

Linguagem de Programação

Programação Pascal

Márcio F. Campos
Carlos A. Pilar

Versão 11 de julho de 2009

PARTE I.....	6
1 MEU PRIMEIRO PROGRAMA.....	7
1.1 OBJETIVO.....	7
1.2 RECURSOS DA LINGUAGEM.....	7
1.3 EXEMPLO:.....	7
1.4 PRÁTICA:.....	8
2 VARIÁVEIS.....	9
2.1 OBJETIVO.....	9
2.2 RECURSOS DA LINGUAGEM.....	9
2.3 EXEMPLO UM.....	9
2.4 PRÁTICA.....	9
2.5 EXEMPLO DOIS.....	10
2.6 PRÁTICA.....	10
3 ALGUMAS FUNÇÕES PRÉ-DEFINIDAS.....	11
3.1 OBJETIVO.....	11
3.2 RECURSOS DA LINGUAGEM.....	11
3.3 EXEMPLOS.....	11
3.4 PRÁTICA.....	11
4 ATRIBUIÇÃO DE VALORES.....	13
4.1 OBJETIVO.....	13
4.2 RECURSOS DA LINGUAGEM.....	13
4.3 EXEMPLO:.....	13
4.4 PRÁTICA:.....	14
5 CONSTANTES.....	15
5.1 OBJETIVO.....	15
5.2 RECURSOS DA LINGUAGEM.....	15
5.3 EXEMPLO:.....	15
5.4 PRÁTICA.....	16
5.5 DICA.....	16
6 VARIÁVEIS ALFANUMÉRICAS.....	17
6.1 OBJETIVO.....	17
6.2 RECURSOS DA LINGUAGEM.....	17
6.3 EXEMPLO.....	17
6.4 PRÁTICA.....	18
7 CONTROLES DO FLUXO.....	19
7.1 OBJETIVO.....	19
7.2 RECURSOS DA LINGUAGEM.....	19
7.3 EXEMPLO.....	19
7.4 PRÁTICA:.....	19
8 OPERADORES MOD E DIV.....	21
8.1 OBJETIVO.....	21
8.2 RECURSOS DA LINGUAGEM.....	21
8.3 EXEMPLO:.....	21
8.4 PRÁTICA:.....	22
9 LISTA DE EXERCÍCIOS.....	23
10 LAÇOS COM REPETIÇÃO FIXA E COM PRÉ-TESTE: O COMANDO FOR..DO.....	25

10.1 OBJETIVO.....	25
10.2 RECURSOS DA LINGUAGEM.....	25
10.3 EXEMPLOS:.....	25
10.4 PRÁTICA:.....	25
11 LAÇOS COM REPETIÇÃO INDETERMINADA E COM PRÉ-TESTE: O COMANDO DO...WHILE.	26
11.1 OBJETIVO.....	26
11.2 RECURSOS DA LINGUAGEM.....	26
11.3 EXEMPLO:.....	26
11.4 PRÁTICA:.....	26
12 LAÇOS III.....	27
12.1 OBJETIVO: APRESENTAR COMANDOS DE REPETIÇÃO REPEAT...UNTIL (LOOP).....	27
12.2 RECURSOS DA LINGUAGEM : PROGRAM, WRITE, WRITELN, DELAY, BEGIN/END; READ, READLN; REPEAT...UNTIL.....	27
12.3 EXEMPLO:.....	27
12.4 PRÁTICA:.....	27
13 EXERCÍCIOS LAÇOS.....	28
13.1 OBJETIVO: LISTA DE EXERCÍCIOS EM LOOP'S.....	28
13.2 RECURSOS DA LINGUAGEM: UTILIZE OS COMANDOS DE LOOP: FOR...DO, WHILE...DO, REPEAT...UNTIL.....	28
13.3 PRÁTICA:.....	28
14 SELEÇÃO MÚLTIPLA.....	29
14.1 OBJETIVO: APRESENTAR O COMANDO DE SELEÇÃO MÚLTIPLA. CASE...OF.....	29
14.2 RECURSOS DA LINGUAGEM: PROGRAM, WRITE, WRITELN, DELAY, BEGIN/END., CASE...OF.....	29
14.3 EXEMPLO:	29
14.4 PRÁTICA:.....	30
15 EXECÍCIOS LAÇOS II.....	31
15.1 OBJETIVOS: APRESENTAR PROBLEMAS DE LAÇOS MÚLTIPLOS(LOOP'S DENTRO DE LOOP'S).....	31
15.2 RECURSOS DA LINGUAGEM: PROGRAM, WRITLN, READLN, BEGIN, END, FOR...DO.....	31
15.3 EXEMPLO:.....	31
15.4 PRÁTICA:.....	31
16 DIFERENÇAS ENTRE OS TIPOS DE LAÇOS.....	32
16.1 OBJETIVO: DESTACAR AS PRINCIPAIS DIFERENÇAS ENTRE OS COMANDOS DE REPETIÇÃO, FOR...DO, REPEAT...UNTIL E WHILE...DO.....	32
16.2 RECURSOS DA LINGUAGEM: PROGRAM, BEGIN...END, FOR...DO, REPEAT...UNTIL E WHILE...DO.....	32
16.3 EXEMPLO:.....	32
16.4 PRÁTICA:.....	32
17 VARIÁVEIS MULTIDIMENSIONAIS.....	33
17.1 OBJETIVO: APRESENTAR O CONCEITO DE VARIÁVEIS MULTIDIMENSIONAIS (VETORES E MATRIZES).....	33
17.2 RECURSOS DA LINGUAGEM: PROGRAM, WRITE, READ, FOR...DO, DELAY,BEGIN..END.....	33
17.3 EXEMPLO:.....	33
17.4 PRÁTICA:.....	33
17.5 DESAFIO:.....	33
18 MATRIZES.....	34
18.1 OBJETIVO: APRESENTAR CONCEITOS DE MATRIZES.....	34
18.2 RECURSOS DA LINGUAGEM: PROGRAM, BEGIN..END, FOR...DO.....	34
18.3 EXEMPLO:.....	34
18.4 PRÁTICA:.....	34
18.5 DESAFIO:.....	34

19 PROCEDIMENTOS.....	35
19.1 OBJETIVO: APRESENTAR A UTILIZAÇÃO DE TRECHOS DE PROGRAMAS REUTILIZÁVEIS (PROCEDIMENTOS), SEM PASSAGEM DE PARÂMETROS.....	35
19.2 RECURSOS DA LINGUAGEM: PROGRAM, BEGIN...END, WRITELN, READLN, PROCEDURE.....	35
19.3 EXEMPLO:.....	35
19.4 PRÁTICA:.....	35
20 PASSAGEM DE PARÂMETRO.....	36
20.1 OBJETIVO: MOSTRAR PASSAGEM DE PARÂMETRO DE PROCEDIMENTOS, TANTO POR VALOR QUANTO POR REFERÊNCIA.	36
20.2 RECURSOS DA LINGUAGEM: PROGRAM, BEGIN...END, WRITELN, READLN, PROCEDURE.....	36
20.3 EXEMPLO:.....	36
20.4 PRÁTICA:.....	36
21 FUNÇÕES.....	37
21.1 OBJETIVO: MOSTRAR O CONCEITO DE FUNÇÕES.....	37
21.2 RECURSOS DA LINGUAGEM: FUNCTION <NOME DA FUNÇÃO>.....	37
21.3 EXEMPLO:.....	37
21.4 PRÁTICA:.....	37
22 ESCOPO DE VARIÁVEIS.	38
22.1 OBJETIVO: APRESENTAR O CONCEITO DE ESCOPO DE VARIÁVEIS.....	38
22.2 RECURSOS DA LINGUAGEM: VAR, PROCEDURE, FUNCTION, BEGIN...END.....	38
22.3 EXEMPLO:	38
22.4 PRÁTICA:.....	38
23 REGISTROS.....	39
23.1 OBJETIVO: APRESENTAR O CONCEITO DE REGISTRO.....	39
23.2 RECURSOS DA LINGUAGEM: TYPE, RECORD.....	39
23.3 EXEMPLO:	39
23.4 PRÁTICA:.....	39
24 REGISTROS.....	41
24.1 OBJETIVO: APRESENTAR O CONCEITO DE REGISTROS E COMANDOS PARA OPERAÇÕES COM ARQUIVOS TEXTOS.....	41
24.2 RECURSOS DA LINGUAGEM: ASSIGN, REWRITE, CLOSE.....	41
24.3 EXEMPLO:.....	41
24.4 PRÁTICA:.....	41
25 ARQUIVOS TEXTO.....	42
25.1 OBJETIVO: LER ARQUIVOS DO TIPO TEXTO, MOSTRANDO NA TELA OU NA IMPRESSORA OS RESULTADOS PESQUISADOS, TENDO COMO BASE O PROGRAMA DA AULA 24.....	42
25.2 RECURSOS DA LINGUAGEM: RESET, ASSIGN, CLOSE, READLN, EOF()	42
25.3 EXEMPLO:.....	42
25.4 PRÁTICA:.....	43
26 ARQUIVO DIRETO.....	44
26.1 OBJETIVO: APRESENTAR O CONCEITO DE ARQUIVO DIRETO (FILE OF.....)	44
26.2 RECURSOS DA LINGUAGEM: ASSIGN, RESET, REWRITE, FILE OF E CLOSE.....	44
26.3 EXEMPLO:.....	44
26.4 PRÁTICA:.....	44
27 ARQUIVO INDEXADO.....	45
27.1 OBJETIVO: INSERIR DADOS EM UM ARQUIVO DO TIPO FILE OF	45
27.2 RECURSOS DA LINGUAGEM: ASSIGN, RESET, WRITE, SEEK, FILESIZE, CLOSE.....	45
27.3 EXEMPLO:.....	45
27.4 PRÁTICA:.....	46

PARTE II.....	47
AULA UM - INICIAÇÃO.....	48
AULA DOIS - LITERAIS.....	49
AULA TRÊS – TIPO INTEIRO.....	50
AULA QUATRO – TIPO REAL.....	52
AULA CINCO – TIPO BOOLEANO.....	54
AULA SEIS – IF THEN ELSE.....	56
AULA SETE - TIPO RECORD.....	58
EXERCÍCIOS.....	59
AULA OITO – COMANDO FOR.....	67
AULA NOVE – COMANDO FOR.....	68
AULA DEZ – VETORES E MATRIZES.....	69
EXERCÍCIOS.....	70

Parte I

1 Meu Primeiro Programa.

1.1 Objetivo.

Apresentar a elaboração de um programa simples em pascal, mostrando a estrutura básica de um programa nessa linguagem.

