

# Treinamento em Linguagem

# Pascal

7ª edição

PROF. FRANCISCO CARLOS MANCIN  
[www.franciscomancin.rg3.net](http://www.franciscomancin.rg3.net)

AMERICANA - 2002

## ***Prefácio***

A linguagem Pascal ocupa posição de destaque entre as diversas linguagens de grande uso pela comunidade de informática. Apreendê-la é uma grande oportunidade que nos é dada.

Foi criada no início da década de 70 pelo Prof. Niklaus Wirth do *Technical University* em Zurique. Foi batizada pelo seu idealizador de PASCAL, em homenagem ao grande matemático Blaise Pascal, inventor de uma das primeiras máquinas lógicas já conhecidas.

O PASCAL somente ganhou popularidade quando foi adotado pela Universidade da Califórnia, San Diego, em 1973. A nível de microcomputadores, começou a ser utilizado a partir de 1983, quando a *softhouse* Borland lançou o TURBO PASCAL para microcomputadores.

Entre as vantagens na utilização da linguagem PASCAL, posso citar:

- é estruturada;
- é fácil de aprender e de usar;
- é rápida;
- dispõe de recursos gráficos;
- oferece suporte à OOP (Programação Orientada a Objetos), entre muitas outras...

Em se tratando de programação, a experiência me mostrou que só se aprende programação quem realmente programa. Então, nesta disciplina vamos cuidar de programar. Quanto mais nos esforçamos, tanto mais nós aprendemos.

Este material apresentado certamente não suprirá toda a necessidade do aluno. Faz-se necessário que o mesmo busque novas informações em outras fontes: livros, guia do usuário e publicações especializadas. O objetivo básico deste é orientá-lo no desenvolvimento de um conhecimento básico da linguagem PASCAL.

Embora estudaremos até Pascal gráfico, outros recursos da linguagem Pascal não serão aqui abordados. Como exemplo, posso citar o controle das portas de comunicação (serial e paralela) que é empregado em automação industrial e a aplicação de arquivos, que, apesar da sua grande importância será trabalhado de outra forma no 2º Ano, dentro da linguagem de programação Delphi devido às facilidades por ele oferecidas. Ele é sem dúvidas o objetivo nosso dentro da programação Pascal. Com ele você estará desenvolvendo aplicativos totalmente gráficos, voltados para o ambiente Windows.

Certo da grande importância da linguagem Pascal na vida do programador, deixo-lhes um conselho:

***Busquem mais conhecimento !***

***Não se contentem com o pouco aqui apresentado !***

---

*Não existem grandes homens. O que existe é que algumas pessoas fazem brilhar perante os outros a dádiva divina que receberam e a grande maioria nem sequer a aceita.*

---

<b>1. AMBIENTE DE PROGRAMAÇÃO</b>	<b>3</b>	<b>10. LAÇOS DE REPETIÇÃO</b>	<b>21</b>
Sistema Operacional	3	O Comando FOR..	21
Compilador	3	Exercício resolvido usando FOR	21
Interpretador	3	O Comando REPEAT.. UNTIL	21
Link-edição	3	Exercício resolvido usando REPEAT.. UNTIL	22
Aplicativos	3	O Comando WHILE	22
Fases de criação de um programa	3	Exercícios resolvidos usando laços de repetição	22
		Exercícios de fixação	24
<b>2. COMPONENTES BÁSICOS DA LINGUAGEM PASCAL</b>	<b>4</b>	<b>11. PROCEDIMENTOS (PROCEDURES)</b>	<b>26</b>
Documentação e comentários em programas	4	Exercícios resolvidos	26
Nome do programa	4	Exercício de fixação	31
Estrutura de um programa em Pascal	4		
Símbolos especiais	5	<b>12. TIPOS ESTRUTURADOS (ARRAY)</b>	<b>33</b>
Bibliotecas externas (também chamadas de UNITS)	5	Exercícios resolvidos	33
Declaração de variáveis	5	Exercícios de fixação	37
Programa principal (BEGIN ..... END.)	5		
<b>3. VARIÁVEIS DE TIPO EXISTENTES NA LINGUAGEM PASCAL</b>	<b>6</b>	<b>13. ORDENAÇÃO DE DADOS</b>	<b>38</b>
Os grupos de variáveis	6	Exercício resolvido	38
Tipos de variáveis disponíveis no Turbo	6	Exercícios de fixação	39
Declarando variáveis globais	6		
Declarando variáveis locais	6	<b>14. FUNÇÕES (FUNCTIONS)</b>	<b>40</b>
<b>4. ESCRREVENDO PROGRAMAS SIMPLES</b>	<b>7</b>	Exercícios resolvidos	40
Composição do monitor de vídeo	7	Exercícios de fixação	42
Comandos simples de entrada / saída (IN/OUT)	7		
IN/OUT (Entrada / Saída)	7	<b>15. CRIANDO SUA PRÓPRIA UNIT</b>	<b>43</b>
Exercícios resolvidos	7	Estrutura de uma unit	43
Exercícios de fixação	8	Fazes na criação de uma unidade	43
		Ajustando o diretório de busca da sua unidade	43
<b>5. INCORPORANDO UNITS</b>	<b>9</b>	Exercício Resolvido	44
O porquê e a importância de uma UNIT	9	Utilizando a Unit personalizada My_Unit	46
Novos comandos: disponíveis para UNIT CRT	9	Exercícios de fixação	47
Escrevendo programas com controle de vídeo	10		
Exercícios resolvidos	10	<b>16. TRABALHANDO COM INTERRUPÇÕES</b>	<b>48</b>
Exercícios de fixação	11	Exercícios Resolvidos	48
<b>6. UNIT CRT: CORES NO VÍDEO E SONS</b>	<b>12</b>		
Controlando a Cor do Texto e do Fundo do Vídeo	12	<b>17. DEFININDO CONSTANTES E TIPOS DEFINIDOS PELO USUÁRIO</b>	<b>51</b>
Exercícios resolvidos	12	Exercícios resolvidos	51
		Exercícios de fixação	52
<b>7. OPERADORES DO TURBO PASCAL</b>	<b>14</b>		
Atribuindo conteúdo às variáveis	14	<b>18. TRABALHANDO COM O MODO GRÁFICO</b>	<b>53</b>
<b>8. UTILIZANDO DESVIO CONDICIONAL (IF..) E CASE</b>	<b>15</b>	Entendendo a tela no modo gráfico	53
A estrutura do comando IF... THEN... ELSE	15	Funções gráficas disponíveis no Turbo Pascal	53
CASE: Tomando decisões	15	Resumo das funções gráficas mais utilizadas	53
Exercícios resolvidos	15	Exercícios resolvidos	56
Exercícios de fixação	17		
<b>9. ESTRUTURAÇÃO DE DADOS, FUNÇÕES MATEMÁTICAS E OUTROS</b>	<b>18</b>	<b>19. ESTRUTURAS DE RECORD</b>	<b>65</b>
Métodos de Estruturação	18	Exercício resolvido	65
Funções Matemáticas Embutidas	18	Exercício de fixação	67
Outros comandos simples	18		
O Comando Window	19	<b>20. TABELA ASCII (AMERICAN NATIONAL STANDARD CODE FOR INFORMATION INTERCHANGE)</b>	<b>68</b>
Exercícios resolvidos	19		
Exercícios de fixação	20	<b>21. BIBLIOGRAFIA</b>	<b>69</b>

# 1. Ambiente de Programação

## Sistema Operacional

O computador é uma máquina que executa programas. Cada programa pode possuir uma ou mais funções, mas de qualquer forma, transforma os dados de entrada em informações de saída. Mas, para que um computador funcione, precisamos de um Sistema Operacional. O Sistema Operacional é composto por um conjunto de programas cuja função básica é controlar o funcionamento do *hardware*<sup>1</sup> e do *software*<sup>2</sup> em questão. Existem diversos tipos de Sistema Operacional e são classificados em função do trabalho que desenvolvem.

