toda vez que uma letra é digitada

end;

procedure TFormBuscaCodigo.FormClose(Sender: TObject;

  var Action: TCloseAction);

begin

  EditCodigo.Clear; //Limpa o Edit

  TBLTimes.Active:=False; //Desativa a base de dados do ADOTable

end;

procedure TFormBuscaCodigo.FormShow(Sender: TObject);

begin

  TBLTimes.Active:=True; //Ativa a base de dados do ADOTable

end;

procedure TFormBuscaCodigo.FormCreate(Sender: TObject);

begin

    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que
seja movido o formulário

end;

end.
```

4.8. Relatório




```
unit UnitRelatorio;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, ADODB, QRCtrls, QuickRpt, ExtCtrls;

type
  TFormRelatorio = class(TForm)
    QuickRepTimes: TQuickRep;
    PageHeaderBand1: TQRBand;
    TitleBand1: TQRBand;
```

```
    DetailBand1: TQRBand;
    PageFooterBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRDBTextNomeDoTime: TQRDBText;
    QRDBTextCodigo: TQRDBText;
    QRDBTextUltimoTitulo: TQRDBText;
    QRDBTextCorCamisa: TQRDBText;
    QRBand1: TQRBand;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel9: TQRLabel;
    TblTimes: TADOTable;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel10: TQRLabel;
    QRDBTextCidadeOrigem: TQRDBText;
    QRDBTextEstadoOrigem: TQRDBText;
    QRDBTextPricipalTorcida: TQRDBText;
    QRDBTextPossuiEstado: TQRDBText;
    QRDBTextDataDeInscricao: TQRDBText;
    QRSysDataPagina: TQRSysData;
    QRSysDataDataSistema: TQRSysData;
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
```

```
  public

    { Public declarations }

  end;

var

  FormRelatorio: TFormRelatorio;

implementation

{$R *.dfm}

procedure TFormRelatorio.FormCreate(Sender: TObject);

begin

     DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que
seja movido o formulário

end;

end.
```

4.9. Valor Cachê


```
unit UnitValorCache;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Buttons, jpeg;

type
  TFormValorCache = class(TForm)
    Image1: TImage;
    BitBtnCalcular: TBitBtn;
    BitBtnSair: TBitBtn;
    BitBtnLimpar: TBitBtn;
    Label3: TLabel;
    EditQuantidade: TEdit;
    procedure BitBtnSairClick(Sender: TObject);
    procedure BitBtnLimparClick(Sender: TObject);
    procedure BitBtnCalcularClick(Sender: TObject);
```

```
    procedure FormCreate(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
  Function CalculaCache(QtdTorcedores:Integer):Real;  //Declaração da função
var
  FormValorCache: TFormValorCache;
  Cache:Real; //Declaração de variáveis Universais do form
  QuantidadeTorcedores:Integer; //Declaração de variáveis Universais do form
implementation
{$R *.dfm}
Function CalculaCache(QtdTorcedores:Integer):Real; //Função para calculo do cachê
Var Cach:Real;
Begin
  If (QtdTorcedores<=3000) Then
    Cach:=10*QtdTorcedores //Efetua Calculo do cachê até 3000
  Else If (QtdTorcedores<=15000) Then
    Cach:=15*QtdTorcedores  //Efetua Calculo do cachê entre 3001 de 15000
  Else If (QtdTorcedores<=80000) Then
    Cach:=18*QtdTorcedores //Efetua Calculo do cachê entre 15001 de 80000
  Else
    Cach:=20*QtdTorcedores; //Efetua Calculo do cachê acima de 80000
  Result:=Cach;  //Retorna o resultado
end;
```

```
procedure TFormValorCache.BitBtnSairClick(Sender: TObject);

begin

  Close; //Fecha o formulário

end;

procedure TFormValorCache.BitBtnLimparClick(Sender: TObject);

begin

  EditQuantidade.Clear; //Limpa o Edit

  EditQuantidade.SetFocus; //Coloca a seleção no Edit

end;

procedure TFormValorCache.BitBtnCalcularClick(Sender: TObject);

begin

  Try //tratamento de erro parte 1

    QuantidadeTorcedores:=StrToInt(EditQuantidade.Text); //Se o campo estiver preenchido copia
os dados do Edit para a variável QuantidadeTorcedores

  Except //tratamento de erro parte 2

    ShowMessage('Digite um valor válido!'); //Se o campo não estiver preenchido apresenta uma
mensagem  de erro

    EditQuantidade.Clear; //Limpa o Edit

    EditQuantidade.SetFocus; //Coloca a seleção no Edit

    Exit; // Aborta o procedimento

  end;    //tratamento de erro parte 3

  Cache:=CalculaCache(QuantidadeTorcedores);  //Executa a função com o valor da variavel
QuantidadeTorcedores

  ShowMessage('Valor do Cachê: R$ '+FloatToStrF(Cache,ffnumber,15,2)); //Apresenta uma
menssagem com o resultado