1.2 Recursos da Linguagem.

A linguagem pascal possui um conjunto de palavras reservadas que são comandos executados pelo computador na resolução de um algoritmo escrito em uma linguagem de programação. As palavras reservadas não podem ser utilizadas de outra forma. Os principais comandos da linguagem pascal apresentados neste primeiro programa são.

Program: esta palavra reservada delimita o início do programa.

Write: esta comando escreve um conjunto de caracteres em um periférico de saída que pode

ser uma impressora, um arquivo ou em uma impressora.

Writeln: este comando faz exatamente aquilo que o comando write, porém faz pular uma linha.

Delay: aguarda um tempo fixo pré-determinado.

Begin/End: este par de comandos define o escopo de um conjunto de comandos/instruções.

{ } : o conjunto de pares de chaves possibilita escrever comentários ao longo do programa.

1.3 Exemplo:

```
program primeiro; {cabeçalho o programa}
uses crt;          {uso da unit crt}
begin             {declara o início do prg}
    clrscr;       {limpa a tela}
    write('alô mundo'); {escreve na tela sem mudar de
                        linha}
    delay(3000); {impede a execução imediata do próximo comando}
end.
```

Programa 1.1 - primeiro

1.4 Prática:

- a. Digite o programa acima no turbo pascal e faça-o executar.
- b. Aumente ou diminua o tempo do comando delay, compile, execute e verifique o que acontece.
- c. O comando textcolor(red) modifica a cor do texto. Escreva este comando antes do comando write. Compile, execute e verifique o que acontece. O pascal possui uma tabela de cores que podem ser usadas.
- d. O comando writeln escreve na tela mudando de linha. Modifique o programa acima de forma imprimir de 5 a 10 linhas sendo cada uma de uma cor diferente.
- e. Leia os programas abaixo e identifique as diferenças de resultado entre eles:

```
program aloMundo_a;      {cabeçalho o programa}
uses crt;
begin
    clrscr;
    writeln('alô mundo');
    delay(3000);
end.
```

Programa 1.2 - aloMundo_a.

```
program aloMundo_b;      {cabeçalho o programa}
uses crt;
begin
    clrscr;
    writeln;
    writel('alô mundo');
    delay(3000);
end.
```

Programa 1.3 - aloMundo_b.

2 Variáveis.

2.1 Objetivo.

Apresentar o conceito de variável, tipos e os comandos de leitura.

2.2 Recursos da Linguagem.

Foi visto na aula UM alguns recursos simples da linguagem pascal em particular da estrutura dos programas nesta linguagem e os comandos de escrita: *write* e *writeln*. Neste capítulo serão tratados os comando de leitura, *read* e *readln*, assim como a iniciação aos tipo de dados como o *integer*.

read: este comando faz a leitura de dados a partir da interface de entrada, que se for a entrada padrão será o teclado.

readln: este comando, à semelhança do *writeln* lê um valor, assim como o *read* e pula uma linha.

integer: um tipo de dado define os valores que uma variável poderá possuir assim como as operações que poder ser realizadas com esta.

variável: uma variável é quando um determinado símbolo é escolhido para guardar valores de um determinado tipo de dados, por exemplo *integer*. Estes valores podem ser alterados a qualquer momento desde que sejam do mesmo tipo.

2.3 Exemplo um.

```
program multiplica2;
uses crt;
var numero : interger;           {aqui é definida a variável numero}
begin
    write('digite um número inteiro qualquer');
    read(numero);
    write(numero*2);
    delay(3000);
end.
```

Programa 2.1 - multiplica2

2.4 Prática.

- Digite, compile e execute o programa acima.
- Modifique o programa acima de forma que o número digitado seja multiplicado por 3 (faça também para 4 ou 5). Não se esqueça de compilar e executar, logo após.
- Altere o programa acima os comandos *write* para *writeln* e *read* para *readln*. Compile e execute.

- d. Faça um programa que aceite dois valores e que dê como resultado a multiplicação dos dois.

2.5 Exemplo dois.

```
program soma;
uses crt;
var numero1, numero2 : integer;
begin
    writeln('digite o primeiro número');
    readln(numero1);
    writeln('digite o segundo número');
    readln(numero2);
    writeln(numero1 + numero2);
    delay(3000);
end.
```

Programa 2.2 - soma.

2.6 Prática.

- Escreva o programa acima no ambiente turbo pascal, compile-o e execute-o.
- Modifique o programa para ao invés de somar, subtraia, multiplique e divida (lembre-se de que não é possível a divisão por zero). Ao fazer a modificação do programa certifique-se de salvá-lo com outro nome.
- Experimente informar dois números inteiros acima de 30000. Verifique o que acontece!
- No caso do problema identificado acima, mude o tipo de *integer* para *longint* ou *real*.
- Execute os programas, 2.3 e 2.4 abaixo e veja o que acontece. Pesquise o que vem a ser a palavra *maxint*.

```
Program testeMaiorInteiro;
Begin
    writeln('o valor de maxint é ', maxint);
End.
```

Programa 2.3 - testeMaiorInteiro.

```
Program testeMenorInteiro;
Begin
    writeln('o valor de maxint é ', -maxint-1);
End.
```

Programa 2.4 - testeMenorInteiro.

3 Algumas Funções pré-definidas.

3.1 Objetivo.

Apresentar algumas funções pré-definidas do PASCAL.

3.2 Recursos da Linguagem.

Duas funções são apresentadas a SQR() e a SQRT ().

SQR (argumento): a função SQR() eleva ao quadrado o valor passado por parâmetro/argumento. O retorno desta função é do mesmo tipo do tipo do parâmetro passado por parâmetro e que pode ser inteiro (*integer*) ou real (*real*)

SQRT (argumento): a função SQRT calcula a raiz quadrada do valor passado por parâmetro/argumento. O retorno da função é sempre real, sendo que, os tipos que são aceitos como argumento podem ser inteiros ou reais não negativos.

3.3 Exemplos.

O exemplo 3.1 mostra a utilização da função SQR() enquanto que o programa 3.2 mostra a utilização da função SRQT().

```
program quadrado;
uses crt;
var numero : integer;
begin
    writeln('digite um número');
    readln(numero);
    writeln(sqr(numero));
    delay(3000);
end.
```

Programa 3.1 - quadrado.

```
program quadrado;
uses crt;
var numero : integer;
begin
    writeln('digite um número');
    readln(numero);
    writeln(sqrt(numero));
    delay(3000);
end.
```

Programa 3.2 - raiz quadrada.

3.4 Prática.

a. Digite o programa acima. Compile-o e execute-o.

- b. Experimente, no programa 3.2, digitar um número negativo. Lembre-se que não existe a raiz quadrada de números negativos no conjunto dos números reais.
- c. Calcule o tamanho da hipotenusa de um triângulo retângulo, sabendo-se que esta é calculada a partir de: raiz quadrada de $a^2 + b^2$.
- d. A energia cinética de um objeto em movimento é calculada pela seguinte fórmula: $E_c = (1/2)mv^2$. Faça um programa que dado a massa este calcule a energia cinética.

4 Atribuição de valores.

4.1 Objetivo.

Apresentar o comando de atribuição de valores à variáveis.

4.2 Recursos da Linguagem.

Sinal de atribuição, " := ": este sinal é utilizado para copiar os valores contidos nas variáveis ou expressões à direita do sinal para a variável à esquerda do mesmo.

4.3 Exemplo:

O programa 4.1 calcula a área do quadrado e atribui este valor à variável *area* que guarda este valor para depois ser mostrada através do comando *writeln*.

```
program area_quadrado;
uses crt;
var lado, area      : integer;
begin
    writeln ('digite o tamanho lado do quadrado');
    readln(lado);
    area := lado * lado;           {ou area := sql(lado)}
    writeln('este é o valor da área do quadrado', area);
    delay(3000);
end.
```

Programa 4.1 - área_quadrado.

Considere o programa 4.2. Qual será o valor de *sum* ao final do programa? 70, correto!

```
program valorFinaldeSum;
var sum      :integer;
begin
    sum := 10;
    sum := 20;
    sum := 70;
    writeln('o valor final de sum é' , sum);
end.
```

Programa 4.2 - valorFinaldeSum.

Agora qual o valor final de *sum* e *tum*?

```
program valorFinaldeSumTum;
var sum, tum      :integer;
begin
    sum := 10;
    tum := 20;
    sum := tum;
    sum := 60
    writeln('o valor final de sum é' , sum);
    writeln('o valor final de tum é' , tum);
end.
```

Programa 4.3 - valorFinaldeSumTum.

Bem, neste caso o valor final de sum é 60 e de tum é 20.

4.4Prática:

- a. Digite, compile e execute o programa 4.1.
- b. Faça um programa que calcule a área do retângulo (base x altura).
- c. Faça um programa que calcule a área do triângulo (base x altura)/2.
- d. Faça um programa que calcule a área do trapézio ((base maior + base menor) x altura)/2.

5 Constantes.

5.1 Objetivo.

Apresentar o conceito de constante.

5.2 Recursos da Linguagem.

const: assim como a palavra reservada *var* inicia a declaração das variáveis a declaração *const* inicia a definição de constantes. Como o nome já diz os identificadores definidos sob esta declaração não poderão alterar os seus valores.