## Compilador

É um programa que converte cada instrução do Programa Fonte<sup>3</sup> em instruções em linguagem de máquina, gerando um programa objeto (.OBJ).

## Interpretador

É a fase da tradução, onde o computador, no momento da execução do programa, converte as instruções de alto-nível para linguagem de máquina.

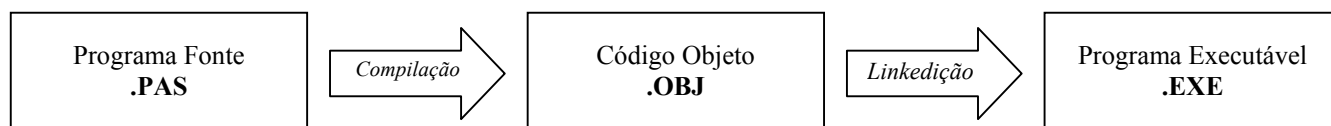
## Link-edição

Processo de ligação de bibliotecas e determinadas rotinas ao programa principal para somente após gerar o código executável (.EXE).

## Aplicativos

São programas que executam determinadas tarefas específicas como por exemplo o processamento de textos, de imagens, de planilhas de cálculos, de banco de dados etc.

## Fases de criação de um programa



---

<sup>1</sup>É o componente físico de um sistema computacional. “Termo da gíria de informática que define o conjunto formado pelas máquinas de processamento de dados ou pelos elementos constitutivos das mesmas, quer sejam do tipo eletrônico, elétricos, magnéticos ou mecânicos”. Ex.: *impressora, drive, teclado, vídeo, joystick etc.*

<sup>2</sup>É o componente lógico de um sistema computacional. “Conjunto de programas, métodos e procedimentos, regras e documentação relacionados com o funcionamento e manejo de um sistema de dados”. Ex.: *DOS, WordStar, Lotus 1-2-3, Word etc.*

<sup>3</sup>Tudo e qualquer programa escrito em linguagem de programação utilizando-se de códigos reconhecíveis pelo ser humano.

## 2. Componentes Básicos da Linguagem Pascal

## Documentação e comentários em programas

Documentar um programa é um hábito que devemos criar desde os primeiros programas. Sua aplicação é vital à manutenção e compreensão futura do programa que hoje escrevemos. Maus programadores não criam o hábito de documentar um programa, julgando uma tarefa tediosa e desnecessária, o que lhes trazem futuros sérios problemas e muitas vezes até mesmo irreversíveis, como o caso de perder totalmente o código fonte, pois sua manutenção muitas vezes é mais lenta do que criarmos um novo programa. Nada disso aconteceria se o programa estivesse todo documentado.

Uma documentação bastante simples que podemos adotar em nossos programas se refere a comentários internos em anexo. Esses comentários são incorporados primeiramente como título inicial de nosso programa e posteriormente em locais onde julgamos necessários, como por exemplo, cálculos específicos, *procedures* e *functions*. *Ex.: documentando um programa com título inicial:*

```
{ Nome do programa..... : MEDIA_1.PAS
  Função..... : Calcula a média aritmética de todos os alunos do 1º Ano
  Ambiente Operacional... : DOS
  Linguagem..... : PASCAL v. 7.0
  Data de criação..... : 10/02/96
  Data da última alteração : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin }
```

Observe que, para incorporarmos comentários em um programa escrito em linguagem Pascal basta escrevermos entre chaves { .....} o texto que desejamos como comentário. Podemos incorporar ao nosso programa comentários simples, composto por uma única linha, ou comentários compostos por mais de uma linha, como é o caso do exemplo anterior.

**N.B.:** Quando você abrir chaves para escrever um comentário nunca se esqueça de fechá-la, porque se não a fecharmos, nosso programa todo tornar-se-á comentário.

**Nome do programa**

O nome do programa não é uma limitação da linguagem ou da sua versão.

Sabemos que todas as operações de controle operacional básico do computador está vinculado ao Sistema Operacional. Assim, também o controle de leitura/gravação de dados está a ele vinculada. Portanto, o nome a ser dado ao programa não é limite da linguagem mas sim do Sistema Operacional e, no nosso caso, o DOS (*Disk Operating System*). Por particularidades próprias, o DOS nos permite criarmos nomes com até 8 caracteres. A extensão do nome do arquivo é obrigatoriamente **.PAS**, por se tratar de um programa escrito em linguagem Pascal.

Ao atribuirmos um nome ao programa fonte devemos tomar o cuidado de não utilizarmos caracteres inválidos, como é o caso de: ?, \* e outros.

## ***Estrutura de um programa em Pascal***

Para escrevermos um programa em Pascal, devemos obedecer a algumas regras básicas: a definição de três áreas distintas: o cabeçalho (*não confunda-o com o título inicial!*), as declarações e as instruções. De forma geral, a primeira e a segunda são opcionais (desde que não se utilize recursos avançados), sendo a única realmente necessária a de instruções, que é feita dentro do bloco principal do programa. Vejamos o exemplo ao lado:

```

{ Nome.....:
  Função.....:
  Autor.....:
  Data.....:
}

```

*cabeçalho do programa*

```

PROGRAM teste;

USES <nomes da unidades a serem incorporadas>;

VAR <variável1>, <variável2>, ..., <variável n>: <tipo da variável>;

BEGIN
    .....
    .....
    comandos...;
    .....
    .....
END.

```

{ início do bloco principal }

{ fim do bloco principal }

## Símbolos especiais

São símbolos comumente utilizados na computação e que na linguagem Pascal possuem significação especial. São eles:

Símbolo	Aplicação
#	Procede um valor inteiro de 0 a 255, representando o caracter correspondente da tabela ASCII.
\$	Procede um valor hexadecimal
'	É usado como delimitador de constantes alfanuméricas
()	Usado para modificar a precedência de operadores e envolver valores nas definições de tipos
,	Separa identificadores em diversas situações. Ex.: <i>A, B, C : REAL;</i>
.	O ponto final indica o fim do bloco principal do programa
..	Indica uma faixa de valores. Ex.: <i>idade: ARRAY[1..25] OF INTEGER;</i>
:	Procede o identificador de tipos nas declarações de variáveis, listas de parâmetros e definições de funções.
:=	Atribuição de valores a variáveis.
;	Delimitador de comandos, declarações e definições de cabeçalhos.
=	Usado como operador de igualdade
[]	Identificador de um elemento de uma matriz, delimita conjuntos.
^	Define um identificador como ponteiro ( <i>pointer</i> ).
{ }	Usado para envolver comentários dentro do programa
@	Indica o endereço de uma variável declarada formalmente

## Bibliotecas externas (também chamadas de UNITS)

São rotinas compiladas separadamente do programa principal. Sua função básica é disponibilizar recursos adicionais da linguagem ao programador, oferecendo dessa forma um maior número de recursos para controle do computador e maior facilidade a nível de programação. A linguagem Pascal possui uma série de bibliotecas externas que podem ser incorporadas e que serão estudadas mais adiante. São elas: SYSTEM, CRT, DOS, GRAPH, PRINTER e OVERLAY. Bibliotecas personalizadas também poderão ser criadas e incorporadas aos programas.

