TIPOS DE ARRAY

Um array é uma coleção ordenada de elementos do mesmo tipo de dados, que faz uso de um índice para dar acesso aos items da coleção. Arrays são úteis em diversas situações. Como o índice permite acesso direto aos elementos da lista, arrays fornecem um poderoso mecanismo para se organizar dados. O exemplo a seguir mostra como declarar uma array :

EXEMPLO 1:

```
Procedure Tform1.Button1Click (Sender: Tobject) ;
var
    Dias_de_Semana: array[1..7] of  String ;

var
     DiaNo: Integer;
      DiadaSemana: String;

begin

          {Inicializa o array com nomes dos dias da semana }
Dias_de_Semana [1] := 'Domingo' ;
Dias_de_Semana [2] := 'Segunda-feira' ;
Dias_de_Semana [3] := 'Terça-feira' ;
Dias_de_Semana [4] := 'Quarta-feira' ;
Dias_de_Semana [5] := 'Quinta-feira' ;
Dias_de_Semana [6] := 'Sexta-feira' ;
Dias_de_Semana [7] := 'Sábado' ;

DiaNo :=DayOfWeek (Date) ;    (*  Retorna o número correspondente ao dia da semana da
                                   data atual *)
ShowMessage(' Hoje é ' + Dias_de_Semana [DiaNo] ) ;
.....
```

EXEMPLO 2: Uso do comando FOR

O comando For deve ser usado sempre que se souber com antecedência quantas vezes o laço deverá ser executado. Como ilustração vamos resolver o seguinte problema :

> No baile Funk encontram-se quatro garotas (Juliana, Natália, Carolina e Adriana) e quatro rapazes (Guilherme, Neto, Leonardo e Eduardo). Pede-se : quantos - e quais - pares podem ser formados com essa turma ?

Para resolver:

⇒ Crie uma nova Aplicação (File | New Application)
⇒ Acrescente um ListBox
⇒ Acrescente um Button

Dê dois cliques sobre o Button e acrescente o código :

```
procedure TForm1.Button1Click(Sender: TObject);

var
       ElasList : array[1..4]  of String [10] ;
       ElesList : array[1..4]  of String[10];

i , j : integer;

begin

     ElasList[1] := 'Juliana' ;
     ElasList[2] := 'Natália' ;
     ElasList[3] := 'Carolina' ;
     ElasList[4] := 'Adriana' ;
     ElesList[1] := 'Guilherme' ;
     ElesList[2] := 'Neto' ;
     ElesList[3] := 'Leonardo' ;
     ElesList[4] := 'Eduardo' ;

 for  i:=1 to 4  do
    for  j:=1 to 4  do
      ListBox1.Items.Add(ElasList[i] + ' '+ElesList[j]);

end;

end.
```

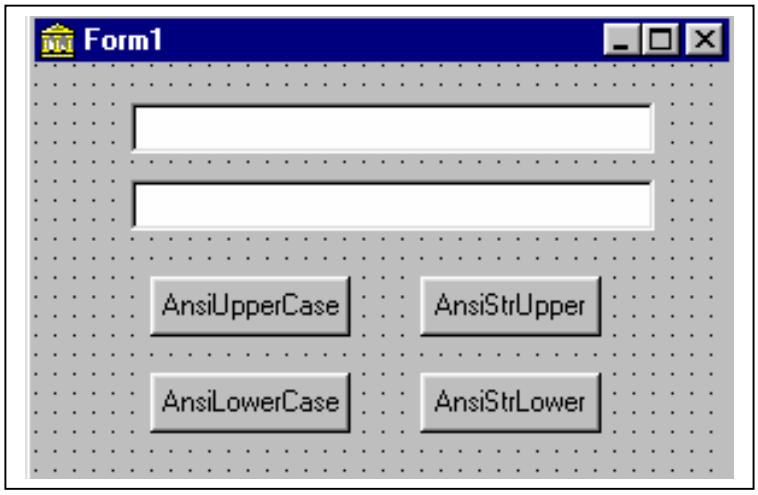
Com base nos novos conhecimentos adquiridos, resolva o ESTUDO DE CASO da aula passada. (FOLHA DA AULA 4).

Certo diretor andava com a memória falhando e para evitar esquecer os aniversários e datas importantes da firma, pediu ao seu analista de sistemas para que criasse um pequeno aplicativo . O programinha deveria ser instalado na pasta Iniciar do Windows 95 do seu micro e , ao ligar o micro todas as manhãs, o faria recordar as referidas datas. Sabendo-se que a firma é pequena, tendo apenas 10 funcionários e o analista é perito em linguagem Delphi, sugira um modo dele resolver esse problema.