5.3 Exemplo:

```
program area_circulo;
uses crt;
const
    pi = 3.1415;      {definição da constante}
var
    area, raio: real; {Observe! Duas variáveis do tipo real sendo
                      declaradas ao mesmo tempo}
begin
    writeln('digite o raio do círculo');
    readln(raio);
    area := pi * sqr(raio); { o valor de pi é igual a 3.1415}
    writeln(area);
    writeln('o valor de pi é', pi);
    delay(3000);
end.
```

Programa 5.1 - área_circulo.

```
program graus;
uses crt;
const
    converte_far_celsius = 0.55 {definição da constante}
var
    fahrenheit, celsius: real;
begin
    writeln('digite a temperatura em graus fahrenheit');
    readln(fahrenheit);
    celsius := converte_far_celsius * (fahrenheit -32);
    writeln ('a temperatura em celsius é ', celsius);
    delay(3000);
end.
```

Programa 5.2 - graus.

5.4Prática.

- a. Faça um programa que solicite ao usuário o valor do raio de forma a calcular o volume de uma esfera (bola de frescobol), cuja fórmula é $\text{Volume} = \frac{4}{3} * \pi * R^3$ (onde R é o raio). Dica: use o comando const.
- b. Faça um programa em que dado uma valor em celsius, obtenha-se um valor em fahrenheit, dado a fórmula $\text{far} = 1.888 * \text{celsius} + 32$. Dica use o comando const.
- c. Faça uma programa que solicite o preço de um produto e este seja majorado em 20%. Dica use o comando const.

5.5Dica.

Como visto os tipo desempenham o papel de informar ao programa como serão interpretados os bytes associados às variáveis. Porém cada tipo possui um escopo de valores. Ex:

Tipo	escopo
Const	: valor único.
Byte	: 0 a 255
Integer	: -32768 a 32767
Longint	: -2147483648 a 2147483647
Shortint	: -128 a 127
Real	: qualquer número real (com precisão de 11 casas após a vírgula).

6 Variáveis Alfanuméricas.

6.1 Objetivo.

Apresentar o conceito de variáveis alfanuméricas.

6.2 Recursos da Linguagem.

String: uma variável alfanumérica é definida com o tipo *String*. Desta forma os valores aceitos pelas variáveis definidas para este tipo são, por exemplo, os seguintes: " ", "@", "João", "\$\$\$\$\$\$\$\$\$\$\$\$\$". Assim qualquer valor entre aspas ou apóstrofo é um literal ou valor aceito por uma variável do tipo *String*.

Char: por sua vez uma variável do tipo *char* corresponde a um único elemento de uma *String*. Assim os valores aceitos para este tipo de variável são, por exemplo: "a", "b", "%" etc.

6.3 Exemplo.

```
program caracteres;
uses crt;
const
    bem_vindo = 'seja bem vindo a escola';
var
    nome : string;
begin
    writeln('digite o seu nome');
    readln(nome);
    writeln(bem_vindo + nome);    { o comando + concatena strings}
    delay(3000);
end.
```

Programa 6.1 - caracteres.

```
program caracteres2;
uses crt;
var
    letraminuscula, letramaiuscula : char; {equivale ao string de
                                             apenas uma posição}
begin
    writeln('digite qualquer letra minuscula');
    readln(letraminuscula);
    letramaiuscula := upcase(letraminuscula); {a função upcase
                                                transforma letras minusculas em letras maiusculas}
    writeln(letramaiuscula);
    delay(3000);
end.
```

Programa 6.2 - caracteres2.

6.4Prática.

- a. Faça um programa que aceite uma frase em letras minúsculas de forma a transformá-las em letras maiúsculas.
- b. Faça um programa que aceite o nome e o sobrenome de uma pessoa e imprima-os invertidos (sobrenome e nome).
- c. A função `length(string)` fornece o número de caracteres da variável informada. Faça um programa que calcule o tamanho, em letras, de uma frase.

7 Controles do fluxo.

7.1 Objetivo.

Apresentar controles de fluxo de programa.

7.2 Recursos da Linguagem.

If <expressão relacional> then <comando>; ou If <expressão relacional> then <comando>;: o comando *if-then-else*, como é conhecido, possibilita o desvio de forma condicional das instruções do programa. Um dos aspectos chaves deste comando é a *expressão relacional* associada ao mesmo. A expressão relacional retorna dois valores: *verdadeiro* ou *falso*. Quando o retorno é verdadeiro o programa executa as instruções associadas ao *then*. Quando o retorno da expressão é *falso*, executa-se os comando associados ao *else*, quando este estiver definido.

7.3 Exemplo.

```
program maior_que_dez;
uses crt;
var valor : interger;
begin
    writeln('digite um valor');
    readln(valor);
    if valor > 10 then writeln ('voce digitou um valor maior que
10');
    delay(3000);
end.
```

Programa 7.1 - maior_que_dez

```
program medias;
uses crt;
var nota1, nota2, media : real    {olhe o tipo! Com inteiro pode dar
problema}
begin
    writeln('digite a sua primeira nota')
    readln(nota1);
    writeln('digite a sua Segunda nota')
    readln(nota2);
    media := (nota1+nota2)/2; {olha o parenteses!}
    if media > 5 then writeln('parabens voce passou!')
    else writeln('bem vindo ao curso de verão'); { repare o ponto e
virgula apenas no final do else!}
end.
```

Programa 7.2 - medias

7.4 Prática:

a. Digite os programas acima e execute-os no ambiente turbo pascal.

- b. Faça um programa que calcule o valor de Δ (delta) em uma equação de segundo grau, informando o seu valor caso seja positivo, caso contrário dar mensagem de Δ negativo. Lembre-se, $\Delta = b^2 - 4ac$.
- c. Faça um programa que calcule a equação do segundo grau, sendo dados os coeficientes a, b e c. Lembre-se $(-b \pm \sqrt{\Delta})/2a$.

8 Operadores mod e div

8.1 Objetivo.

Apresentar os operadores MOD e DIV.

8.2 Recursos da linguagem.

MOD e DIV: Os inteiros possuem um conjunto de operadores aritméticos já conhecidos tais como +(soma), -(subtração), *(multiplicação). A divisão não faz parte deste operadores pois o seu resultado pode não ser um inteiro. Desta forma, a linguagem Pascal, assim como outras linguagens oferecem duas outras operações que resultam em inteiros. O operador MOD resulta no resto da divisão e DIV no quociente.

8.3 Exemplo:

```
program e_par?;
uses crt;
var numero, resto : integer;
begin
    writeln('digite um número inteiro');
    readln(numero);
    resto := numero mod 2; {a função mod fornece o resto de uma
    divisão inteira}
    if resto = 0 then
    begin {quando o if/then tem mais de um
        comando deve-se usar os comandos begin/end}
        writeln('o numero digitado e par');
        delay(3000);
    end {quando existe else após o end, não se usa ;}
    else
    begin
        writeln ('o número é ímpar');
        delay(3000);
    end; {fim do else}
end.
```

Programa 8.1 - e_par?

```
program dividindoMesada;
uses crt;
var mesada, filhos, mesada_filho: integer;
begin
    writeln('informe o total da mesada no mês');
    readln(mesada);
    writeln('informe o número de filhos');
    readln(filhos);
    mesada_filho := mesada div filhos;
    writeln('este é o valor da mesada_filhos ', mesada_filhos);
    delay(3000);
end.
```

Programa 8.1 - dividindoMesada

8.4Prática:

- a. Digite os programas acima e faça-os executar em turbo pascal.
- b. Elabore um programa em que dado um ano, verifique se este é bissexto ou não (divida-o por 4 e verifique o resto).

9 Lista de Exercícios.

- a. Deseja-se calcular o consumo de um automóvel. Sabe-se a quilometragem inicial(KI) e a final(KF) e o total de litros consumidos(TLC). Para se obter o consumo em Km/L deve-se usar a fórmula: $(KF-KI)/TLC$.
- b. Elabore um programa para calcular o perímetro do círculo. Deve-se solicitar o raio (r). O cálculo do perímetro é dado por $Perímetro(p) = 2\Pi r$, onde Π (pi) é uma constante de valor 3.14.
- c. Sabendo-se que um automóvel percorreu 500m, faça um programa que solicite um tempo de percurso de forma a calcular a velocidade média. Sabe-se que $velocidade\ média = espaço / tempo$.
- d. Sabendo-se que um automóvel atinge a velocidade de 100 m/s em t segundos, calcule a aceleração, onde $aceleração = velocidade / tempo$. O programa deve mostrar as diferentes acelerações para cada uma dos tempos informados.
- e. Deseja-se dar um aumento para um grupo de funcionários de uma empresa. A regra básica é a que se segue. Caso o salário seja maior ou igual que 1500 reais, então terá aumento de 10%; caso contrário 15%. O programa deve aceitar um salário e calcular o respectivo aumento.
- f. Deseja-se classificar um grupo de pessoas conforme a faixa de idade, da forma de se segue:

< 13 anos: infantil.

de 13 aos 19 : adolescente.

de 20 aos 35: jovem.

de 36 aos 60: meia idade.

61 em diante: idoso.

- g. Um hospital local necessita de um programa que calcule o pagamento devido por cada paciente. Os custos para cada paciente são os seguintes:

De quarto:

privado R\$200,00;

semi-privado R\$100,00;

coletivos R\$50,00;

Telefone: R\$10,00 ao dia.

Televisão: R\$15,00 ao dia.

Ar-condicionado: R\$30,00 ao dia.

A entrada de dados necessitará da quantidade de dias no hospital, o tipo de quarto, as opções de telefone, televisão e ar-condicionado. A saída deverá ser do seguinte modo:

Hospital Local

Recibo do Paciente

Número de dias no hospital: 3

Tipo de quarto: Privado.

Total quarto: 600,00

Total Telefone: 30,00.

Total Televisão: 45,00.

Total ar-condicionado: 90,00

Total devido: 755,00.

10 Laços com repetição fixa e com pré-teste: o comando *For..Do*.

10.1 Objetivo.

Apresentar comandos de repetição for...do (loop).