Para a utilização de uma ou mais unidades, é necessário o uso da declaração USES à frente do nome da *unit* desejada, exceção feita apenas à *unit* SYSTEM.

## Declaração de variáveis

Devido à linguagem de programação Pascal ser uma linguagem estruturada, precisamos inicialmente declarar todas as variáveis antes da sua utilização.

Esta declaração deverá ser efetuada logo após o cabeçalho do programa, antes do programa principal. Existem duas classes de variáveis na linguagem Pascal: as variáveis GLOBAIS e as LOCAIS. As variáveis globais são aquelas às quais posso fazer referência em qualquer parte do programa e as variáveis locais só podem ser acessadas em determinadas partes do programa. A declaração de uma variável global é feita logo no início do programa, e a local é feita dentro da *procedure* que a utiliza. Declaração de variáveis locais serão estudadas mais adiante. Ex.: *declarando variáveis globais*:

<b>PROGRAM</b> carros; <b>VAR</b> modelo, fabricante, cor : STRING [25]; ano : INTEGER; valor : REAL;  <b>BEGIN</b> ..... .....  <b>END.</b>	       	       	       
---	------------------------------	------------------------------	------------------------------

declaração de variáveis  
  
programa principal

As definições de variáveis devem ser precedidas da declaração VAR. Cada variável que tiver sua utilização prevista no programa, deve ter um nome de identificação e o seu tipo (conforme visto anteriormente).

O nome de uma variável deve ser único em cada bloco, pode ser de qualquer tamanho, mas apenas os 63 primeiros caracteres é que são significativos. Pode ser composto por letras, números e sublinhado “\_”, porém deve começar sempre com uma letra. Também não pode ser uma palavra reservada. Não pode ser acentuada. É aconselhável que se defina o nome da variável de forma que este lembre a função que ela terá no contexto do programa.

## Programa principal (BEGIN ..... END.)

Corresponde ao “tronco” do programa propriamente dito. A partir dele, podemos encontrar *procedures* e até mesmo *functions*, mas sem ele nada funciona. É o elemento indispensável na confecção de um programa em linguagem Pascal. Ver exemplo no item anterior.

### 3. Variáveis de Tipo Existentes na Linguagem Pascal

#### Os grupos de variáveis

As variáveis na linguagem Pascal estão divididas em quatro grupos, como veremos:

<b>numéricas</b>	que podem ser números inteiros ou reais, formadas pelos dígitos 0 a 9, sinais + ou - e pelo "." para determinar a casa decimal.
<b>alfanuméricas</b>	podem ser formadas por qualquer caracter da tabela ASCII.
<b>lógicas</b>	podem assumir apenas dois valores: TRUE (verdadeiro) ou FALSE (falso).
<b>ponteiros</b>	podem armazenar apenas endereços de memória.

#### Tipos de variáveis disponíveis no Turbo

Classe	Tipo	Valores Comportados	Bytes ocupados na memória	Particularidade
BOOLEAN	Lógico	TRUE ou FALSE	1 Byte	
BYTE	Número inteiro	0 a 255	1 Byte	
CHAR	Caracter	Todos os caracteres da tabela ASCII	1 Byte	
COMP	Número real	-9.2E18 a 9.2E18	8 Bytes	Requer co-processador matemático
DOUBLE	Número real	5.0E-324 a 1.7E308	8 Bytes	Requer co-processador matemático
EXTENDED	Número real	3.4E-4932 a 1.1E4932	10 Bytes	Requer co-processador matemático
INTEGER	Número inteiro	-32768 a 32767	2 Bytes	
LONGINT	Número inteiro	-2147483648 a 2147483647	4 Bytes	
REAL	Número real	2.9E-39 a 1.7E38	6 Bytes	
SHORTINT	Número inteiro	-128 a 127	1 Byte	
SINGLE	Número real	1.5E-45 a 3.4E38	4 Bytes	Requer co-processador matemático
STRING	Caracter	Cadeia de caracteres	Varia de 2 a 256	
WORD	Número inteiro	0 a 65535	2 Bytes	

#### Declarando variáveis globais

A declaração de **variável global** é efetuada logo após a declaração "PROGRAM.....", correspondente ao início do programa. Essas variáveis podem ser utilizadas em todo o programa em questão. São exemplos:

<b>VAR</b>		<b>VAR</b>	
indice	: BYTE;	nome	: ARRAY[1..10] OF STRING[40];
resposta	: CHAR;	idade	: ARRAY[1..10] OF INTEGER;
valorunit	: REAL;	telefone	: ARRAY[1..10] OF STRING[7];
descricao	: STRING[20];	ddd	: ARRAY[1..10] OF STRING[4];
s_n	: BOOLEAN;	fim	: BOOLEAN;
día, mes, ano	: WORD;	cont, tot_id	: INTEGER;
qtde	: INTEGER;	idade_media	: real;

#### Declarando variáveis locais

**Variáveis locais** são todas aquelas pertencentes exclusivamente a uma **PROCEDURE** (uma parte de um programa que possui "vida" própria, com um nome único). Sua declaração é feita dentro da própria **PROCEDURE** e sua utilização ficará restrita unicamente a essa **PROCEDURE**.

Estudaremos as variáveis locais mais adiante, no capítulo destinado às **PROCEDURES**.

## 4. Escrevendo Programas Simples

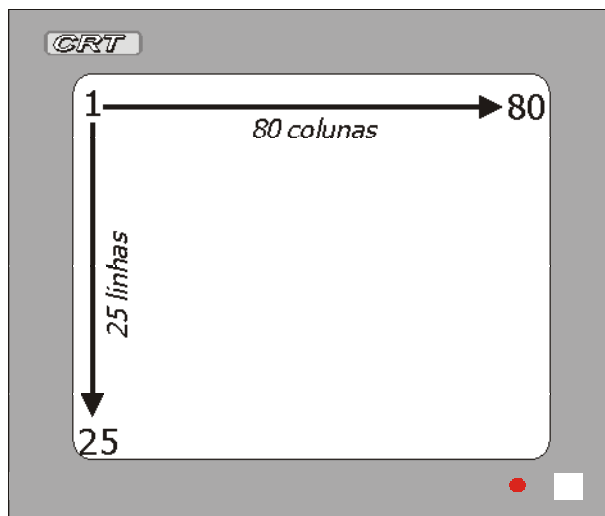
### Composição do monitor de vídeo

O monitor de vídeo ou CRT (*Cathode Ray Tube* - *Tubo de Raios Catódicos*) é sem dúvidas um dos elementos de maior contato tanto com o usuário quanto com o programador.

Operando em modo texto (sem explorar o modo gráfico) o monitor de vídeo pode ser visto como uma matriz de 80 colunas por 25 linhas. Em cada uma dessas posições conseguimos colocar um único caracter da tabela ASCII (*American Code Interchange Information* - *Código Americano Padrão para Intercâmbio de Informações*).

Existe uma linha em especial que não deverá ser utilizada: a linha 25. Essa linha deverá ser reservada ao envio de mensagens ao usuário (se preferir) ou então para a composição de uma moldura em volta da tela.

O posicionamento do cursor na tela é efetuado de forma a indicarmos primeiramente o **número da coluna** seguido do **número de linha**. Isto é, inversamente ao praticado em *Clipper*. O comando utilizado será estudado posteriormente.