ROTINAS DE STRING

A capacidade de se manipular strings é uma habilidade chave entre programadores. Muitas vezes, por exemplo, você precisará decompor (isto é, separar em partes significativas) uma string, e analisar suas diversas partes. Um exemplo típico seria extrair um item específico, tal como um código, letra de unidade de disco, ou caminho de busca em um mermorando. O Delphi oferece uma rica biblioteca de funções e examinaremos abaixo algumas delas , por meio de aplicativos específicos que demonstrarão ao aluno sua utilidade.

EXERCÍCIO 1:



Proceda da seguinte forma :

- ⇒ **Inclua uma nova aplicação (File | New Application)**
- ⇒ **Inclua dois componentes Edit**
- ⇒ **Alterne as propriedades Text para ' '(branco)**
- ⇒ **Altere Enabled do segundo Edit para False**
- ⇒ **Inclua um componente Button**
- ⇒ **Altere Caption para: AnsiUpperCase**
- ⇒ **Dê dois cliques no Button e digite: Edit2.Text:=AnsiUpperCase(Edit1.Text);**
- ⇒ **Inclua outro componente Button**
- ⇒ **Altere Caption para : AnsiStrUpper**
- ⇒ **Dê dois cliques no Button e digite:**

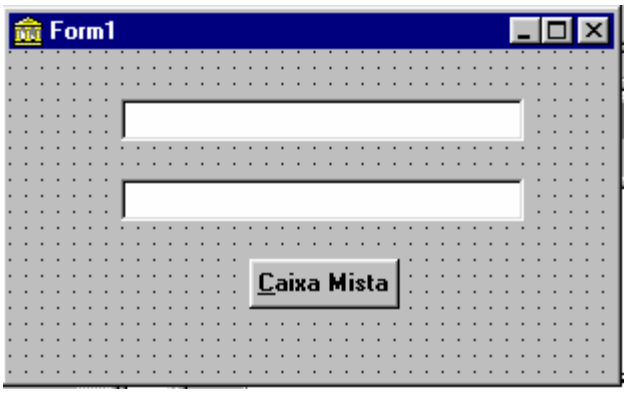
```
var
    s : pchar;
begin
    s := ' Estácio de Sá - Instituto Politécnico' ;
     Edit2.Text :=AnsiStrUpper(s);
end;
```

- ⇒ **Inclua outro componente Button**
- ⇒ **Altere Caption para: AnsiLowerCase**
- ⇒ **Dê dois cliques no Button e digite: Edit2.Text:=AnsiUpperLower(edit1.Text);**
- ⇒ **Inclua outro componente Button**
- ⇒ **Altere Caption para: AnsiStrLower**
- ⇒ **Dê dois cliques no Button e digite:** →
- ⇒ **Execute o programa.**

```
var
    s : pchar;
begin
    s := ' Estácio de Sá - Instituto Politécnico' ;
     Edit2.Text :=AnsiStrLower(s);
end;
```

EXERCÍCIO 2: Exibindo Caixa Mista

APLICAÇÃO DAS FUNÇõES POS(), COPY(), TRIMRIGHT() , LENGTH()



Muitas vezes os dados (principalmente nome de pessoas) estão armazenados em caixa alta(todas as letras em maiúsculo); na hora de imprimirmos os relatórios, entretanto, as palavras em caixa alta ocupam muito espaço e dão a impressão de que uma linha está muito próxima das outras.

Por isso é importante que, na hora da impressão, transformemos as linhas em caixa mista (a primeira letra em maiúsculo e as demais em minúsculo). Neste exemplo trabalhamos com a função POS(). Para transformar texto para a caixa mista, proceda da seguinte forma:

- ⇒ **Inclua um novo formulário (File | New Application)**
- ⇒ **Inclua dois componentes Edit**
- ⇒ **Altere as propriedades Text para ' ' (branco)**
- ⇒ **Inclua um componente Button**
- ⇒ **Altere Caption para : &Caixa Mista**
- ⇒ **Dê dois cliques sobre o Button e digite:**

Edit2.Text :=CaixaMista(Edit1.Text) ;

⇒ **Localize as linhas:** implementation
{$R*.DFM}

Logo abaixo digite :