10.2 Recursos da Linguagem.

For..DO: Invariavelmente necessita-se repetir um trecho de código seguidamente, como por exemplo, em um somatório onde é necessário somar seguidamente várias parcelas. O comando *For..Do* realiza a repetição um número fixo de vezes. A sintaxe básica é: FOR <indexador>:= valor inicial TO <valor final> DO <comandos>. Neste caso <comando> pode ser um único comando ou um conjunto de comandos. O indexador é somado de um até se igualar ao valor final, quando a repetição se encerra.

10.3 Exemplos:

```
program mostraFor;
uses crt;
const max = 20;
var incremento    : integer; {deve ser uma variável inteira!}
begin
    for incremento := 1 to max do { o begin/end do for é executado
max vezes; a variável incremento é atualizada automaticamente até
atingir o número max }
        begin
            writeln('este é o valor do incremento dentro do loop',
incremento);
            delay(3000);
        end;
end.
```

Programa 10.1 - mostraFor

10.4 Prática:

- Idem para a raiz quadrada dos 20 primeiros números inteiros. Dica: use a função sqrt.
- Idem para o quadrado dos 20 primeiros números inteiros. Dica: use a função sqr.
- Faça um programa que solicite ao usuário digitar 10 números distintos, tendo como resposta do mesmo o maior deles. Idem para o menor.

11 Laços com repetição indeterminada e com pré-teste: o comando *Do...While*.

11.1 Objetivo.

Apresentar comandos de repetição *Do...While*(loop).

11.2 Recursos da Linguagem.

Do...While: assim como *For...Do* utiliza-se este comando para repetir um trecho de código seguidamente. Entretanto, sua utilização está associada a uma repetição indeterminada, ou seja, não se sabe de antemão quantas vezes a repetição será executada. A sintaxe do *Do...While* é

11.3 Exemplo:

```
program mostraWhile;
uses crt;
var incremento    : integer;
begin
  incremento := 1;      {atenção ! }
  while incremento <= 20 do
  begin
    writeln('o incremento está valendo', incremento);
    delay(3000);
    incremento := incremento + 1 {atenção !}
  end; {fim do enquanto}
end.
```

Programa 11.1 - mostraWhile.

11.4 Prática:

- a. Faça um programa que imprima os 20 primeiros pares a partir do 0(zero). Dica: use a função mod. Idem para os ímpares.
- b. Idem para os primos. Lembre-se que os números primos são aqueles divididos por si mesmos e por 1

12 Laços III.

12.1 Objetivo: apresentar comandos de repetição repeat...until (loop).

12.2 Recursos da Linguagem : Program, Write, Writeln, Delay, Begin/End; read, readln; repeat...until.

12.3 Exemplo:

(prg12a)

```
program mostra_repeat;
uses crt;
var incremento    : integer;
begin
    incremento := 1;    {atenção ! }
    repeat
        writeln('o incremento está valendo', incremento);
        delay(3000);
        incremento := incremento + 1 {atenção !}
    until keypressed; { observe que o teste é feito ao final; isto
quer dizer que o programa entra no loop pelo menos uma única vez}
end.
```



12.4 Prática:

- Faça um programa que imprima os 20 primeiro pares a partir do 0 (zero). Dica: use a função mod.
- Idem para os ímpares
- Faça um programa que imprima os 20 primeiro pares a partir do 0 (zero). Dica: use a função mod.
- Idem para os ímpares.
- Idem para os primos. Lembre-se que os números primos são aqueles divididos por si mesmos e por 1
- Idem para a raiz quadrada dos 20 primeiros números inteiros. Dica: use a função sqrt.
- Idem para o quadrado dos 20 primeiros números inteiros. Dica: use a função sqr.
- Faça um programa que solicite ao usuário digitar 10 números distintos, tendo como resposta do mesmo o maior deles.
- Idem para o menor.

13 Exercícios Laços.

13.1 Objetivo: lista de exercícios em loop's.

13.2 Recursos da Linguagem: utilize os comandos de loop: for...do, while...do, repeat...until.

13.3 Prática:

- Em um frigorífico deseja-se verificar qual o boi mais gordo e o mais magro. Cada boi possui um número de identificação. Na medida que o boi passa na balança um funcionário digita o peso e o número de identificação do animal. Sabendo-se que serão pesados 25 bois, faça um programa que imprima o número de identificação do boi mais pesado e do mais leve.
- Deseja-se mostrar na tela a tabuada de um determinado número. Faça um programa em que dado um número, apresente a respectiva tabuada (1 a 10) .
- Dada a seguinte equação: $f(x) = 2x + 1$. Faça um programa que verifique o valor de $f(x)$ para os valores de x maiores que -4 e menores que 15 .
- Dada uma equação do segundo grau, elabore um programa que solicite os coeficientes a, b e c de forma que se calcule o Δ e caso o Δ seja positivo ou zero, calcule e imprima os vértices ($x_v = -b/(2a)$ e $y_v = -\Delta/(4a)$). Deve-se testar os resultados para cinco equações diferentes. Dica utilize um loop.
- Faça um programa que mostra a tabuada exponencial do número 5. Calcule o valor de 5 elevado aos expoentes de 0 a 20.

14 Seleção Múltipla.

14.1 Objetivo: apresentar o comando de seleção múltipla. **Case...of.**

14.2 Recursos da Linguagem: Program, Write, Writeln, Delay, Begin/End., Case...of.

14.3 Exemplo:

(prg14a)

```
program case_of;
uses crt;
var  operador : char;
     operando1, operando2 : real;
begin
  writeln('digite o operando 1');
  readln(operando1);
  writeln('digite o operando 2');
  readln(operando2);
  writeln('digite qual a operação desejada: +, -, *, /');
  readln(operador);
  case operador of
    '+' : writeln(operador1 + operador2);
    '-' : writeln(operador1 - operador2);
    '*' : writeln(operador1* operador2);
    '/' : writeln(operador1 / operador2);
  end; {case}
end.
```



(prg14b)

```
program case_of_else;
uses crt;
var  operador : char;
     operando1, operando2 : real;
begin
  writeln('digite o operando 1');
  readln(operando1);
  writeln('digite o operando 2');
  readln(operando2);
  writeln('digite qual a operação desejada: +, -, *, /');
  readln(operador);
  case operador of
    '+' : writeln(operador1 + operador2);
    '-' : writeln(operador1 - operador2);
    '*' : writeln(operador1* operador2);
    '/' : writeln(operador1 / operador2);
  else: writeln('operador inválido');
  end; {case}
end.
```



14.4Prática:

- Faça um programa que apresente um menu de opções para o cálculo da área das seguintes figuras geométricas: (a) quadrado, (b) retângulo, (c) triângulo, (d) trapézio e (e) círculo.
- Um banco classifica seus clientes segundo a média do saldo mensal, conforme a tabela abaixo:

Média do saldo mensal	status
< 500	uma estrela
de 500,01 a 1.500	duas estrelas
de 1.500,01 a 5.000	três estrelas
acima de 5.000,01	vip

Faça uma programa que aceite o saldo médio do cliente e imprima em qual status este se encontra.

15 Execícios Laços II.

15.1Objetivos: apresentar problemas de laços múltiplos(Loop's dentro de loop's).

15.2Recursos da linguagem: program, writln, readln, begin, end, for...do.

15.3Exemplo:

(Prg15a)

```
program tabuada;
uses crt;
var i,j      : integer;
begin
    for i := 1 to 9 do
    begin
        clrscr;
        writeln('esta é a tabuada de', i);
        for j := 1 to 9 do
        begin
            writeln(i, " x ",j, " = ", i*j);
        end; {fim do j}
        writeln('para a proxima tabela digite qualquer
tecla');
        readkey();
    end; {fim do i}
end.
```



15.4Prática:

- Faça um programa que usando-se um loop dentro de outro, como o do exemplo 14.1, em que mostre quando o valor de $i > j$.
- Faça uma tabuada que informe mostre os valores de 1 a 20.
- Faça um programa que mostre uma tabuada dado uma faixa de valores.
- Faça um programa que calcule quantos triângulos retângulos existem, dado que os lados podem variar na faixa de valores de 1 a 25. Dica: use o teorema de Pitágoras ($a^2 = b^2 + c^2$); um triângulo só existe se cada lado for menor que a soma de outros dois; atenção, este programa necessita de 3 loop's embutidos um dentro do outro!
- Idem para o problema acima, exceto que para a fórmula $a^3 = b^3 + c^3$. Sendo que neste caso esta fórmula não está associada a nenhuma figura geométrica!

16 Diferenças entre os tipos de laços.

16.1 Objetivo: destacar as principais diferenças entre os comandos de repetição, `for...do`, `repeat...until` e `while...do`.

16.2 Recursos da Linguagem: `program`, `begin...end`, `for...do`, `repeat...until` e `while...do`.

16.3 Exemplo:

FOR	WHILE	REPEAT
<pre>program forexemp; var i: integer; begin for i := 1 to 10 do begin writeln(3*i); delay(1000); end; end.</pre>	<pre>Program whileexemp; Var i : integer; Begin i:= 1; while i < 11 do begin writeln(3*i); delay(1000) i := i+ 1; end; end.</pre>	<pre>Program repeatexemp; Var i : integer; Begin i:= 1; repeat writeln(3*i); delay(1000); i := i +1 until i > 10 end.</pre>
<p>O comando <code>for...do</code> deve ser utilizado quando se sabe a priori o número de repetições que se deseja. Atenção: no <code>for...do</code> a variável de controle (i) é atualizada automaticamente.</p>	<p>O <code>while...do</code> faz um teste de condição no início do loop, caso esta seja verdadeira a repetição é executada (<code>begin..end</code> do <code>while</code>), caso seja falsa o controle do programa passa para a primeira instrução após o <code>begin..end</code> do <code>while</code>. Atenção: observe que a variável de controle (i) deve ser atualizada dentro do próprio loop.</p>	<p>O <code>repeat...do</code> permite a execução de pelo menos uma única vez dentro do corpo do comando, já que o teste do controle do loop ocorre apenas ao final do mesmo. Repare que o corpo do comando <code>repeat...do</code> é executado enquanto a condição do <code>until</code> é falsa. Ao se tornar verdadeira o loop é encerrado. Atenção: semelhantemente ao <code>while...do</code>, a variável de controle deve ser atualizada dentro do próprio loop.</p>

16.4 Prática:

- Digite e execute os programas acima.
- Para os programas acima, faça as seguintes alterações:
 - ❑ No programa FOR faça listar a tabuada de 1 a 20.
 - ❑ No programa WHILE inicialize a variável de controle i com 11.
 - ❑ No programa REPEAT inicialize a variável de controle i com 11.

