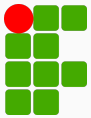
Pascal

05 - Estruturas de Dados Heterogêneas

Marcos Roberto Ribeiro



Instituto Federal Minas Gerais - Campus Bambuí

2018

Introdução

- Em aulas anteriores trabalhamos com estruturas de dados homogêneas (vetores, strings e matrizes) que permitem criar um agrupamento de dados do mesmo tipo;
- Todavia existem situações em que precisamos agrupar tipos de dados diferentes. Por exemplo, quando precisamos manipular vários dados de um aluno como código, nome, notas;
- Em tais situações devemos utilizar estruturas de dados heterogêneas, também conhecidas como registros.

Especificando Registros

- A especificação de registros é feita nas definições de tipo **type**;
- Basicamente a declaração de um registro possui a seguinte sintaxe:

```
type
  TRegistro = record
    Campo1: Tipo1;
    Campo2: Tipo2;
    ...
    CampoN: TipoN;
  end;
```

- Depois que declaramos um registro podemos declarar variáveis com o mesmo, por exemplo:

```
var
  Variavel: TRegistro;
```

- Depois de declarada uma variável do tipo registro podemos acessar seus campos acrescentando um ponto e o nome do campo desejado. Por exemplo:

```
  Variavel.Campo1 := 10;
  X := Variavel.Campo2;
```

Exemplo Simples (Aprovação de Aluno) I

```
program ExemploRegistro;
const
  {Quatro bimestres}
  NUM_NOTAS = 4;

type
  TVetNotasBimestres = array [1..NUM_NOTAS] of Real;
  TAluno = record
    {Delimitação para trabalhar com arquivos}
    Nome: String[60];
    {Podemos ter campos que são vetores}
    Notas: TVetNotasBimestres;
  end;

var
  Aluno: TAluno;
  Cont: Integer;
  Soma: Real;
```

Exemplo Simples (Aprovação de Aluno) II

```
BEGIN
  Write('Informe o nome do aluno: ');
  ReadLn(Aluno.Nome);
  Soma := 0;
  for Cont:= 1 to NUM_NOTAS do begin
     Write('Bimestre ', Cont,'. Informe a nota: ');
     ReadLn(Aluno.Notas[Cont]);
     Soma := Soma + Aluno.Notas[Cont];
  end;
  if (Soma >= 60) then begin
    WriteLn('O aluno ', Aluno.Nome, ' foi aprovado com a nota ', Soma:0:2);
  end else begin
    WriteLn('O aluno ', Aluno.Nome, ' foi reprovado com a nota ', Soma:0:2);
  end;
END.
```

Vetores de Registros e Exemplo

- O programa do exemplo anterior permite trabalhar apenas com um aluno de cada vez;
- Para trabalhar com uma turma de alunos podemos criar um vetor de registros **TAluno**;
- Este vetor funciona da mesma forma que os vetores já estudados, a única diferença é que em cada posição do vetor existe um registro **TAluno**;
- Esta alteração é demonstrada no próximo programa de exemplo.

unit alunos; I

```
interface

const
  NUM_ALUNOS = 10;
  NUM_NOTAS = 4;

type
  TVetNotasBimestres = array [1..NUM_NOTAS] of Real;
  TAluno = record
    Nome: String[60];
    Notas: TVetNotasBimestres;
  end;
  TVetAlunos = array [1..NUM_ALUNOS] of TAluno; {Vetor de TAluno}

function LeAluno():TAluno;
function CalculaMediaNotas(VetorAlunos: TVetAlunos):Real;
procedure ListaAlunosAcima(VetorAlunos: TVetAlunos; Nota:Real);

implementation
```

unit alunos; II

```
{Função que lê e retorna um registro TAluno}
function LeAluno():TAluno;
var
  Aluno:TAluno;
  ContN: Integer;
begin
  Write('Informe o nome do aluno: ');
  ReadLn(Aluno.Nome);
  for ContN:= 1 to NUM_NOTAS do begin
    Write('Bimestre ', Cont,'. Informe a nota: ');
    ReadLn(Aluno.Notas[ContN]);
  end;
  LeAluno:=Aluno;
end;
```

unit alunos; III

```
{Função que calcula a soma de todas as notas}
function CalculaSomaNotas(VAlunos: TVetAlunos):Real;
var
  ContA, ContN: Integer;
  Soma: Real;
  Aluno: TAluno;
begin
  Soma:= 0;
  {Considera todos os alunos e suas notas}
  for ContA := 1 to NUM_ALUNOS do begin
    for ContN := 1 to NUM_NOTAS do begin
      Aluno := VAlunos[ContA];
      Soma := Soma + Aluno.Notas[ContN];
    end;
  end;
  CalculaSomaNotas:= Soma;
end;
```

unit alunos; IV

```
{Função que calcula a média final das notas}
function CalculaMediaNotas(VAlunos: TVetAlunos): Real;
begin
  CalculaMediaNotas:= CalculaSomaNotas(VAlunos) / NUM_ALUNOS;
end;
```

unit alunos; V

```
{Procedimento que lista os alunos acima de uma nota}
procedure ListaAlunosAcima(VAlunos: TVetAlunos; Nota:Real);
var
  ContA, ContN: Integer;
  Soma: Real;
  Aluno: TAluno;
begin
  for ContA := 1 to NUM_ALUNOS do begin
    Soma:= 0; {Soma receberá a nota final do aluno}
    Aluno := VAlunos[ContA];
    for ContN := 1 to NUM_NOTAS do begin
      Soma := Soma + Aluno.Notas[ContN];
    end;
    if (Soma > Nota) then begin
      WriteLn(Aluno.Nome);
    end;
  end;
end;

end.
```

program notas;

```
uses alunos;

var
  VAlunos:TVetAlunos;
  Cont: Integer;
  Media: Real;

begin
  for Cont := 1 to NUM_ALUNOS do begin
    VAlunos[Cont] := LeAluno();
  end;
  Media := CalculaMediaNotas(VAlunos);
  WriteLn('Alunos acima da média de ', Media:0:2);
  ListaAlunosAcima(VAlunos, Media);
end.
```

Referências I

Canneyt, M. V.

Free pascal reference guide.

`http://www.freepascal.org/docs-html/ref/ref.html`.

Evaristo, J. (1999).

***Programando com Pascal*.**

Book Express, Rio de Janeiro, 2 edition.

Manzano, J. A. N. G. and Yamatumi, W. Y. (2001).

***Programando em turbo pascal 7.0 e free pascal compiler*.**

Érica, São Paulo, 9 edition.