### Comandos simples de entrada / saída (IN/OUT)

Através dos comandos de IN/OUT, obtemos uma maior interação com o usuário. A princípio usaremos somente o dispositivo padrão de IN/OUT, ou seja, a CONsole (vídeo + teclado).

Comando	Função	Sintaxe
WRITE	permite escrever no vídeo qualquer conjunto de caracteres.	WRITE ('mensagem', variável); Ex.: WRITE ('Boa Noite !');
WRITELN	permite escrever no vídeo qualquer conjunto de caracteres, <u>avançando para o início da próxima linha</u> .	WRITELN ('mensagem', variável); Ex.: WRITELN ('Estou feliz !');
READ	permite a entrada de dados. <b>Não é recomendado sua utilização para ler dados provenientes do teclado.</b> Sua utilização dá-se normalmente junto a arquivos em disco.	READ (arquivo, campo:char); Ex.: READ (alunos, sexo);
READLN	permite a entrada de dados em variáveis oriundas de diversos dispositivos, assumindo CON como padrão. Para o dispositivo CON, é fixado o limite de 128 caracteres por entrada.	READLN (variável); Ex.: READLN (nome);

### IN/OUT (Entrada / Saída)

Para efetuarmos a entrada de dados num programa, basta atribuímos essa entrada a uma variável, conforme já estudado. Assim, precisaremos inicialmente declarar tal variável. O comando a ser utilizado para entrada de dados (via teclado) é o READLN.

Semelhante ao processo de leitura, o procedimento de escrita de mensagens na tela é bastante simples: basta utilizarmos o comando WRITE ou WRITELN. Como exemplo, analisemos o programa a seguir.

Todos os cálculos que precisarmos efetuar dentro de um programa Pascal deverão estar atribuídos a uma variável. Esta variável deve ser previamente declarada, em função da necessidade da operação.

### Exercícios resolvidos

1. Escreva um programa em linguagem Pascal que leia o nome de qualquer usuário do computador e escreva-o na tela.



```

{ Nome.....: MY_NAME.PAS
  Função....: Solicita que o usuário digite seu nome e recebe saudação
  Data ....: 10/02/96
  Autor....: Francisco Carlos Mancin
}

PROGRAM my_name;
VAR nome: STRING[40];                                {declara a variável}

BEGIN                                                  {inicia programa principal}
  WRITE ('Digite seu nome: ');                        {escreve mensagem na tela}
  READLN (nome);                                     {obtem nome do usuário}
  WRITELN;                                           {pula linha em branco}
  WRITE (nome, ', Muito Prazer !!! Este é o meu primeiro programa em PASCAL');
  WRITELN;                                           {pula linha em branco}
  READLN;                                           {espera pressionar <ENTER>}
END.                                                  {finaliza o programa}

```

### Listagem do programa 1

- Escreva um programa em linguagem Pascal que leia o salário bruto de uma pessoa e calcule seu respectivo salário líquido. Sabe-se que: Salário líquido = Salário Bruto - INSS. Considere o valor percentual de 10% para INSS.

```

{ Nome do programa .....: SAL_LIQ.PAS
  Função.....: Calcula o salário líquido de um funcionário qualquer
  Ambiente Operacional.....: DOS
  Linguagem.....: PASCAL v. 7.0
  Data de criação .....: 10/02/96
  Data da última alteração .....: 2/4/2003
  Escrito por .....: Francisco Carlos Mancin
}

PROGRAM sal_liq;
VAR salariobru, saliq, inss: REAL;                    {declara as variáveis}

BEGIN                                                  {inicia programa principal}
  WRITE ('DIGITE O SALÁRIO BRUTO: ');                {envia mensagem na tela}
  READLN (SALARIOBRU);                               {obtem salário bruto}
  INSS:= SALARIOBRU * 0.1;                            {calcula valor a pagar de INSS}
  SALIQ:= SALARIOBRU - INSS;                          {calcula salário líquido}
  WRITE (' O Salário Líquido É: ', SALIQ:10:2);       {mostra salário líquido na tela}
  READLN;                                           {espera pressionar <ENTER>}
END.                                                  {finaliza programa}

```

### Listagem do programa 2

## Exercícios de fixação

- Escreva um programa que leia a idade de uma pessoa e calcule quantos dias e horas essa pessoa já viveu. Ao final, mostre o resultado no vídeo.
- Escreva um programa em linguagem Pascal que leia dois números reais quaisquer e calcule a soma e o produto entre os mesmos. Os resultados da soma e do produto deverão ser apresentados em vídeo.
- Escreva um programa em linguagem Pascal que leia 3 valores inteiros quaisquer e calcule a média aritmética dos mesmos. A média calculada deverá ser apresentada no vídeo.
- Escreva um programa em linguagem Pascal que calcule a média bimestral de um aluno, sabendo-se que a média é dada por:

$$Média = \frac{Nota_{trabalho} * 4 + Nota_{prova} * 6}{10}$$

## 5. Incorporando UNITS

### O porquê e a importância de uma UNIT

A linguagem Pascal, assim como outras, possui diversas bibliotecas externas que podem ser incorporadas aos nossos programas. Essas bibliotecas são também conhecidas como *units*. Sempre que precisamos adicionar novos recursos (comandos) aos nossos programas, nós recorremos às *units*. Este conceito é relativamente recente na linguagem Pascal, já que surgiu apenas na versão 4 do Turbo. São elas:

<b>SYSTEM</b>	esta é a única unidade que pode ser utilizada sem ser citada. Nela está contida a maior parte das rotinas padrões do Pascal.
<b>CRT</b>	nesta unidade, está contida a maioria das rotinas e variáveis de controle de vídeo e geração de som.
<b>DOS</b>	incorpora as rotinas que envolvem o sistema operacional, permitindo, na maioria das vezes, controles de baixo nível.
<b>GRAPH</b>	possui uma grande quantidade de rotinas gráficas, bastante poderosas. Tem suporte para diversos tipos de vídeo e é bastante rápida.
<b>PRINTER</b>	esta unidade tem apenas a função de definir LST como sendo um arquivo texto, já direcionado para a impressora. Sua operação pode ser feita pelos usuário sem muito esforço.
<b>OVERLAY</b>	permite a geração de unidades para ocuparem um mesmo espaço na memória. Disponível a partir da versão 5.0 do Turbo.

Para incorporarmos uma ou mais unidades num programa, basta que a declaremos logo abaixo da declaração “*PROGRAM xxxxxxxxx;*” em qualquer programa.

### Novos comandos: disponíveis para UNIT CRT

Comando	Função	Sintaxe
CLRSCR	permite limpar a tela e colocar automaticamente o cursor no canto superior esquerdo da mesma. É a contração das palavras CLear e SCReen.	CLRSCR; Ex.: CLRSCR;
DELAY	permite-nos fazer uma pausa programada em qualquer parte do programa, antes da execução da próxima linha. O parâmetro a ser passado é representado em milissegundos.	DELAY (milissegundos: BYTE); Ex.: DELAY (1000);
GOTOXY	permite posicionar o cursor na tela nas coordenadas X e Y, ou seja, coluna e linha. Os valores de coluna e linha deverão ser válidos, caso contrário o comando não será executado.	GOTOXY(coluna,linha); Ex.: GOTOXY (40,17);
CLREOL	permite que se limpe todos os caracteres de uma linha (CLear End Of Line) a partir da posição do cursor sem que sua posição corrente seja alterada e, respeitando ainda, os limites da janela ativa (que será visto posteriormente).	CLREOL; Ex.: CLREOL;
KEYPRESSED	esta função retorna verdadeiro, caso tenha sido pressionada alguma tecla, ou seja, libera a execução do programa somente após ser pressionada alguma tecla.	KEYPRESSED; Ex.: REPEAT UNTIL KEYPRESSED
DELLINE	este procedimento elimina a linha em que estiver posicionado o cursor, efetuando o rolamento das linhas que estiverem abaixo desta e incrementando uma nova linha ao final do vídeo.	DELLINE; Ex.: DELLINE;
INSLINE	permite a inserção de uma linha em branco na posição que se encontra o cursor, efetuando a rolagem automática das demais linhas. A última linha do vídeo é perdida e não poderá mais ser retornada.	INSLINE; Ex.: INSLINE;