17 Variáveis Multidimensionais

17.1 Objetivo: apresentar o conceito de variáveis multidimensionais (vetores e matrizes).

17.2 Recursos da linguagem: program, write, read, for...do, delay, begin..end.

17.3 Exemplo:

(prg17a)

```
program vetor_manual;
uses crt;
const nmax = 5;
type
    vetor = array[1..nmax] of integer;
var
    i: integer;
    vet: vetor;
begin
    for i := 1 to nmax do
        begin
            writeln('entre como número');
            readln(vet[i]); {preenchendo o vetor, na posicao i}
        end;
    end.
end.
```



17.4 Prática:

- Faça um programa que imprima o vetor acima e na ordem inversa. É necessário digitar o programa acima.
- Idem acima, so que imprime na ordem correta da entrada de dados.
- Faça um programa que preencha um vetor de 10 posições, apenas com números pares.
- Idem para ímpares.
- Faça um programa que aceite uma palavra de até 10 caracteres e imprima na ordem inversa. A entrada da palavra deve ser de caracter a caracter.
- Acione ao programa acima a seguinte funcionalidade: imprimir a posição i do vetor onde ocorrer uma letra informada. Faça uma menu de opções.

17.5 Desafio:

- Faça um programa que, dado um nome e o sobrenome de um indivíduo, imprima o sobrenome e o nome, nesta ordem! Ex: Rui Barbosa , o programa imprime Barbosa, Rui.

18 Matrizes

18.1 Objetivo: apresentar conceitos de matrizes.

18.2 Recursos da Linguagem: program, begin..end, for...do.

18.3 Exemplo:

(prg18a)

```
program matriz;
uses crt;
const nlinmax = 10; ncolmax = 10;
type matriz = array[1..nlinmax, 1..ncolmax];
var i, j : integer;
    minha_matriz : matriz;

begin
  for i := 1 to nlinmax do
  begin
    for j := 1 to ncolmax do
    begin
      randomize;
      valor := random(j);
      minha_matriz[i,j] := valor;
    end; {end fo for j}
  end; {endo do for i}
end. {end do pgm}
```



18.4 Prática:

- Digite e execute o programa acima do exemplo18.1
- Atualize o programa acima para que este leia uma matriz randômica e a imprima ;
- Faça uma programa que imprima apenas os valores da matriz cujos os números da linha sejam iguais aos números das colunas.
- Faça uma programa que imprima apenas os valores da matriz cujos os números da linha sejam iguais ou menores aos números das colunas.
- Faça uma programa que imprima apenas os valores da matriz cujos os números da linha sejam iguais ou maiores aos números das colunas.

18.5 Desafio:

- Faça um programa que preencha automaticamente uma matriz 11x11 sendo que toda a coluna 6 e linha 6 devem conter o número 1, de outra forma deve conter 0 (forma de cruz). O programa deve, também, imprimir a respectiva matriz.
- Faça um programa que preencha automaticamente uma matriz 11x11 sendo que deve ser formado uma estrela. Tenha como base as linhas 3 e 9 e as coluna 6. O programa deve, também, imprimir a respectiva matriz.

19 Procedimentos.

19.1 Objetivo: apresentar a utilização de trechos de programas reutilizáveis (procedimentos), sem passagem de parâmetros.

19.2 Recursos da Linguagem: program, begin...end, writeln, readln, procedure.

19.3 Exemplo:

(prg19a)

```
program procedimento;
uses crt;
var
    lado, base, altura, area : real; {variáveis globais, vistas por
todo o pgm}
    opcao : integer;
procedure quadrado;
begin
    writeln('digite o lado ');
    readln(lado);
    area := lado*lado;
    writeln(area:4:2);
    delay(1000);
end; {fim do procedure quadrado}

procedure retangulo;
begin
    writeln('digite a base e altura');
    readln(base, altura);
    area := base * altura;
    writeln(area:4:2);
    delay(1000);
end; {fim do procedimento retângulo}

Begin {inicio do programa principal!!!}
Writeln('escolha qual a sua opção');
Writeln('1 - quadrado');
Writeln('2 - retângulo');
Readln(opcao);
Case opcao of
    1 : quadrado;
    2: retangulo;
    else writeln('opcao invalida');
end; {fim do case}
end. {end do pgm}
```



19.4 Prática:

- Digite e execute o programa acima (19.1).
- Adicione ao programa acima os procedimentos para cálculo da área do triângulo, trapézio, círculo e losango. Insira no programa uma opção para saída.
- Faça um programa que implemente uma calculadora. Para cada operação utilize um procedimento diferente.

20 Passagem de parâmetro.

20.1 Objetivo: mostrar passagem de parâmetro de procedimentos, tanto por valor quanto por referência.

20.2 Recursos da Linguagem: program, begin...end, writeln, readln, procedure.

20.3 Exemplo:

(prg20a)

```
program procedimento2;
uses crt;
var
    a,b,c : integer; { estas variáveis são globais }

procedure incrementa (x:integer, var y:integer);
    { Em x a passagem do parâmetro é dita por valor, ou seja, e
    passado uma cópia do valor da variável para o procedimento. Em y a
    passagem é por referência, ou seja, é passado um endereço de memória,
    sendo assim se o valor da variável for alterado no procedimento este
    volta alterado ao programa principal; x e y são os parâmetros do
    procedimento. }
begin
    x := 2*x + y;
    y := 3*x;
    writeln('este é o valor de x', x);
    writeln('este é o valor de y', y);
    delay(2000);
end; { fim do procedimento}

begin { inicio do programa principal}
    a := 2;
    b:= 3;
    writeln('este é o valor de a' , a);
    writeln('este é o valor de b', b);
    incrementa(a,b);
    writeln('este é o valor de a' , a);
    writeln('este é o valor de b', b);
end.
```



20.4 Prática:

- Reescreva os programa de cálculo da área do triângulo e retângulo passando a base e a altura por valor, usando procedimentos.
- Reescreva os programa de cálculo da área do triângulo e retângulo passando a base e a altura por referência, usando procedimentos.

21 Funções

21.1 Objetivo: mostrar o conceito de funções.

21.2 Recursos da Linguagem: function <nome da função>

21.3 Exemplo:

(prg21a)

```
program eleva_ao_cubo;
uses crt;
var
    x      : real;
function cubo(x :real);
begin
    cubo := x*x*x;
end; { fim da função}

Begin
    Writeln('digite um número');
    Readln(x);
    Writeln('este é o cubo de x: ', cubo(x));
End.
```



21.4 Prática:

- Elabore uma função que solicite um número e eleve ao expoente 15 (use o comando `exp(ln)`).
- Elabore uma função em que solicite uma cadeia de caracteres imprima os 5 primeiros.
- Dado a fórmula matemática $f(x) = x^2 - 3x + 5$. Faça uma programa que calcule o $f(x)$. Tendo como base o programa acima faça o mesmo para as seguintes funções matemáticas:

$$f(x) = 2x + 5$$

$$f(x) = x^7 + 1$$

$$f(x) = e^3$$

22 Escopo de variáveis.

22.1 Objetivo: apresentar o conceito de escopo de variáveis.

22.2 Recursos da Linguagem: var, procedure, function, begin...end.

22.3 Exemplo:

(prg22a)

```
program escopo_de_variáveis;
var
    global_x, global_y : integer; {variáveis globais, visíveis em
todo o programa}
procedure incrementa;
var local_x : integer; {esta variável só é vista dentro do
procedimento incrementa}
begin
    local_x := 3;
    local_x := local_x + 1;
    global_x := local_x;
    writeln('local_x = ', local_x);
    writeln('global_x = ', local_x);
    delay(1000);
end;

procedure troca (var x : integer, y : integer); { x é passado por
referência e y por valor}
var temp :integer;
begin
    temp := x;
    x := y;
    y := temp;
    writeln('dentro do procedimento x é igual a ', x);
    writeln('dentro do procedimento y é igual a ', y);
    delay(1000);
end;

begin {programa principal}
    writeln('digite dois números:');
    readln(global_x);
    readln(global_y);
    incrementa;
    troca(global_x, global_y);
    writeln('global_x = ', global_x);
    writeln('global_y = ', global_y);
end.
```



22.4 Prática:

- Copie e execute o programa acima. Observe o que acontece com cada variável.
- Adicione ao programa acima o seguinte comando: `writeln(local_x)`, dentro do programa principal. Execute-o.

23 Registros.

23.1 Objetivo: apresentar o conceito de registro.

23.2 Recursos da Linguagem: type, record.

23.3 Exemplo:

(prg23a)

```
program cadastro;
const
    nmax = 5;
type
    ficha = record
        nmat      : integer;
        nome      : string;
    end; { observe que na descrição do registro existe um end sem
begin}
    vetor = array[1..nmax] of ficha;
var
    vetfichas   : vetor;
    i           : integer;
procedure inclui;
begin
    for i := 1 to nmax do
        begin
            writeln('digite o número de matrícula do aluno');
            readln(vetfichas[i].nmat);
            writeln('digite o nome do aluno');
            readln(vetfichas[i].nome);
        end; {fim do for}
    end;
end;

procedure imprime;
begin
    writeln('matricula e nome');
    for i := 1 to nmax do
        begin
            writeln(vetfichas[i].nmat);
            writeln(vetfichas[i].nome);
        end; {end do for}
    end; {end da procedure}

Begin { programa principal}
    inclui;
    imprime;
    delay(1000);
end. {end do programa principal}
```



23.4 Prática:

- Digite e execute o programa acima.
- Faça um programa, que utilize um vetor de registros(máximo de 6 proprietários), que mostre o número do chassi do carro, o nome do

proprietário e a data de pagamento do IPVA. Ao final imprima na ordem inversa ao que foi digitado.

