

UNISINOS - UNIVERSIDADE DO VALE DO RIO DOS SINOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS (C6/6) – Curso: Informática

PROGRAMAÇÃO I – AULA 02

Disciplina: Linguagem de Programação PASCAL

E-mail: osorio@exatas.unisinos.br

Professor responsável: *Fernando Santos Osório*

Web:

Semestre: 2001/2

<http://inf.unisinos.br/~osorio/prog1.html>

Horário: 63

Xerox : Pasta 54 (Xerox do C6/6)

1. LINGUAGEM PASCAL – Tipos de Dados

A Linguagem Pascal possui três tipos de dados principais...

NUMÉRICOS (inteiros e reais), CARACTERES e LÓGICOS

* Tipos de dados: Caracteres:

Os CARACTERES podem armazenar letras ('a'..'z'; 'A'..'Z'), números ('0'..'9') ou outros caracteres especiais ('@', '#', '\$', '%', '*', '\', ...). Eles podem aparecer isolados (**tipo char**), ou agrupados formando palavras/frases (**tipo string**).

- Em Pascal nós usamos sempre um apóstrofe para indicar a existência de um texto e poder diferenciar de outros tipos de informações (números, comandos, nomes de variáveis).
Exemplos: 'Nome' (string) é diferente de Nome (variável)
 '1999' (string) é diferente de 1999 (número) '1' ≠ 1
 'begin' (string) é diferente de BEGIN (comando Pascal)
- String e Char podem conter qualquer tipo de caracteres, inclusive caracteres especiais, caracteres acentuados e espaços em branco.
- Dois apóstrofes seguidos dentro de um string indicam o caracter apóstrofe ('It''s Ok!').

Tipo CHAR: Serve para armazenar 1 único caracter. Exemplos:

```
VAR
  Sexo, Sim_ou_Nao:CHAR;
BEGIN
  Sexo:='M'; Sexo:='F'; Sim_Ou_Nao:='S'; Sim_ou_Nao:='N';
```

Tipo STRING: Serve para armazenar de 0 até 255 caracteres formando uma cadeia de caracteres (string). O tamanho do texto à ser armazenado no string pode variar, onde podemos armazenar desde uma string vazia (texto:= '');) até uma string de no máximo 255 caracteres. Exemplo:

```
VAR
  Nome, Telefone: STRING;
  Resposta: STRING[3];    { Tamanho máximo reservado na memória = 3 caracteres }
BEGIN
  Nome:='Fernando Osório';           Nome:='';
  Telefone:='(051) 590-3333';         Telefone:='5903333';
  Resposta:='Sim';                   Resposta:='Não';
```

Observações importantes sobre o tipo STRING:

- O Pascal controla sempre, de maneira automática, o tamanho da string (número de caracteres ocupados). Atenção: isso não garante que não possamos “transbordar” para fora do espaço previsto para a string. Este é um famoso problema do TurboPascal...
Exemplo: Resposta := ‘Talvez’; <= Problema grave (supondo a declaração feita acima)
- Para aqueles que já conhecem outras linguagens...
 - + O string é um vetor de caracteres, disfarçado como tipo String.
 - + Portanto nós podemos escrever Nome[1]:=’F’; (o primeiro caracter da string Nome recebe a letra ‘F’). Em compensação, se fazemos uso deste acesso direto as letras da string, isso faz com que o tamanho da mesma não seja atualizado corretamente pelo Pascal, o que pode levar a erros de representação da string.
 - + A posição 0 do vetor é um byte que armazena o tamanho da string:
 - Nome:= ‘Fernando Osório’; =====> Nome[0] possuirá o valor 15
 - Nome:= ‘’; =====> Nome[0] possuirá o valor 0
 - Nome[0]:=0; equivale ao comando imediatamente acima.

Manipulação (procedures/funções) de dados do tipo caracter:

⇒ Usados com strings:

- Length - Indica qual é o tamanho atual ocupado pela string (quantos caracteres tem)
- Copy - Copia um pedaço de uma string para uma outra string
- Concat - Concatena (gruda) duas strings gerando uma string que contém ambas
- Delete - Apaga um pedaço de uma string
- Insert - Insere uma string dentro de uma outra
- Pos - Procura um texto dentro de uma string indicando sua posição nesta
- Str - Converte um número em string
- Val - Converte uma string de números em um número comum

⇒ Usados com chars

- Ucase - Converte caracteres de letras minúsculas para maiúsculas (não funciona nos caracteres acentuados, somente nas letras de ‘a’ até ‘z’).
- Chr - Indica qual é a letra correspondente a um número do código ASCII. O código ASCII associa caracteres aos valores dos bytes contidos na memória do micro.
- Ord - Indica qual é o código ASCII correspondente a um certo caracter. Faz o inverso da função CHR (Chr : Número do código ASCII => Letra), ou seja ORD (Ord: Letra => Número do código ASCII).

