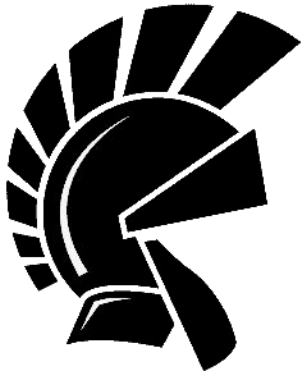

Curso de Delphi

4.Arrays e Records

Mateus Schwede - UB Social

Componentes

- Menus
  - PopupMenu
  - MainMenu
- Timer
- Arrays e StringGrid
  - array[lin,col] //Backend
  - strinGrid[col,lin] //Frontend
- Records
- Gráficos (TPaintBox, propriedade Canvas)
- Arquivos de texto e DialogsMessages

Menus

- PopupMenu
  - Abrir: popMenOpcoes.Popup(3, 20);
  - Fechar: popMenOpcoes.CloseMenu;
- MainMenu

Data e Hora do sistema

- Timer (Interval=1000, código abaixo)
  - lblDataHora.Caption := TimeToStr(time) + ' ' + FormatDateTime('dddd, dd "de" mmmm "de" yyyy',now);

Arrays (Sintaxe)

```
procedure TForm1.Exemplo;
var
    vetor: array [0..5] of Integer; //Lista de 6 itens inteiros
    matriz: array[1..3, 1..3] of integer;
    mat3d: array[1..12, 1..10, 1..3] of integer;
    vet: array of Integer;
begin
    ShowMessage(SizeOf(arrayInt).ToString);
    ShowMessage(SizeOf(arrayDinamico).ToString);
end;
```

Arrays multidimensionais

```
procedure TForm1.Exemplo2;
var
        MatrizMulti: array [0..3, 0..1] of integer;
begin
        MatrizMulti[0][0] := 5;
        MatrizMulti[0][1] := 10;

        MatrizMulti[1][0] := 15;
        MatrizMulti[1][1] := 20;

        MatrizMulti[2][0] := 25;
        MatrizMulti[2][1] := 30;

        MatrizMulti[3][0] := 35;
        MatrizMulti[3][1] := 40;

        ShowMessage(SizeOf(MatrizMulti).ToString);
end;
```

Arrays dinâmicos

```
procedure TForm1.Exemplo3;
var
    arrayDin: array of integer;
    I: Integer;
begin
    SetLength(arrayDin, 5); //arrayDin conterá 5 itens
    arrayDin[0] := 10;
    arrayDin[1] := 20;

    arrayDin[6] := 60;
    arrayDin[99] := 10999;

    ShowMessage(arrayDin[6].ToString);
    ShowMessage(arrayDin[99].ToString);
```

- Não pode ser acessada em iteração baseada no "length" do array
- Só possível acessar valor conhecendo a posição exata do índice
    - No caso são 6 e 99

Arrays dinâmicos - Percorrer

```
for I := Low(arrayDin) to High(arrayDin) do
begin
        ShowMessage(arrayDin[I].ToString);
end;
```

Arrays dinâmicos - Aumentar

```
SetLength(arrayDin, Length(arrayDin) + 1);
arrayDin[Length(arrayDin) - 1] := 55;

for I := 0 to Pred(Length(arrayDin)) do
begin
        ShowMessage(arrayDin[I].ToString);
end;
```

Comparação de arrays

```
procedure TForm1.Exemplo5;
var
        lstA, lstB, lstC: array of Integer;
begin
        setLength(lstA, 3);
        setLength(lstB, 3);

        lstA[0] := 10;
        lstA[1] := 20;
        lstA[2] := 30;

        lstB[0] := 10;
        lstB[1] := 20;
        lstB[2] := 30;
```

Comparação dos arrays A e B:

```
if (lstA = lstB) then
begin
        ShowMessage('Iguais');
end
else ShowMessage('Não iguais'); //Aqui

lstC:= lstA;
```

Comparação dos arrays A e C:

```
if (lstA = lstC) then
begin
        ShowMessage('Iguais'); //Aqui
end
else ShowMessage('Não iguais');
```

Operações com arrays - Soma

Somar matrizes:

```
MatrizA[0] := 0;
MatrizA[1] := 1;
MatrizA[2] := 2;

MatrizB[0] := 3;
MatrizB[1] := 4;
MatrizB[2] := 5;

MatrizC := Concat(MatrizA, MatrizB);
ShowMatriz(MatrizC);
```