24 Registros.

24.1 Objetivo: Apresentar o conceito de registros e comandos para operações com arquivos textos.

24.2 Recursos da Linguagem: assign; rewrite; close.

24.3 Exemplo:

(prg24a)

```
program arquivo_texto;
uses crt;
var
    nome      : string[30];
    nota      : real;
    meuarquivo : text;
    grava     : boolean;
    op        : string;

begin
    assign(meuarquivo, 'meuarq.dat')
    { observe que meuarquivo é o nome lógico do arquivo}
    {meuarquivo.dat é o nome físico do arquivo}
    rewrite(meuarquivo)
    { este comando cria o arquivo para gravação pela primeira vez}
    grava := true;
    while grava do
    begin
        writeln('digite o nome do aluno ');
        readln(nome);
        writeln('digite a nota do aluno ');
        readln(nota);
        writeln(meuarquivo, nome, nota(4:2));
        writeln('deseja gravar mais registros? S/N ');
        readln(op);
        if op = 'N' or op = 'n' then grava := false;
    end; {while}
    close(meuarquivo); {fechabdo o arquivo}
end.
```



24.4 Prática:

- Elabore um programa para gravar os dados de uma agenda: nome, endereço, telefone, esporte preferido.
- Elabore um programa para gravar os dados de uma agenda cujo nome seja maior que C e menor que M.

25 Arquivos texto.

25.1 Objetivo: ler arquivos do tipo texto, mostrando na tela ou na impressora os resultados pesquisados, tendo como base o programa da aula 24.

25.2 Recursos da Linguagem: reset; assign; close; readln; eof()

.

25.3 Exemplo:

(prg25a)

```
program ler_arquivo_texto;
uses crt;
var
    nome          : string[30];
    nota          : real;
    meuarquivo   : text;
begin
    assign(meuarquivo, 'meuarq.dat');
    reset(meuarquivo) ;
    { este comando abre o arquivo e no primeiro registro.}
    writeln('este é o arquivo');
    readln(meuarquivo, nome, nota); {primeira leitura}
    while not eof(meuarquivo) do
    begin
        writeln(nome);
        writeln(nota);
        readln(meuarquivo, nome, nota); {leitura de reposição}
    end; {while}
    close(meuarquivo); {fechabdo o arquivo}
end.
```



(prg25b)

```
program adiciona_dados_em_arquivo_texto;
uses crt;
const nmax = 10;
var
    nome          : string[30];
    nota          : real;
    meuarquivo   : text;
    i             : integer;
begin
    assign(meuarquivo, 'meuarq.dat');
    reset(meuarquivo);
    for i := 1 to nmax do
    begin
        append(meuarquivo); {posiciona na última posição do
arquivo}
```

```
writeln('digite o nome');
readln(nome);
writeln('digite a nota');
readln(nota);
writeln(meuarquivo, nome,nota);

end; {for}
close(meuarquivo); {sempre que houver um assign haverá um close}
end.
```



25.4Prática:

- Elabore um menu de opções que permita as seguintes opções:
 - ❑ Grava pela primeira vez dados de um poço de petróleo (número do poço, vazão, em barris por hora, e a profundidade do poço).
 - ❑ Insere mais 15 novos poços.
 - ❑ Ler os dados dos arquivos e imprime na tela.
- Pesquise a função IORESULT(nome do arquivo), e verifique como o programa acima pode ser melhorado.

26 Arquivo Direto.

26.1Objetivo: apresentar o conceito de arquivo direto (file of...).

26.2Recursos da Linguagem: assign, reset, rewrite, file of e close.

26.3Exemplo:

(prg26a)

```
program gravar_arquivo_indexado;
use crt;
const nmax = 10;
type registro = record
    nome : string[10];
    nota : real;
end;
var
    reg : registro;
    i : integer;
    meuarquivo_index : file of reg

Begin
    Assign(meuarquivo_index, 'meuarqid.dat');
    Rewrite(meuarquivo_index);
    For i:= 1 to nmax do
        Begin
            Writeln('digite o nome');
            Readln(reg.nome);
            Writeln('digite a nota');
            Readln(reg.nota);
            Write(meuarquivo_index, reg); {note que o comando write
foi usado em arquivo indexado}
        End;
        Close(meuarquivo_index);
    End.
```



Atenção: o arquivo do tipo texto é mais lento, no que diz respeito ao acesso, do que o arquivo indexado. Exemplo, para ler o enésimo registro em um arquivo texto, é necessário percorrer todos os registros anteriores.

26.4Prática:

- Faça um programa que crie um arquivo indexado que implemente uma agenda com nome, endereço, telefone e e-mail.

27 Arquivo indexado.

27.1 Objetivo: inserir dados em um arquivo do tipo FILE OF

27.2 Recursos da Linguagem: assign, reset, write, seek, filesize, close.

27.3 Exemplo:

(prg27a)

```
program insere_em_arquivo_direto;
{ este programa insere mais dados em um arquivo já existente}
uses crt;
const nmax = 10;
type
    registro_direto = record
        nome      : string[10];
        nota      : real;
    end;
var
    reg : registro_direto;
    meuarquivo_index : file of registro_direto;
    i      : integer;

Begin {programa principal}
Assign(meuarquivo_index, 'meuarqid.dat');
Reset(meuarquivo_index);
For i := 1 to nmax do
Begin
    Writeln('qual o nome do aluno');
    Readln(reg.nome);
    Writeln('qual a nota do aluno');
    Readln(reg.nota);
    Seek(meuarquivo_index, filesize(meuarquivo_index))
    {procura o final do arquivo para inserir mais dados}
    Write(meuarquivo_index, reg);
    { note que para gravar o arquivo direto deve-se usar
      apenas o comando write; não é necessário especificar
      os atributos individuais do registro}
end; {end do for}
close(meuarquivo_index); {não esqueça de fechar seu arquivo}
end.
```

(prg27b)

```
program ler_dados_em_arquivo_direto;
{ este programa lê dados de um arquivo já existente}
uses crt;
type
    registro_direto = record
        nome : string[10];
        nota : real;
    end;
var
    reg : registro_direto;
    meuarquivo_index : file of registro_direto;

Begin {programa principal}
    Assign(meuarquivo_index, 'meuarqid.dat');
    Reset(meuarquivo_index);
    While not(eof(meuarquivo_index)) do
        { eof é a função que identifica o fim do arquivo}
        Begin
            Read(meuarquivo_index, reg);
            Writeln(reg.nome);
            writeln(reg.nota);
            delay(3000);
            { somente se pode ler do arquivo file of com o comando
              read}
        end; {end do for}
    close(meuarquivo_index); {não esqueça de fechar seu arquivo}
end.
```



27.4Prática:

- Faça um programa que utilize procedimentos de forma a criar um arquivo, inserir mais dados, exibir um arquivo gravado na tela e sair do programa. Escolha o tema.

Parte II

Aula Um - Iniciação

```
1.
program meuprimeiroprograma
begin
    writeln("bom dia mundo");
end.

2.
program meuprimeiroprogramamodificado;
begin
    writeln;
    writeln("bom dia mundo");
    writeln;
end.

3.
program meuprimeiroprogramamodificado2;
begin
    write("b");
    write("o");
    write("m");
    writeln(" dia mundo");
end.
```

Aula Dois - Literais

```
1.
program literais
begin
  writeln ("alo" + "mundo");
end.
```

```
2.
program tipoliteral2;
var
  a:   char;
  b:   string;
begin
  a:= "a";
  b:= "lo mundo";
  b:= a + b;
  writeln (b);
end.
```

```
3.
program tiposliterais3;
var
  a   : string;
  b   : string;
  temp :string;
begin
  a:= "joão";
  b:= "maria";
  temp:= b;
  b:= a;
  a:=temp;
end.
```

```
4.
program literais5;
var
  a   : string;
begin
  a:= "joão";
  writeln ("bom dia", a);
end.
```

Aula Três – Tipo Inteiro

1.

```
program tipoInteiro;
var
    lado : integer;
    area : integer;
begin
    lado := 5;
    area := lado * lado;
    writeln (area);
end.
```

2.

```
Program Delta;
var
    a : integer;
    b : integer;
    c : integer;
    meuDelta : integer;
begin
    a:=5; b:=6; c:=7;
    meuDelta := b**2 - 4*a*c;
    writeln (meuDelta);
end.
```

3.

```
program média;
var
    p1 : integer;
    p2 : integer;
    média : integer;
begin
    p1:= 7;
    p2:=9;
    media:= (p1 + p2)/2;
end.
```

4.

```
program funcaoLinear;
var
    x : integer;
    fx : integer;
    coef : integer;
begin
    x:= 5;
```

```
    coef:=6;  
    fx:= coef + 7*x;  
end.
```

5.

```
program funcaoLinear;  
Const  
    coef = 6;  
var  
    x      :integer  
    fx     :integer;  
begin  
    x:= 5;  
    coef:=6;  
    fx:= coef + 7*x;  
end.
```

Aula Quatro – Tipo Real

1.

```
program medias;
var
    p1          : real;
    p2          : real;
    minhamedia  : real;

Begin
    p1:= 6.2;
    p2:= 9.8;
    minhamedia := (p1 + p2)/2;
    writeln(minhamedia);
end.
```

2.

```
Program areacirculo;
const
    pi = 3.14;

var
    r          : real;
    area       : real;

begin
    r:= 5.2;
    area:= pi*r*r;
    writeln(area);
end.
```

3.

```
program margemlucro;
var
    pv          : real;
    pc          : real;
    margem      : real;

begin
    pv := 100;
    pc := 80;
    margem := (pv - pc)/pc * 100;
end.
```

4.

```
program probabilidades;
const
    numero_de_cartas = 52;
var
    ocorrencia    : integer;
    probabilidade  : real;
```

```

begin
  { vermelha }
  ocorrencia := 26;
  probabilidade := ocorrencia / numero_de_cartas * 100;
  writeln ("a probabilidade de carta vermelha e", probabilidade,
"%");

  {preta}
  probabilidade := 1 - probabilidade;
  writeln ("a probabilidade de carta preta e", probabilidade, "%");

  { rei vermelho }
  ocorrencia := 2;
  probabilidade := ocorrencia / numero_de_cartas * 100;
  writeln ("a probabilidade de carta preta e", probabilidade, "%");

  {dama de copas}
  ocorrencia := 1;
  probabilidade := ocorrencia / numero_de_cartas * 100;
  writeln ("a probabilidade de carta preta e", probabilidade, "%");

end.