Comando	Função	Sintaxe
READKEY	fica lendo o teclado até que o usuário pressione qualquer tecla para prosseguir o processamento do programa.	READKEY; Ex.: READKEY;

## Escrevendo programas com controle de vídeo

Quando precisarmos colocar uma pausa incondicional durante o processamento, basta incluir o comando DELAY onde a desejarmos. O único inconveniente na utilização do comando DELAY é que não conseguimos obter um controle no seu tempo de execução, ou seja, uma vez determinado o intervalo de atraso ele sempre será o mesmo. Para ilustrar uma aplicação deste tipo, analisemos a listagem a seguir:

```
{ Nome do programa ..... : SOMA_1.PAS
  Função ..... : Efetua a soma de 2 números inteiros quaisquer, usando PAUSE
  Ambiente Operacional ..... : DOS
  Linguagem ..... : PASCAL v. 7.0
  Data de criação ..... : 10/02/96
  Data da última alteração .. : 2/4/2003
  Escrito por ..... : Francisco Carlos Mancin  }

PROGRAM soma_1;
USES CRT;                                {incorpora a unidade CRT}
VAR a, b, soma: INTEGER;                  {declara as variáveis}

BEGIN                                   {inicia programa principal}
  CLRSCR;                               {limpa a tela}
  GOTOXY(20,8); WRITE ('Digite o 1º valor: '); {coluna 20, linha 8 e escreve mensagem}
  READLN (a);                            {obtem o valor de a}
  GOTOXY (20,10); WRITE ('Digite o 2º valor: '); {coluna 20, linha 10 e escreve mensagem }
  READLN (b);                            {obtem o valor de b}
  soma:= a + b;                          {calcula a somatória}
  GOTOXY(30,15); WRITE ('Computador pensando....'); {coluna 30, linha 15 e escreve mensagem}
  DELAY (2000);                          {espera ±2 segundos}
  GOTOXY(45,18); WRITE ('O valor calculado é: ', soma); {exibe resultado na tela}
  READKEY;                               {aguarda pressionar qq tecla para
  encerrar}
END.                                     {finaliza o programa}
```

### Listagem do programa 3

## Exercícios resolvidos

- Escreva um programa em linguagem Pascal que leia dois números reais quaisquer e calcule a soma e o produto entre eles, apresentando o resultado em vídeo. **N.B.:** Limpe a tela inicialmente e distribua as mensagens pela tela, posicionando-as onde achar conveniente.

```
{ Nome do programa ..... : SOMA_PRO.PAS
  Função ..... : Calcula a soma e o produto entre dois números reais quaisquer
  Data de criação..... : 10/02/96
  Data da última alteração..... : 2/4/2003
  Escrito por ..... : Francisco Carlos Mancin  }

PROGRAM soma_pro;
USES CRT;
VAR x, y, soma, produto: REAL;

BEGIN
  CLRSCR;
  GOTOXY(15,5); WRITE('Calcula a soma e o produto entre dois números reais');
  GOTOXY(25,10); WRITE ('Digite o primeiro valor: '); READLN (x);
  GOTOXY(25,12); WRITE ('Digite o segundo valor.: '); READLN (y);
  soma:= x + y; produto:= x * y;
  GOTOXY(24,18);
  WRITELN ('O valor da soma é: ', soma:6:2, ' E do produto é: ', produto:6:2);
  READKEY;
END.
```

### Listagem do programa 4

2. Escreva um programa em linguagem Pascal que, após ler a idade de 5 pessoas, calcule a idade média e a apresente em vídeo. **N.B.:** Limpe a tela inicialmente e distribua as mensagens pela tela, posicionando-as onde achar conveniente.

```
{ Nome do programa ..... : IDADE_MD.PAS
  Função ..... : Calcula a idade média de 5 pessoas
  Data da última alteração .. : 2/4/2003
  Escrito por ..... : Francisco Carlos Mancin }

PROGRAM idade_md;
USES CRT;
VAR id1, id2, id3, id4, id5 : INTEGER;
    idademedias : REAL;

BEGIN
  CLRSCR;
  GOTOXY(23,5);   WRITE('Calcula a idade média de 5 pessoas');
  GOTOXY(20,10);  WRITE('Digite a idade da 1ª pessoa: '); READLN(id1);
  GOTOXY(20,12);  WRITE('Digite a idade da 2ª pessoa: '); READLN(id2);
  GOTOXY(20,14);  WRITE('Digite a idade da 3ª pessoa: '); READLN(id3);
  GOTOXY(20,16);  WRITE('Digite a idade da 4ª pessoa: '); READLN(id4);
  GOTOXY(20,18);  WRITE('Digite a idade da 5ª pessoa: '); READLN(id5);
  idademedias:= (id1 + id2 + id3 + id4 + id5)/5;
  GOTOXY(50,22);  WRITELN('A idade média é: ', idademedias:6:3);
  READKEY;
END.
```

Listagem do programa nº 5.

## Exercícios de fixação

- Escreva um programa em linguagem Pascal que leia o preço unitário, a quantidade de um produto qualquer e calcule o valor total do pedido. **N.B.:** Limpe a tela inicialmente e distribua as mensagens pela tela, posicionando-as onde achar conveniente.
- Escreva um programa em linguagem Pascal que calcule o consumo médio de um veículo ao longo de uma viagem. Sabe-se que o consumo é obtido pela seguinte fórmula:

$$\text{Consumo}_{\text{médio}} = \frac{Km_{\text{final}} - Km_{\text{inicial}}}{\text{Litros}_{\text{consumidos}}}$$

Pede-se para exibir o resultado em vídeo. **N.B.:** Limpe a tela inicialmente e distribua as mensagens pela tela, posicionando-as onde achar conveniente.

- Escreva um programa em linguagem Pascal que após ler uma mensagem qualquer, escreva-a devidamente centralizada na tela, na linha que o usuário desejar. (**Dica:** pesquise no Turbo Pascal a função *LENGTH* → digite a função *LENGTH* e, posicionando o cursor sobre ela, pressione as teclas CTRL + F1).
- Escreva um programa em linguagem Pascal que calcule o volume de uma esfera de raio R. **N.B.:** Limpe a tela inicialmente e distribua as mensagens pela tela, posicionando-as onde achar conveniente.

$$\text{Volume} = \frac{4 * \pi}{3} * R^3$$

- Escreva um programa em linguagem Pascal que leia 4 números e some o 1º como 2º, multiplique o 3º com o 4º, divida a multiplicação pela soma e apresente todos os resultados no vídeo: soma, multiplicação e divisão.
- Escreva um programa em linguagem Pascal que calcule a área de um quadrado qualquer. A área calculada deverá ser apresentada em vídeo.
- Escreva um programa em linguagem Pascal que calcule a área de um triângulo retângulo. A área calculada deverá ser apresentada em vídeo.