2. ESTRUTURA DOS PROGRAMAS EM PASCAL

Os programas em Pascal tem uma estrutura bem organizada conforme segue a descrição abaixo:

```
PROGRAM <nome_do_programa>;
USES
  <nome_das_units>;
CONST
  <nome_da_constante> = <valor>;
TYPE
  <nome_do_novo_tipo_de_dados> = <definição_do_tipo>;
```

{ Comentário: Procedures e Functions serão estudadas mais tarde } (* Isto também é um comentário *)

```
PROCEDURE <nome_da_procedure> [ <lista_de_parâmetros> ] ;
VAR
  <declaração_das_variáveis_da_procedure>;
BEGIN
  <comandos_da_procedure>;
END;

FUNCTION <nome_da_função> [ <lista_de_parâmetros> ] : <tipo_retornado> ;
VAR
  <declaração_das_variáveis_da_função>;
BEGIN
  <comandos_da_função>;
  <nome_da_função> := <valor_retornado>;
END;
```

```
VAR
  <nome_da_variável> : <tipo_da_variável>;
BEGIN
  <comandos>;
END.
```

Exemplo de um programa completo em Pascal:

```
PROGRAM Hello_World;
USES
  Crt; { Unit CRT = Biblioteca de extensão das funções de tela }
CONST
  Mensagem = 'Hello World';
VAR
  Ano:Integer;
BEGIN
  clrscr; { Limpa a tela }
  Ano :=1999; { A variável Ano recebe o valor 1999 }
  writeln(Mensagem); { Escreve 'Hello 'World' na tela }
  writeln('Ano = ', Ano); { Escreve 'Ano = 1999' na tela }
  readln; { Espera que seja teclado enter para terminar o programa }
END.
```


WriteLn (0.123);	<==== Na tela: 1.2300000000E-01
Var_String:= 'Hello World';	
Write (Var_String);	<==== Na tela: Hello World_
Write (' ');	<==== Na tela: Hello World _
Write (Var_String);	<==== Na tela: Hello World Hello World
WriteLn (Var_String);	<==== Na tela: Hello World (pula para nova linha)

3.2 O comando Read/Readln:

Para ler uma entrada de dados via teclado **use apenas o READLN!** Explicação: quando você digita algo pelo teclado, termina pelo “enter” que indica ao computador que os dados de entrada estão prontos e disponíveis para o programa. O ReadLn lê corretamente os dados, ou seja, o dado seguido pelo “enter”, já o Read vai ler apenas o dado e o “enter” ficará esperando para ser lido na próxima leitura, gerando um comportamento esquisito no programa.

Tipos de comandos ReadLn – Leitura de dados do teclado

```
ReadLn;
ReadLn (<variável>);
ReadLn (<variável> , <variável> , ... );
```

Exemplos:

ReadLn;	<==== O programa fica parado até ser teclado um “enter”
ReadLn (Var_Int);	<==== Lê um valor inteiro seguido de “enter” (Um valor real causará um erro abortando o programa)
ReadLn (Var_Int1, Var_Int2);	<==== Lê um valor inteiro, seguido de um espaço em branco, seguido de um outro valor e terminando com “Enter”. Também funciona se for fornecido um valor, seguido de “enter” e o outro valor seguido de “enter”.
ReadLn (Var_Real);	<==== Lê um número real seguido de um “enter”.
ReadLn (Var_String);	<==== Lê uma string seguida de “enter”

⇒ Em Pascal, através da função padrão de entrada (ReadLn), não temos como limitar erros decorrentes de entradas incorretas feitas pelo usuário. Erros do tipo: entrada de textos no lugar de números, entrada de um número real no lugar de um inteiro, entrada de um texto maior que o esperado, ... são difíceis de evitar. A melhor solução é criar a sua própria ReadLn, mas para isso é preciso conhecer bem a linguagem Pascal e o Turbo Pascal.

⇒ Para terminar sugerimos ao aluno a consulta à:

- Lista de Procedures e Funções pré-definidas da Linguagem Pascal
- Tabela ASCII de caracteres
- Tabela dos caracteres especiais do Turbo Pascal

EXERCÍCIOS – AULA 02

1. Qual é o resultado da execução dos seguintes trechos de código abaixo? Responda conforme indicado.

1.1. Texto:= '123456789';	Indique o conteúdo de:
L := Length(Texto);	T1 :
T1:= copy(Texto, L-3, 3);	T2 :
T2:= concat(Texto,'-FIM');	L :

1.2. VR:= 1.25;	Indique o valor de:
VI:= 35;	R :
R:= ((VI DIV 3) – Trunc(VR)) Mod 4;	

2. Escreva o trecho de programa que obtenha o resultado pedido:

2.1. Colocar o valor arredondado do conteúdo da variável real Valor_Obtido em Valor_Estimado.

2.2. Atribuir à variável Raiz_Dado a raiz quadrada do valor absoluto (sem sinal) da variável Dado.

2.3. Obter a soma ponderada de A, B, C com pesos 3, 2, 1 respectivamente, colocando em SPond.

2.4. Sabendo que a variável real X contém o resultado da divisão de 2 números inteiros A e B, obtenha a partir do valor desta variável o valor que seria obtido através do comando A Mod B. Apenas o valor de B é conhecido e poderá ser usado no cálculo.