```

Aula Cinco – Tipo Booleano

1.

```
program meuBoolean;  
  
var  
  
    a      :boolean;  
    b      :boolean;  
  
Begin  
    a:= true;  
    b:= false;  
    writeln("o valor de a e", a);  
    writeln("o valor de b e", b);  
end.
```

2.

```
program meuBooleano2;  
Var  
    a      : booeelan;  
    b      : boolean;  
    c      : booeelan;  
  
begin  
    a:= true;  
    b:= true;  
    c:= a and b;  
    writeln ("o valor de c e", c);  
end.
```

3.

```
program meuBooleano3;  
Var  
    a      : booeelan;  
    b      : boolean;  
    c      : booeelan;  
  
begin  
    a:= true;  
    b:= true;  
    c:= a or b;  
    writeln ("o valor de c e", c);  
end.
```

4.

```
program meuBooleano4;  
Var  
    a      : booeelan;  
    b      : boolean;  
    c      : booeelan;  
  
begin  
    a:= true;
```

```
b:= true;  
c:= not (a or b);  
writeln ("o valor de c e", c);  
end.
```

Aula Seis – If Then Else

1.

```
program amaiornota;
var
    p1,p2,maior :real;
begin
    writeln ("entre com a primeira nota");
    readln(p1);
    writeln ("entre com a segunda nota");
    readln (p2);
    maior := p2;
    if p1 > p2 then maior := p1;
    writeln (" a amior nota é", maior);
end.
```

2.

```
Program introducaoseentaaoo;
const
    media_curso = 6;
var
    p1, p2, media      : real;
begin
    writeln("entre com a nota da primeira prova");
    readln(p1);
    writeln("entre com a nota da segunda prova");
    readln(p2);
    media:= (p1 + p2)/2;
    if media >= media_curso then writeln ("parabens vc passou");
    writeln("sua media final foi", media);
end.
```

3.

```
Program introducaoseentaaoo2;
const
    media_curso = 6;
var
    p1, p2, media      : real;
begin
    writeln("entre com a nota da primeira prova");
    readln(p1);
    writeln("entre com a nota da segunda prova");
    readln(p2);
    media:= (p1 + p2)/2;
    if media >= media_curso then begin
        writeln ("parabens vc passou");
        writeln ("faça matricula em 2009.1");
    end;
    writeln("sua media final foi", media);
end.
```

4.

```
Program introducaoseentaaoo3;
const
```

```

media_curso = 6;
var
  p1, p2, media      : real;
begin
  writeln("entre com a nota da primeira prova");
  readln(p1);
  writeln("entre com a nota da segunda prova");
  readln(p2);
  media:= (p1 + p2)/2;
  if media >= media_curso then begin
    writeln ("parabens vc passou");
    writeln ("faça matricula em 2009.1");
    writeln("sua media final foi", media);
  end
  else begin
    writeln("aproveitamento insuficiente");
    writeln ("refaça matricula na disciplpina em 2009.1");
    writeln("sua media final foi", media);
  end;
end.

```

Aula Sete - Tipo Record

1.

```
program aluno;
type
  tipoPessoa=record
    nome: string;
    idade: integer;
  end;

var  aluno : tipoPessoa;
     media,soma : real;

Begin
  readln(aluno.nome);
  readln(aluno.idade);
  writeln(aluno.nome);
  writeln(aluno.idade);
end.
```

Exercícios

1.

Considere o programa abaixo:

```
1.
2. program equation;
3. {Resolve equacao de 2o grau}
4. var
5.   A,B,C,Det,X1,X2 : integer;
6. begin
7.   write('Digite o valor de A: '); readln(A);
8.   write('Digite o valor de B: '); readln(B);
9.   write('Digite o valor de C: '); readln(C);
10.  Det := sqr(B)-4*A*C;
11.  if A = 0
12.  then
13.    begin
14.      writeln('Essa nao e uma equacao do 2o0 grau.');
```

```
15.      writeln('Raiz da equacao: X =',-C/B:8:2,' ');
16.    end
17.  else
18.    begin
19.      if Det > 0
20.      then
21.        begin
22.          X1 := (-B+sqr(Det))/2*A;
23.          X2 := (-B-sqr(Det))/2*A;
24.          writeln('As raizes da equacao sao: X1 = ',X1:8:2,' e X2 =
',X2:8:2,' ');
25.        end;
26.      if Det = 0 then writeln('As raizes da equacao sao: X1 = X2 = ',
(-B)/2*A:8:2,' ');
27.      if Det < 0 then writeln('A equacao nao possui raizes reais.');
```

```
28.    end;
29.  readln;
30.end.
```

Fonte: <http://www.inf.ufrgs.br/aulas/JEduc/exercicios/equation.html>

Pede-se:

- verifique se os tipos estão de acordo: Verifique o valor retornado da expressão em **u** e **v**.
- O que acontece se a variável **A** for igual a zero?
- Em que situação os comandos de **t** a **x** serão executados?
- Indique os comandos executados quando os valores de **A,B** e **C** forem iguais a -4, 0 e 0 respectivamente.
- Indique os comandos executados quando os valores de **A,B** e **C** forem iguais a -4, 4 e 0 respectivamente.
- Indique os comandos executados quando os valores de **A,B** e **C** forem iguais a -4, 4 e 5 respectivamente.

2.

Considere o programa abaixo:

```
1. Program Circunferencia;
2. {A partir do raio, calcula area ou circunferencia}
3. var
4.   oper: string;
5.   raio, area, perimetro: char;
6. begin
7.   writeln('Calculos referentes a uma circunferencia:');
8.   write('(1 - calcula area; 2 - calcula perimetro): ');
9.   readln(oper);
10.  write('Raio: ');
11.  readln(Raio);
12.  if oper = 1 then
13.    begin
14.      area := pi*sqr(raio);
15.      writeln('A area da circunferencia de raio ',raio,' eh
16.      ',area);
17.    end;
18.  if oper = 2 then
19.    begin
20.      perimetro := 2*pi*raio;
21.      writeln('O perimetro da circunferencia de
22.      raio',raio,' eh perimetro);
23.    end;
24.  if (oper <> 1) and (oper <> 2)
25.  then writeln('Operacao invalida - deve ser 1 ou 2');
26.  readln
27. end.
```

baseado em:

<http://www.inf.ufrgs.br/aulas/JEduc/exercicios/circulo.html>

Pede-se

- a) Explique o que o programa faz.
- b) Qual a importância da variável **oper**.
- c) Quando **oper** for igual a 5 qual será o resultado do programa?
- d) Quando **oper** for igual a 1 e raio igual a 3 qual será o resultado do programa?
- e) Quando **oper** for igual a 2 e o raio igual a 3 qual será o resultado do programa?
- f) Qual o valor retornado pelas em **n** e **s**? Verifique se os tipos estão de acordo.

3.

Considere o programa abaixo:

```
1. program OrdemCrescente;
2. var
3.   n1, n2, n3: integer;
4. begin
5.   write ('Entre com tres numeros: ');
6.   read (n1,n2,n3);
7.   if n3>n1 then
8.     begin
9.       if n1>n2 then write ('A ordem crescente: ',n2,' ',n1,'
10.        ',n3)
11.         else if n2<n3 then write('A ordem crescente:
12.          ',n1,' ',n2,' ',n3);
13.       end;
14.     if n1>n2 then
15.       begin
16.         if n2>n3 then write ('A ordem crescente: ',n3,' ',n2,'
17.          ',n1)
18.         else if n3<n1 then write('A ordem crescente:
19.          ',n2,' ',n3,' ',n1);
20.       end;
21.     if n1<n2 then
22.       begin
23.         if n3<n1 then write ('A ordem crescente: ',n3,' ',n1,'
24.          ',n2)
25.         else if n3<n2 then write('A ordem crescente:
26.          ',n1,' ',n3,' ',n2);
27.       end;
28.     end;
29. end.
```

Baseado em:

<http://www.ime.usp.br/~macmulti/exercicios/inteiros/19Pascal.html>

Pede-se:

- O que o programa faz?
- O que deveria ser feito para ordenar caracteres ou literais?
- Faça um programa que ordene dois números inteiros.
- O que aconteceria se ao invés de se utilizar o sinal de ">" utilizássemos pelo sinal "<"?
- Qual linha que corresponde a saída do programa quando a entrada de dados for 12 8 16?
- Qual linha que corresponde a saída do programa quando a entrada de dados for 12 34 16?

4.

Considere o programa abaixo:

```
1. program triangulos;
2.   var a, b, c : integer;
3. begin read(a, b, c);
4.   if (a<b+c) and (b<a+c) and (c<a+b)
5.       then if (a=b) or (a=c) or (b=c)
6.           then if (a=b) and (b=c) then writeln('Triangulo
   Equilatero')
7.               else writeln('Triangulo Isosceles')
8.           else writeln('Triangulo Escaleno')
9.       else writeln('Nao pode ser um Triangulo');
10.end.
```

Pede-se:

- a) Explique o que o programa faz?
- b) Qual o resultado do programa quando a, b e c possuem o valor 5?
- c) Qual o resultado do programa quando a igual a 7, b igual a 2 e c igual a 4?
- d) Quais as modificações no programa necessárias para melhorar a qualidade da informação de saída, considerando que ao invés de 'Triângulo xxxxxxxxx' mostrar: "O triângulo cujos lados são a, b e c é um triângulo <tipo do triângulo>".