## 6. Unit CRT: cores no vídeo e sons

### Controlando a Cor do Texto e do Fundo do Vídeo

<b>TEXTCOLOR</b>	este procedimento permite que selecionemos a cor do texto que aparecerá no vídeo, sendo as cores representadas por valores numéricos que variam de 0 a 15, correspondentes a: 0 → preto      4 → vermelho      8 → cinza escuro      12 → vermelho claro 1 → azul      5 → magenta      9 → azul claro      13 → magenta claro 2 → verde      6 → marron      10 → verde claro      14 → amarelo 3 → ciano      7 → cinza claro      11 → ciano claro      15 → branco <i>N.B.:</i> Se somarmos 128 ( <i>Blink</i> ) a qualquer valor, o texto ficará piscante.
<b>TEXTBACKGROUND</b>	este procedimento permite que selecionemos a cor de fundo que aparecerá no vídeo, sendo as cores representadas por valores numéricos que variam de 0 a 7, correspondentes a: 0 → preto      2 → verde      4 → vermelho      6 → marron 1 → azul      3 → ciano      5 → magenta      7 → cinza claro
<b>HIGHVIDEO</b>	este procedimento habilita a exibição da cor do texto para alta intensidade. <i>Ex.: HIGHVIDEO;</i>
<b>LOWVIDEO</b>	este procedimento habilita a exibição da cor do texto em baixa intensidade. <i>Ex.: LOWVIDEO;</i>
<b>NORMVIDEO</b>	este procedimento faz com que a exibição em vídeo retorne ao modo <i>default</i> de intensidade. <i>Ex.: NORMVIDEO;</i>
<b>WINDOW</b>	este procedimento permite que se defina a área útil do vídeo que será utilizada, dentro dos limites <i>default</i> de 80 colunas e 25 linhas. <i>Ex.: WINDOW(10,5,70,20); ou seja, início na coluna 10 da linha 5 e final na coluna 70 da linha 20.</i>
<b>SOUND</b>	emite um sinal sonoro no alto-falante do computador na frequência especificada como parâmetro. <i>Ex.: SOUND(275);</i>
<b>NOSOUND</b>	é utilizado para desligar o som que está sendo emitido pelo comando SOUND.

### Exercícios resolvidos

- Escreva um programa em linguagem Pascal que escreva uma barra colorida centralizada na tela, testando todas as cores disponíveis. Coloque um atraso de 1 segundo na execução do programa antes de apresentar a nova cor. *Sugestão: Utilize código ASCII 219.*

```
{ Nome do programa .....: TEST_COR.PAS
Função .....: Testa todas as cores de texto disponíveis no Turbo Pascal
Data da última alteração ....: 2/4/2003
Escrito por .....: Francisco Carlos Mancin }
```

```
PROGRAM test_cor;
USES CRT;
BEGIN
  CLRSCR;
  GOTOXY(24,2); WRITE('Testando as cores no Turbo Pascal');
  TEXTCOLOR(0); GOTOXY(25,5); WRITE(' 0. Preto'); DELAY(1000);
  TEXTCOLOR(1); GOTOXY(25,6); WRITE(' 1. Azul'); DELAY(1000);
  TEXTCOLOR(2); GOTOXY(25,7); WRITE(' 2. Verde'); DELAY(1000);
  TEXTCOLOR(3); GOTOXY(25,8); WRITE(' 3. Ciano'); DELAY(1000);
  TEXTCOLOR(4); GOTOXY(25,9); WRITE(' 4. Vermelho'); DELAY(1000);
  TEXTCOLOR(5); GOTOXY(25,10); WRITE(' 5. Magenta'); DELAY(1000);
  TEXTCOLOR(6); GOTOXY(25,11); WRITE(' 6. Marron'); DELAY(1000);
  TEXTCOLOR(7); GOTOXY(25,12); WRITE(' 7. Cinza claro'); DELAY(1000);
  TEXTCOLOR(8); GOTOXY(25,13); WRITE(' 8. Cinza escuro'); DELAY(1000);
  TEXTCOLOR(9); GOTOXY(25,14); WRITE(' 9. Azul claro'); DELAY(1000);
  TEXTCOLOR(10); GOTOXY(25,15); WRITE('10. Verde claro'); DELAY(1000);
  TEXTCOLOR(11); GOTOXY(25,16); WRITE('11. Ciano claro'); DELAY(1000);
  TEXTCOLOR(12); GOTOXY(25,17); WRITE('12. Vermelho claro'); DELAY(1000);
  TEXTCOLOR(13); GOTOXY(25,18); WRITE('13. Magenta claro'); DELAY(1000);
  TEXTCOLOR(14); GOTOXY(25,19); WRITE('14. Amarelo'); DELAY(1000);
  TEXTCOLOR(15); GOTOXY(25,20); WRITE('15. Branco'); DELAY(1000);
```

```

TEXTCOLOR(Yellow + Blink);  GOTOXY(25,23);  WRITE('Pressione qualquer tecla para
continuar...');
  READKEY;
END.

```

### Listagem do programa 5

2. Após a execução do programa anterior, substitua todos os comandos *DELAY(1000)* por *SOUND (100)*, incrementando cada SOUND em 100. Ao final do programa, não se esqueça de desligar o som.
3. Escreva um programa em linguagem Pascal que mude a cor do fundo da tela toda vez que o usuário pressionar qualquer tecla, mostrando dessa forma todas as cores de fundo que poderão ser utilizadas em um programa escrito do modo texto.

```

{ Nome do programa .....: COR_FUND.PAS
  Função .....: Testa todas as cores de fundo disponíveis no Turbo Pascal
  Data da última alteração ....: 2/4/2003
  Escrito por .....: Francisco Carlos Mancin  }

PROGRAM cor_fund;
USES CRT;
BEGIN
  TEXTCOLOR(WHITE);      CLRSCR;
  GOTOXY(24,2);  WRITE('Testando as cores no Turbo Pascal');
  GOTOXY(15,5);  WRITE('Pressione qualquer tecla para continuar... ');      READKEY;
  TEXTBACKGROUND(1); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
  TEXTBACKGROUND(2); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
  TEXTBACKGROUND(3); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
  TEXTBACKGROUND(4); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
  TEXTBACKGROUND(5); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
  TEXTBACKGROUND(6); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
  TEXTBACKGROUND(7); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
  TEXTBACKGROUND(0); CLRSCR; GOTOXY(25,24); WRITE('Pressione qq tecla...'); READKEY;
END.

```

### Listagem do programa 6

## 7. Operadores do Turbo PASCAL

### Atribuindo conteúdo às variáveis

A atribuição de valores às variáveis **alfanuméricas** deve ser feita **envolta por apóstrofes**. As variáveis numéricas nunca deverão ter atribuições de valores maiores que as faixas pré-determinadas, conforme visto no capítulo 3.

Os operadores do Turbo Pascal são classificados em 3 grupos: aritméticos, relacionais e booleanos. Além desses 3 grupos, encontramos também: **conjuntos** e **strings**.

Os **aritméticos** normalmente são utilizados em operações entre valores numéricos inteiros ou reais.

+	cálculos de adição entre números reais ou inteiros.
-	cálculos de subtração entre números reais ou inteiros.
*	cálculos de multiplicação entre números reais ou inteiros.
/	cálculos de divisão com resultado real, para operandos <b>reais</b> ou inteiros.
DIV	cálculos de divisão entre números inteiros com <b>resultado também inteiro</b> .
MOD	fornece o resto de uma divisão entre <b>números inteiros</b> .