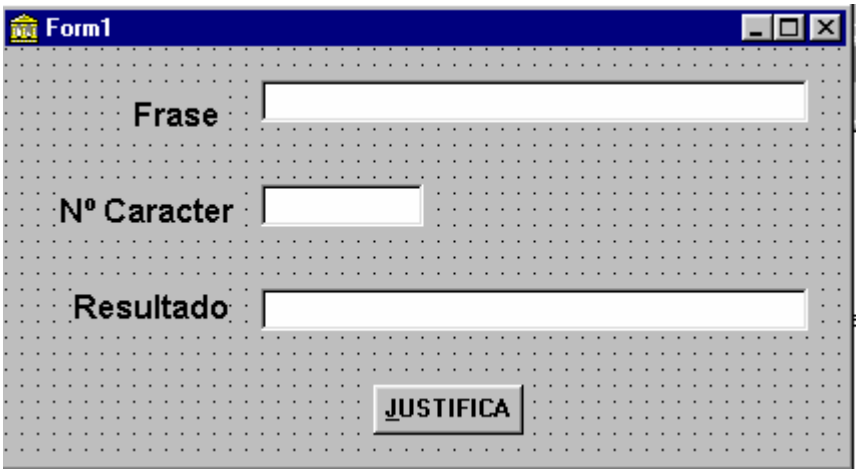
```
function CaixaMista (mNome: string): string;

 var tam,pos1,pos2 : integer ;   pal : string;

 begin
    tam := Length(mNome);
    mNome := TrimRight(mNome) + ' ';

    while True do

    begin
       pos1:=POS( ' ' , mNome) ;

       if pos1 = 0 then break;

          pal := Copy(mNome,1,pos1) ;
          pos2 := pos(pal, 'DA - DAS -DE -DO -DOS ');
```

```
                    If pos2 > 0 then pal :=AnsiLowerCase (pal)
                        else pal:=Copy(pal,1,1) + AnsiLowerCase(Copy(pal,2,tam))  ;
                     result := result + pal ;
                     mNome := copy(mNome,pos1+1,tam)
                    end;
            end;
```

⇒ Execute o programa.

JUSTIFICANDO TEXTOS :

A tradição exige que um texto deve ser alinhado (justificado) à esquerda e à direita. A rotina de justificação deve ser usada principalmente com o componente ***Memo.*** O Delphi permite o alinhamento à esquerda, à direita e centralizado. Para vermos uma string sendo justificada à esquerda e à direita, agimos da seguinte maneira :



⇒ Insira um novo formulário ;
⇒ Insira no formulário três Label com Caption = N0. Caracter e Resultado
⇒ Troque o tipo de Fonte das Caixas Edit Para Tipo Courier
⇒ Insira três ***Edit*** no formulário e troque suas propriedades Caption para ' '(branco)
⇒ Insira um Button no componente e troque sua propriedade Caption para &Justifica
⇒ Dê dois cliques no Button , localize as palavras implementation
*{$R *.DFM}*

Logo abaixo digite a função :

```
function Justifica(mCad : string ; mMAx : integer) : string;
var
   mPos,mPont,mTam,mNr,mCont : integer;
   mStr                      : string;
begin
     mTam := Length (mCad);
```

```
if mTam >= mMax then
    Result := copy(mCad,1,mMax)
else
    mStr := '  ';
    mCont : = 0 ;
    mPont := 1;
    mNr := mMax - mTam ;

   while mCont < mNr do
       begin
         mPos := pos(mStr,copy(mCad,mPont,100)) ;
         if mPos = 0 then
              begin
                    mStr := mStr + '  ' ;
                    mPont : = 1;
                    continue;
          end
           else
                   begin
                         mCont := mCont + 1;
                          Insert( ' ', mCad, mPos + mPont);
                          mPont := mPont + mPos + length(mStr);
            end;
           Result := mCad;
          end;

end;
```

SOLUÇÃO OFICIAL DO ESTUDO DE CASO DA PÁGINA 2 :

```
unit PNIVER;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm1 = class(TForm)
    Label1: TLabel;
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.FormCreate(Sender: TObject);

var
DataNiver:ARRAY[1..10] of String ;
var
i:Integer;
var
Indice:Integer;

begin
 Indice:=0;
 DataNiver[1]:='12/01/98';
 DataNiver[2]:='23/09/98';
 DataNiver[3] :='04/02/98';
DataNiver[4] :='05/01/98';
DataNiver[5] :='06/02/98';
DataNiver[6] :='07/02/98';
DataNiver[7] :='08/02/98';
DataNiver[8] :='09/02/98';
DataNiver[9] :='01/02/98';
DataNiver[10] :='14/02/98';
```

```
for i:=1 to 10 do
   if DatetoStr(Date)=DataNiver[i] then
      Indice:=i ;


Case Indice of
 1: Label1.Caption:='Francisco faz anos hoje !' ;
 2: Label1.Caption:='Vera faz anos hoje !' ;
 3: Label1.Caption :='Ronaldo faz anos hoje !' ;
 4: Label1.Caption :='Isaura faz anos hoje !' ;
 5: Label1.Caption :='Genésio faz anos hoje !' ;
 6: Label1.Caption :='Roberto faz anos hoje !' ;
 7: Label1.Caption :='Rita faz anos hoje !' ;
 8: Label1.Caption :='Rafael faz anos hoje !' ;
 9: Label1.Caption :='Rogério faz anos hoje !' ;
 10: Label1.Caption :='Carlos faz anos hoje !' ;



 end;
end;



end.
```