  EditQuantidade.Clear; //Limpa o Edit

  EditQuantidade.SetFocus; //Coloca a seleção no Edit

end;

procedure TFormValorCache.FormCreate(Sender: TObject);

begin
```

```
    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que
seja movido o formulário

end;

procedure TFormValorCache.FormClose(Sender: TObject;

  var Action: TCloseAction);

begin

  EditQuantidade.Clear; //Limpa o Edit

  EditQuantidade.SetFocus; //Coloca a seleção no Edit

end;

procedure TFormValorCache.FormShow(Sender: TObject);

begin

  Cache:=0.0;  //Inicia a variável Zerada

end;

end.
```

4.10. Verificar Confirmações


```
unit UnitVerificarConfirmacoes;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, jpeg, ExtCtrls, StdCtrls, Buttons, Mask;
type
  TFormVerificarConfirmacoes = class(TForm)
    BitBtnVerificar: TBitBtn;
    BitBtnLimpar: TBitBtn;
    BitBtnSair: TBitBtn;
    Image1: TImage;
    Label3: TLabel;
    MaskEditCodigo: TMaskEdit;
    procedure BitBtnSairClick(Sender: TObject);
    procedure BitBtnVerificarClick(Sender: TObject);
    procedure BitBtnLimparClick(Sender: TObject);
```

```
    procedure FormClose(Sender: TObject; var Action: TCloseAction);

    procedure FormCreate(Sender: TObject);

  private

    { Private declarations }

  public

    { Public declarations }

  end;

  Procedure VerificarDigito(Numero:String; Var DigVerificador:Integer); //Declaração do
procedimento

var

  FormVerificarConfirmacoes: TFormVerificarConfirmacoes;

  Codigo,Verificar:String[6];  //Declaração de variáveis Universais do form

  DigitoVerificador:String[1];  //Declaração de variáveis Universais do form

  i,DigitoVerificador1:Integer;  //Declaração de variáveis Universais do form

implementation

{$R *.dfm}

Procedure VerificarDigito(Numero:String; Var DigVerificador:Integer);   //Procedimento para
verificação do digito verificador do código do Time

Var Soma,Peso,Digito,Indice,Cod,DigitoVerif:Integer;

Begin

  Soma:=0; //Inicia a variável com valor 0

  Peso:=2; //Informa o primeiro multiplicador

  For Indice:=Length(Numero) Downto 1 do  //Deecrementa do Valor do tamanho total do índice
até o primeiro digito informado

    Begin

      Val (Numero[Indice],Digito,Cod); //Seleciona o digito do índice do código informado

      Soma:=Soma+(Digito*Peso); //Efetua a soma da multiplicação dos 4 dígitos por 5,4,3 e 2

      Peso:=Peso+1; //Incrementa o multiplicador em 1

    End;

    DigitoVerif:=Soma Mod 11; // Efetua o calculo para verificar se resto é 1 ou 0
```

```
    If (DigitoVerif=0) or (DigitoVerif=1) Then

      DigVerificador:=0    //se o resto obtido for 0 ou 1 o digito verificador será 0

    Else

      DigVerificador:=11-DigitoVerif; //Se o digito obtido for diferente de 0 ou 1 o digitoverificador
será 11 - o digito obtido

End;

procedure TFormVerificarConfirmacoes.BitBtnSairClick(Sender: TObject);

begin

  Close; //Fecha o formulário

end;

procedure TFormVerificarConfirmacoes.BitBtnVerificarClick(Sender: TObject);

begin

  Codigo:=MaskEditCodigo.Text; //Coloca os dados da MaskEdit em uma variável

  i:=Length(Codigo); //Gera o índice com o tamanho do código

  Verificar:=Copy(Codigo,1,i-2); //Seleciona somente o numero do código informado

  VerificarDigito(Verificar,DigitoVerificador1); //Executa o procedimento informando o numero
informado e retornando um numero inteiro

  STR(DigitoVerificador1,DigitoVerificador); //Transforma o numero inteiro obtido em string

  Verificar:=Verificar+'-'+DigitoVerificador; //Junta o digito verificador gerado com o numero
copiado

  MaskEditCodigo.Text:=Verificar;

  If Verificar=Codigo Then //Compara o numero gerado com o numero informado

    Showmessage('O Codigo '+Codigo+' é Válido') //Apresenta uma mensagem se o código for
válido

  Else

    Showmessage('O Codigo '+Codigo+' é Inválido'); //Apresenta uma mensagem se o código for
inválido

  MaskEditCodigo.Clear; //Limpa o MaskEdit

  MaskEditCodigo.SetFocus; //Coloca a seleção no MaskEdit

end;

procedure TFormVerificarConfirmacoes.BitBtnLimparClick(Sender: TObject);
```

```
begin

  MaskEditCodigo.Clear; //Limpa o MaskEdit

  MaskEditCodigo.SetFocus; //Coloca a seleção no MaskEdit

end;

procedure TFormVerificarConfirmacoes.FormClose(Sender: TObject;

  var Action: TCloseAction);

begin

  MaskEditCodigo.Clear; //Limpa o MaskEdit

  MaskEditCodigo.SetFocus; //Coloca a seleção no MaskEdit

end;

procedure TFormVerificarConfirmacoes.FormCreate(Sender: TObject);

begin

    DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); ); //Impede que
seja movido o formulário

end;

end.
```

4.11. Autor Informações


```
unit UnitAutorInformacoes;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, jpeg, ExtCtrls, StdCtrls, Buttons;


type
  TFormAutorInformacoes = class(TForm)
    Image1: TImage;
    BitBtnSair: TBitBtn;
    Image2: TImage;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
```

```
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    procedure BitBtnSairClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  FormAutorInformacoes: TFormAutorInformacoes;
implementation
{$R *.dfm}
procedure TFormAutorInformacoes.BitBtnSairClick(Sender: TObject);
begin
  Close; //Fecha o formulário
end;
procedure TFormAutorInformacoes.FormCreate(Sender: TObject);
begin
  DeleteMenu(GetSystemMenu(Handle, False), SC_MOVE, MF_BYCOMMAND); //Impede que seja
movido o formulário
end;
end.
```