Os **relacionais** podem ser usados com todos os tipos de variáveis mas, normalmente, são usados em expressões condicionais, retornando sempre um valor lógico TRUE ou FALSE.

=	igual	<>	diferente
>	maior que	<	menor que
>=	maior ou igual a	<=	menor ou igual a

Os **booleanos** executam as operações lógicas da Álgebra de *Boole*.

NOT	inverte o resultado de uma expressão lógica.
AND	somente resulta em verdadeiro se ambos os operandos forem verdadeiros.
OR	é verdadeiro quando pelo menos um dos operandos for verdadeiro.
XOR	<i>exclusive or</i> , só é verdadeiro se apenas um dos operandos for verdadeiro

Os **conjuntos** são potencialmente utilizados quando precisamos representar um grupo de elementos que estão contidos dentro de uma determinada classe. O operando utilizado é o **IN**, o qual retorna verdadeiro, caso o operando pertença ao conjunto especificado. Por exemplo:

*opcao:= tecla IN ['a'..'z', 'A'..'Z'];*

As **strings** permitem a concatenação entre seqüências de caracteres. Por exemplo:

*ling:="PASCAL "*      *recado:="É legal !!!"*      *mensagem:= ling + recado*

## 8. Utilizando Desvio Condicional (IF..) e CASE

### A estrutura do comando IF... THEN... ELSE

Este comando permite alterarmos a sequência lógica de execução em um programa. Será utilizado toda a vez em que o programa precisar tomar alguma decisão, por exemplo:  $idade \geq 18$  ?;  $preço < 250,00$  ?.

Para a utilização do comando IF.. em Pascal, dispomos de uma estrutura única que pode ser representada em quatro modelos diferentes:

<b>IF</b> <condição> <b>THEN</b> <comando>;	<b>IF</b> <condição> <b>THEN</b> <comando> <b>ELSE</b> <comando>;	<b>IF</b> <condição> <b>THEN</b> <b>BEGIN</b> <comandos>; <b>END</b> ;	<b>IF</b> <condição> <b>THEN</b> <b>BEGIN</b> <comandos>; <b>END</b> <b>ELSE</b> <b>BEGIN</b> <comandos>; <b>END</b> ;
---	---	--	--

### CASE: Tomando decisões

Este comando é um seletor de opções, onde apenas a opção que for igual à expressão é que será executada. As opções podem ser do tipo: BYTE, CHAR, INTEGER, LINGINT, SHORTINT, WORD, BOOLEAN ou outros tipos definidos pelo usuário (os quais estudaremos mais adiante). Sua estrutura é:

```

CASE <opção> OF
    'X': BEGIN
            <comandos>;
        END;
    'Y': BEGIN
            <comandos>;
        END;
    'Z': BEGIN
            <comandos>;
        END;
    ELSE <comandos>;
END;
  
```

### Exercícios resolvidos

1. Escreva um programa em linguagem Pascal que após ler duas letras quaisquer informe ao usuário se a primeira é ou não maior que a segunda.

```

{ Nome do programa ..... : COMPARA.PAS
  Função ..... : Compara se a 1ª letra que o usuário digitou é maior que a 2ª
  Ambiente Operacional ..... : DOS
  Linguagem ..... : PASCAL v. 7.0
  Data de criação ..... : 10/02/96
  Data da última alteração . : 2/4/2003
  Escrito por ..... : Francisco Carlos Mancin  }

PROGRAM compara;
USES CRT;
VAR let1, let2: CHAR;

BEGIN
    {inicia programa principal}
    TEXTCOLOR(WHITE); TEXTBACKGROUND(BLUE); CLRSCR;
    GOTOXY(10,08); WRITE('Digite a primeira letra: '); READLN(let1);
    GOTOXY(10,10); WRITE('Digite a segunda letra: '); READLN(let2);
    GOTOXY(20,15);
    IF let1 > let2
        {compara as variáveis}
  
```



```

      THEN
        WRITE('A primeira letra é maior que a segunda !')
      ELSE
        WRITE('A segunda letra é maior que a primeira !');
    READKEY;
END.                                     {finaliza o programa}

```

**Listagem do programa 7**

2. Escreva um programa em linguagem Pascal que após ler dois números reais quaisquer informe ao usuário qual é o menor e qual é o maior.

```

{ Nome do programa .....: ANALISA.PAS
  Função .....: Lê dois números e informa qual é o menor e qual é o maior
  Ambiente Operacional .....: DOS
  Linguagem .....: PASCAL v. 7.0
  Data de criação .....: 10/02/96
  Data da última alteração ...: 2/4/2003
  Escrito por .....: Francisco Carlos Mancin  }

PROGRAM analisa;
USES CRT;
VAR a, b, maior, menor: REAL;

BEGIN
  CLRSCR;
  GOTOXY(12,05); WRITE('Lê dois números e informa qual é o maior e qual é o menor');
  GOTOXY(20,10); WRITE('Informe o 1º número: '); READLN(a);
  GOTOXY(01,10); DELLINE;
  GOTOXY(20,10); WRITE('Informe o 2º número: '); READLN(b);
  GOTOXY(20,10); CLREOL;
  IF a>b
  THEN
    BEGIN
      maior:= a;
      menor:= b;
    END
  ELSE
    BEGIN
      maior:= b;
      menor:= a;
    END;
  GOTOXY(14,15); WRITE('O menor número é:          <20esp.>          O maior número é:');
  GOTOXY(17,17); WRITE(menor:8:3);
  GOTOXY(57,17); WRITE(maior:8:3);
  READKEY;
END.

```

**Listagem do programa 8**

3. Escreva um programa em linguagem Pascal que leia um número inteiro qualquer digitado pelo usuário e informe se o número é par ou ímpar.

```

{ Nome do programa .....: PARIMPAR.PAS
  Função .....: Verifica se o número é par ou ímpar
  Data da última alteração      : 2/4/2003
  Escrito por .....: Francisco Carlos Mancin  }
Program par_impa;
Uses Crt;
Var n : Integer;

Begin
  ClrScr;
  GotoXY (25,3); Write ('Determina se o número é par ou ímpar');
  GotoXY (20,10); Write ('Digite um número inteiro qualquer: '); ReadLn (n);
  GotoXY (40,15); Write (n, ' é ');
  If ODD (n)
  Then Write ('ímpar')
  Else Write ('par');
  Readkey;
End.

```

**Listagem do programa 9**

4. Escreva um programa em linguagem Pascal que leia uma tecla qualquer pressionada no teclado e informe na tela se a tecla pressionada é uma: vogal, número, consoante ou uma exceção (qualquer tecla diferente de vogal ou consoante).

```
{ Nome do programa ..... : TST_CASE.PAS
  Função ..... : Aplicação típica de CASE
  Ambiente Operacional .... : DOS
  Linguagem ..... : PASCAL v. 7.0
  Data de criação ..... : 10/02/96
  Data da última alteração : 2/4/2003
  Escrito por ..... : Francisco Carlos Mancin  }
```

```
PROGRAM tst_case;
USES CRT;
VAR c, opcao: CHAR;

BEGIN
  CLRSCR;
  GOTOXY(25,5); WRITE ('Verifica a tecla pressionada');
  GOTOXY(10,12); WRITE ('Pressione uma tecla: ');
  c:= UPCASE(READKEY);
  GOTOXY(1,12); DELLINE;
  GOTOXY(10,12); WRITE('Você pressionou a tecla : ', c);

  CASE c OF
    '0'..'9': BEGIN
      GOTOXY(30,15);
      WRITE('O Character é Numérico');
    END;
    'A','E','I','O','U': BEGIN
      GOTOXY(30,15);
      WRITE('O Character é uma Vogal');
    END;
    'B'..'Z': BEGIN
      GOTOXY(30,15);
      WRITE('O Character é uma Consoante');
    END;
  ELSE
    BEGIN
      GOTOXY(30,15);
      WRITE('O Character é uma Exceção');
    END;
  END;
  opcao:= UPCASE (READKEY);
END.
```