5.

```
1. Program converte;  
2.  
3. var  
4. celsius           : integer;  
5. fahrenheit       : real;  
6.  
7. begin  
    8. writeln ("entre com a temperatura em Farenheit");  
    9. readln(fahrenheit);  
   10. celsius = (fahrenheit - 32)/9 * 5;  
   11. writeln(" a temperatura em Celsius é", Celsius);  
12. End.
```

Pede-se:

- a) Explique o que o programa faz.
- b) programa está correto? Faça as alterações necessárias, em caso de erro.
- c) O que aconteceria se a linha **h** fosse alterada para: `celsius = (fahrenheit - 32)/(9 * 5);`
- d) Modifique o programa acima de forma a se registrar o dia, o mês e o ano da temperatura em celsius e a respectiva temperatura em fahrenheit. Utilize a estrutura record.

6.

Considere o programa abaixo.

```
1. Program IMC ;
2. Var      nome:STRING;
3.   peso,altura,imc:REAL;
4. Begin
5.   writeln('Digite seu nome:');
6.   readln(nome);
7.   writeln('Digite seu peso e sua altura respectivamente:');
8.   readln(peso,altura);
9.   imc:=peso/(altura*altura);
10.  If (imc<=18.5) then Begin
11.      writeln('Você está abaixo do peso:',imc:0:2);
12.      End
13.  Else if (imc>18.5) and (imc<25) then Begin
14.      writeln('Você está no peso normal:',imc:0:2);
15.      End
16.  Else if (imc>=25) and (imc<=30) then Begin
17.      writeln('Você está acima do peso:',imc:0:2);
18.      End
19.  Else writeln('Você está obeso:',imc:0:2);
20.End.
```

Disponível em <http://forum.imasters.uol.com.br/index.php?showtopic=171660>

Pede-se:

1. Explique o que programa faz?
2. Qual o resultado do programa quando a variável IMC = 21, explique passo a passo?
3. O que seria necessário para adicionar a faixa de IMC abaixo de 15 e situação 'você está subnutrido'.
4. O que deveria ser feito para tratar nome, peso, altura e imc de forma conjunta e como esta mudança afetaria o resto do programa.

7.

Considere o programa abaixo.

```
1. Program xpto;
2. var
3.     saldo_medio           :integer;
4.     conceito              :integer;
5. begin
6.     writeln ("entre com o saldo médio");
7.     readln(saldo_medio);
8.     if saldo_medio > 9000 then classificacao := 'cinco estrelas';
9.     if saldo_medio > 8000 then classificacao := 'quatro estrelas';
10.    if saldo_medio > 7000 then classificacao := 'três estrelas';
11.    if saldo_medio > 5500 then classificacao := 'duas estrelas';
12.    if saldo_medio < 5500 then classificacao := 'uma estrela';
13.    writeln('para o saldo médio', saldo_medio, ' a classificação é',
             classificacao);
14. End.
```

Pede-se:

- a) O que o programa faz?
- b) Para o saldo médio igual a 8000 qual a saída do programa.
- c) Ele está correto? Justifique apontando, se for o caso, as correções pertinentes.
- d) Reescreva o programa considerando a nova entrada de dados "total de anos como cliente" e para o esse tempo igual ou superior a 5 e classificação duas estrelas escrever na saída "isento de tarifas"
- e) De que forma seria possível alterar o programa tratando de forma conjunta o nome do cliente, sua conta, o saldo médio, sua classificação e se é isento de tarifa e qual o impacto desta alteração no programa desenvolvido no item 4.

8.

Considere o seguinte programa.

```
1. Program EstimativaPontosFuncao;
2. var
3.     arquivoLogicoInterno, arquivoLogicoExterno, entradaExterna,
   consulta, saidasExternas: integer;
4.     totalGeral, totalArquivoLogicoInterno,
   totalArquivoLogicoExterno, totalEntradaExterna, totalConsulta,
5.     TotalSaidaExterna: integer;
6.
7. Begin
8.     writeln (' entre com o total estimado dos arquivos lógicos
   internos');
9.     readln (arquivoLogicoInterno);
10.    writeln (' entre com o total estimado dos arquivos lógicos
   externos');
11.    readln (arquivoLogicoExterno);
12.    writeln (' entre com o total estimado de entradas externas');
13.    readln (entradaExterna);
14.    writeln (' entre com o total estimado de consultas');
15.    readln (consultas);
16.    writeln (' entre com o total estimado de saidas externas');
17.    readln (saidasExternas);
18.    totalArquivoLogicoInterno := arquivoLogicoInterno * 10;
19.    totalArquivoLogicoExterno := arquivoLogicoExterno * 7;
20.    totalEntradaExterna := entradaExterna * 4;
21.    totalConsultas := consultas * 4;
22.    totalSaidasExternas := saidasExternas * 5;
23. totalgeral := totalArquivoLogicoInterno + totalArquivoLogicoExterno
   + totalEntradasExternas + totalConsultas + totalSaidasExternas;
24. End.
```

Pede-se:

- a) O que o programa faz.
- b) O que deve ser feito para garantir que os valores de `arquivoLogicoInterno`, `arquivoLogicoExterno`, `entradaExterna`, `consultas` e `saidasExternas` sejam sempre positivos.
- c) Para os valores de `totalGeral` temos: a. menor que 300 PF - sistema pequeno. b. entre 300 e 800 PF sistema médio. c. acima de 800PF sistema grande. Escreva o código para produzir a classificação especificada.
- d) Adicione o código ao programa que mostre com detalhes os valores lidos, os totais parciais e o total geral do cálculo por pontos por função.
- e) O que deve ser feito para tratar as variáveis do problema de forma conjunto e como esta solução impacta no resto do programa.

Aula Oito – Comando For.

1.

```
program conta;

var
    meu_contador      : integer;

begin
    meu_contador := 1;
    meu_contador:= meu_contador + 1;
    meu_contador:= meu_contador + 1;
    meu_contador:= meu_contador + 1;
    meu_contador:= meu_contador + 1;
    meu_contador:= meu_contador + 1;
    writeln(meu_contador);
end;
```

2.

```
program conta2;
var
    max_contador, meu_contador :integer;
begin
    max_contador := 10;
    for meu_contador:=1 to max_contador writeln(meu_contador);
end.
```

3.

```
program somador;
var
    max_contador, somador, meu_contador: integer
begin
    max_contador:= 10;
    somador := 10;
    for meu_contador := 1 to max_contador do begin
        somador := somador + 10;
    end;
end.
```

4.

```
program somador;
var
    max_contador, somador, valor, meu_contador: integer
begin
    max_contador:= 10;
    for meu_contador := 1 to max_contador do begin
        readln(valor)
        somador := somador + valor;
    end;
end.
```

Aula Nove – Comando For.

1.

```
program preenchedoTela;
var
  i, j :integer;
Begin
  for i := 1 to 25 do for j := 1 to 80 do writeln ('*');
end.
```

2.

```
program preenchedotela2;
var
  iniciolinha, fimlinha, iniciocoluna, fimcoluna, i, j :integer;
begin
  writeln('entre com a linha inicial');
  readln(iniciolinha);
  writeln('entre com a linha final');
  readln(linhafinal);
  writeln('entra com a coluna inicial');
  readln(colunainicial);
  writeln('entre com a coluna final');
  for i :=linhainicial to linhafinal do begin
    for j := colunainicial to colunafinal do write('*');
    writeln;
  end;
end.
```

3.

```
program FATORIAL;

procedure FATOR(N : integer);
var
  I, FAT : integer;

var
  LIMITE : integer;
begin
  clrscr;
  writeln('Programa Fatorial');
writeln;
write('Informe um valor inteiro: ');
readln(LIMITE);
begin
  FAT := 1;
  for 1 to LIMITE do
    FAT := FAT * I;
  writeln('A fatorial de ', N, ' equivale a: ', FAT);
end;
```

Aula Dez – Vetores e Matrizes.

1.

```
program meuPrimeiroVetor
const
    maxArrayNota = 20

type
    meuArrayNota = array[1..maxArrayNota] of real;

var
    notasTurma : meuArrayNota;
    i : integer;

Begin
    for i:= 1 to maxArrayNota do begin
        writeln('entre com a nota de um aluno');
        readln(notasTurma[i]);
    end;

    writeln(' Listagem das notas da turma ');
    for i:= 1 to maxArrayNota do begin
        writeln(notasTurma[i]);
    end;

end.
```

2.

```
program minhaPrimeiraMatriz;
const
    maxArrayMes = 12;
    maxArrayDia = 30;

type
    meuArrayMatriz = array[1..maxArrayMes, 1..maxArrayDia] of real;

var
    ano : meuArrayMatriz;
    i, j : integer;

Begin
    for i:= 1 to maxArrayMes do begin
        for j:= 1 to maxArrayDia
            writeln('entre com a temperatura do dia');
            readln(ano[i,j]);
        end;

        for i:= 1 to maxArrayMes do begin
            for j:= 1 to maxArrayDia begin
                writeln('entre com a temperatura do dia');
                write(ano[i,j]);
                write;
            end;
            writeln;
        end;

end.
```

Exercícios

1.

Considere o programa abaixo.

```
program Fibonacci;
{ o objetivo do programa é o cálculo de uma sequência de Fibonacci}
var
  i, x, m, n : integer;

begin
  writeln ('Entre com o tamanho da sequencia');
  readln (x);
  n := 0;
  m:=1;
  writeln (n);
  writeln (m);

  for j:=3 to x do begin
    m:=m+n;
    n:=m-n;
    writeln(m);
  end;
end.
```

Pede-se:

5. Explique como o programa faz para calcular uma seqüência de Fibonacci?
6. Mostre, através de uma tabela contendo as variáveis **j**, **x**, **m** e **n**, como o programa funciona, registrando passo a passo os seus valores.
7. Faça modificações no programa para guardar os números em um **array/vetor** de no máximo 50 elementos.
8. Que modificações devem ser feitas para guardar, também, o número ordinal da referida seqüência em uma estrutura **record** do mesmo **array/vetor**. Quais os impactos dessa mudança no programa.