2.5. Dada uma variável Nome do tipo string, obtenha as cinco primeiras letras do nome contido nesta variável e coloque numa variável chamada Prim_Nome.

2.6. Dada uma variável Nome do tipo string, localize o caracter ponto (‘.’) dentro desta string e coloque o conteúdo da string que antecede o ponto (do início até o ponto) dentro de uma string chamada Nome_Arq, e coloque o conteúdo da string após o ponto (do ponto até o fim) dentro de uma string chamada Tipo_Arq.

2.7. Faça um programa que leia 2 números reais, calcule a raiz quadrada da soma do quadrado destes números e imprima na tela.

Exercícios de Prática de Programação - Programas Seqüenciais

1) Faça um programa para ler 2 números, sabendo-se que o usuário irá digitar os valores para nota1 e nota2 entre 0.0 e 10.0. Por enquanto não é necessário testar se os dados foram fornecidos corretamente. Calcular a média simples das duas notas, exibindo o resultado na tela.

2) Alterar o programa anterior para obter uma média ponderada, onde será considerado um peso 2 para a nota1 e um peso 3 para a nota2. O programa deverá exibir a seguinte saída na tela:

Digite a Nota1: 5.0 => Os valores sublinhados são fornecidos pelo usuário
 Digite a Nota2: 5.0
 A média do aluno é: **5.0** => O valor em negrito é a resposta obtida pelo programa
 Tecle <enter> para terminar a execução do programa...

3) Faça um programa para ler 1 número, calcular o valor da divisão inteira deste número por 7, e o respectivo resto desta divisão (resto da divisão inteira por 7). Exibir o resultado na tela conforme o exemplo abaixo:

Digite um número: 23 => O valor sublinhado é fornecido pelo usuário
23 dividido por **7 = 3** => Os demais valores são gerados pelo programa.
23 dividido por **7** resta **2** => O valor 7 é assumido como um valor *constante*

4) Altere o programa anterior, de maneira a ler dois números reais, um o dividendo e o outro o divisor. Em seguida exiba os seguintes dados na tela:

Dividendo: 10
 Divisor: 3
 Resultado da divisão: **3.3333**
 Parte inteira do resultado da divisão: **3**
 Parte fracionária do resultado da divisão: **0.3333**
 Resto da divisão das partes inteiras do dividendo pela parte inteira do divisor: **1**

5) Fazer um programa que leia 2 valores inteiros A e B, escreva os seus conteúdos na tela, e em seguida troque os conteúdos das duas variáveis, exibindo novamente os seus conteúdos na tela.

Valor de A: 53
 Valor de B: 34
 Os valores de A e de B são respectivamente **53** e **34**
 Após a troca o valor de A é igual a **34** e o valor de B é igual a **53**

6) Ler os seguintes dados:

- Nome: Nome do funcionário.
- HT: Número total de horas trabalhadas no mês.
- VH: Valor pago por hora de trabalho.
- PD: Percentual de descontos sobre o salário bruto total.

Exibir na tela as seguintes informações, calculadas à partir dos dados lidos:

- SB: Salário Bruto - Valor total das horas trabalhadas em função do valor por hora.
- TD: Total de descontos sobre o salário bruto.
- SL: Salário líquido final - Valor obtido à partir do salário bruto onde são aplicados os descontos.

- 7) Escreva um programa que leia o nome de uma pessoa (o usuário irá digitar no máximo 45 caracteres, onde os nomes fornecidos não irão ultrapassar este limite) e escreva na tela uma mensagem como segue:

Nome: Fulano

O nome Fulano tem **6** letras

Obs.: O número de letras é obtido automaticamente pelo programa

- 8) Escreva um programa que leia o nome de uma pessoa e escreva na tela o seu primeiro nome, conforme o exemplo abaixo:

Nome: Fulano da Silva e Siva

O primeiro nome de **Fulano da Silva e Siva** é... **Fulano**

- 9) Ler uma data no seguinte formato DD/MM/AA, onde DD indica o dia, MM indica o mês e AA indica o ano. Exibir na tela as informações da seguinte forma:

Entre com uma data (DD/MM/AA): 31/12/97

Dia: **31**

Mes: **12**

Ano: **1997**

- 10) Faça um programa que leia a dada atual (como o programa anterior) e mostre na tela quantos dias se passaram do início do ano até a data atual. O programa deve considerar a possibilidade de termos anos bissextos.