Operações com arrays - Inserir e deletar

```
MatrizC := [];
MatrizC := MatrizA + [10, 20, 30];

MatrizC := [10, 20, 30];
Insert([5, 15], MatrizC, 0); //Insere 5 e 15 no inicio da MatrizC
ShowMatriz(MatrizC);

Insert([85, 95], MatrizC, Length(MatrizC)); //Insere 85 e 95 no fim da MatrizC
ShowMatriz(MatrizC);

Insert(MatrizB, MatrizC, Length(MatrizC)); //Insere a MatrizB no fim da MatrizC
ShowMatriz(MatrizC);

Delete(MatrizC, 0, 5); //Deleta nos índices 0 ao 5 da matrizC
ShowMatriz(MatrizC);
```

Arrays e StringGrid

```
procedure TForm1.BitBtn2Click(Sender: TObject);
var
        col, num, lin: Integer;
        vetor: array[1..10] of Integer;
        vet: array of Integer;
        matriz: array[1..3, 1..3] of Integer;
begin
        //Inicialização
        for lin := 0 to sgMat.RowCount-1 do
                sgMat.Cells[0,lin+1] := IntToStr(lin+1);
                for col := 0 to sgMat.ColCount-1 do
                        sgMat.Cells[col+1,0] := IntToStr(col+1);

        num := 0;
        for lin := 1 to sgMat.RowCount-1 do
        begin
                for col := 1 to sgMat.ColCount-1 do
                begin
                        sgMat.Cells[col,lin] := IntToStr(num);
                        matriz[lin,col] := StrToInt(sgMat.Cells[col,lin]);
                        inc(num,2);
                end;
        end;
```

Arrays e StringGrid

```
    Randomize;
    for col := 0 to sgVet.ColCount-1 do
    begin
        sgVet.Cells[col,1] := IntToStr(Random(50));
        vetor[col] := StrToInt(sgVet.Cells[col,1]);
    end;

    {
        sgMat.Cells[0,1] := 'Janeiro';
        sgMat.Cells[0,2] := 'Fevereiro';
        sgMat.Cells[0,3] := 'Março';
    }
end;
```

Records e Gráficos

```
type
    Pontos = Record
        X,Y: Integer;
        Cor: string;
        OnOff: Boolean;
    end;

    TRegistro = Class
        Nome: String;
        Telefone: String;
        Cidade: String;
    end;

var
    PontosTela: Array [1..1024] of Pontos;
```

Records e Gráficos

```
procedure TForm1.Button1Click(Sender: TObject);
begin
        PontosTela[1].X := 300;
        PontosTela[1].Y := 679;
        PontosTela[1].Cor := 'Vermelho';
        PontosTela[1].OnOff := True;
        PontosTela[2] := PontosTela[1];
        ShowMessage(PontosTela[2].X.ToString);
end;
```

Records - StringList

```
procedure AddRecord(lista:TStringList; strNome, strTelefone, strCidade:String);
var
        recNum: Integer;
begin
        lista.AddObject(strNome, TRegistro.Create);
        recNum := Lista.Count-1;
        With TRegistro(Lista.Objects[RecNum]) do
        begin
                nome := strNome;
                telefone := strTelefone;
                cidade := strCidade;
        end;
end;
```

Records - Memo e ListBox

```
procedure TForm1.Button3Click(Sender: TObject);
var
        lista: TStringList;
        x: Integer;
begin
        lista := TStringList.Create;
        addRecord(Lista,'Joao', 'Belo Horizonte');
        addRecord(Lista,'Ana', ,'São Paulo');
        addRecord(Lista,'Iris', ,'Rio de Janeiro');
        lista.Sorted := True;

        for x := 0 to Lista.Count-1 do
        begin
                mem1.Lines.Add ( TRegistro(lista.Objects[x]).nome + ' ' +TRegistro(lista.Objects[x]).cidade );
        end;

        for x := 0 to Lista.Count-1 do
        begin
                lista.Objects[x].Destroy;
        end;
        lista.Free;
end;
```

Complementos

- Módulo 10 da apostila: Gráficos (TPaintBox, propriedade Canvas)
- Módulo 24 da apostila: Mais informações sobre arquivo de texto
- Módulo 25 da apostila: Complementa sobre arquivos de texto, pois mostra como interagir com as principais janelas de diálogos do sistema

- Códigos na descrição, ou em ‘ubsocial.github.io’
  - Atividade extra (vetor, matriz, pageControl, trackBar e timer)
  - Desafio das matrizes matemáticas

UB Social

ubsocial.github.io