### Listagem do programa 10

## Exercícios de fixação

1. Escreva um programa em linguagem Pascal que calcule as raízes de uma equação de segundo grau qualquer, apresentando-as em vídeo. Sabe-se que só existirão raízes se o  $\Delta \geq 0$ . Considere:

$$\Delta = b^2 - 4.a.c \qquad x_1 = \frac{-b + \sqrt{\Delta}}{2.a} \qquad x_2 = \frac{-b - \sqrt{\Delta}}{2.a}$$

2. Escreva um programa em linguagem Pascal que leia três números inteiros e coloque-os em ordem crescente, apresentando-os em vídeo.
3. Escreva um programa em linguagem Pascal que apresente na tela um menu com cinco opções quaisquer e que informe ao usuário qual foi a opção por ele escolhida. Incorpore também validação de dados para a leitura da opção do usuário, ou seja, o usuário não poderá escolher uma opção diferente das 5 apresentadas.
4. Escreva um programa em linguagem Pascal, utilizando o comando CASE, que implemente uma calculadora de operações básicas para dois números reais quaisquer. O resultado deverá ser apresentado em vídeo.

## 9. Estruturação de Dados, Funções Matemáticas e Outros

### Métodos de Estruturação

A estruturação de dados define a forma pela qual os dados encontram-se organizados dentro de um programa. Existem quatro métodos de estruturação em Pascal, a saber:

<b>ARRAY</b>	cria uma estrutura (matriz) com um número fixo de elementos de um tipo base (inteiro, real, <i>char</i> , <i>string</i> , etc.).
<b>FILE</b>	cria uma estrutura com um número indeterminado de valores de um tipo base. Equivale à definição de um arquivo seqüencial.
<b>RECORD</b>	estrutura com um número fixo de elementos chamados “campos”, em geral de tipos diferentes.
<b>SET OF</b>	variáveis deste tipo assumem valores que são os conjuntos que podem ser formados com valores de um tipo base. Este tipo será visto posteriormente.

### Funções Matemáticas Embutidas

<b>ABS (X)</b>	retorna o valor absoluto de X. Ex.: <i>ABS(-28)</i> ;
<b>ARCTAN (X)</b>	calcula arcotangente de X. Ex.: <i>ARCTAN(46)</i> ;
<b>COS (X)</b>	calcula cos-seno de X. Ex.: <i>COS(PI)</i> ;
<b>DEC (X)</b>	permite decrementar X (inteiro) em uma ou mais unidades. Ex.: <i>DEC(5)</i> ; <i>DEC(7,2)</i> ;
<b>EXP (X)</b>	função exponencial de $e^x$ . Ex.: <i>EXP(1.0)</i> ;
<b>FRAC (X)</b>	retorna a parte fracionária de X. Ex.: <i>FRAC(-12.7689)</i> ;
<b>INC (X)</b>	permite incrementar X (inteiro) em uma ou mais unidades. Ex.: <i>INC(5)</i> ; <i>INC(7,2)</i> ;
<b>INT (X)</b>	retorna o valor inteiro de X (real). <i>INT(-56.978)</i> ;
<b>LN (X)</b>	calcula logaritmo natural de X. Ex.: <i>LN(EXP(1.0)):5.2</i> ;
<b>ODD (X)</b>	retorna verdadeiro se X ( <i>LONGINT</i> ) for ímpar, caso contrário retorna falso. Ex.: <i>ODD(27)</i> ;
<b>ORD (X)</b>	Retorna o código ASCII de um determinado caracter. Ex.: <i>ORD('C')</i> ;
<b>PI</b>	retorna o valor de pi (3.141592654...). Ex.: <i>PI</i> ;
<b>RANDOM</b>	esta função retorna um número aleatório que pode variar de 0 a 1, ou por um limite passado como parâmetro. Ex.: <i>RANDOM</i> ; <i>RANDOM(100)</i> ;
<b>RANDOMIZE</b>	após sua execução, permite ao <i>RANDOM</i> gerar uma nova seqüência de números aleatórios. Se não utilizarmos tal declaração, o número a ser gerado será sempre o mesmo. Ex.: <i>RANDOMIZE</i> ;
<b>ROUND (X)</b>	sendo X um número real, retorna o inteiro mais próximo de X. Ex.: <i>ROUND(-87.8765)</i> ;
<b>SIN (X)</b>	calcula seno de X. Ex.: <i>SIN(50)</i> ;
<b>SQR (X)</b>	retorna X elevado ao quadrado, ou seja, $X^2$ . Ex.: <i>SQR(3)</i> ;
<b>SQRT (X)</b>	calcula a raiz quadrada de X. Ex.: <i>SQRT(125)</i> ;
<b>TRUNC (X)</b>	sendo X um número real, retorna a parte inteira de X. Ex.: <i>TRUNC(PI)</i> ;

### Outros comandos simples

<b>CHR</b>	esta função retorna o caracter ASCII correspondente ao valor do parâmetro passado.
<b>UPCASE</b>	esta função converte automaticamente todos os <i>strings</i> minúsculos de uma sentença por seu correspondente maiúsculo.
<b>LENGTH</b>	retorna a quantidade de caracteres contidos em uma <i>string</i> .

<b>VAL</b>	este procedimento converte uma <i>string</i> passada como parâmetro para valor numérico. Caso o conteúdo da string não seja numérico, o fato será informado em uma variável de retorno de erro. Se o retorno de erro for $\neq 0$ , implica em um erro de conversão. O número indicado corresponde à posição onde ocorreu o erro. Ex.: <i>VAL(num)</i> ; onde <i>num</i> é uma variável do tipo <i>string</i> .
<b>STR</b>	este procedimento faz a conversão de um valor numérico e seu correspondente em <i>string</i> . Ex.: <i>STR(987.65,s)</i> ;

## O Comando Window

É utilizado para redefinir uma nova janela no modo texto que será responsável pela exibição do programa a partir de agora. Ao acionar uma nova janela, precisamos tomar cuidado com o número de linhas e colunas existentes pois elas dependerão das dimensões da nova janela e não mais das definições padrões, que correspondem ao comando *WINDOW(1,1,80,25)*.

Para criar uma nova janela precisamos informar alguns parâmetros ao comando window e o fazemos da seguinte forma:

***WINDOW (col\_ini, lin\_ini, col\_fin, lin\_fin);***

Se informar algum parâmetro inválido, o comando window não executará nada.

Outro detalhe que devemos prestar atenção é que o window cria uma janela a partir das coordenadas da janela atual, ou seja, se existe uma janela ativa, a criação de uma outra dentro desta estará vinculada às suas coordenadas e não às coordenadas padrão. Isso pode gerar algumas dúvidas ao recriar uma nova janela.

Para retornar à janela padrão (80 colunas, 25 linhas) precisamos executar o comando:

***WINDOW(1,1,80,25);***

## Exercícios resolvidos

- Escreva um programa em linguagem Pascal verifique se um número é positivo, negativo ou nulo. Envie a mensagem de saída em amarelo intenso e utilize fundo em azul.

```
{ Nome do programa..... : POS_NEGA.PAS
Função..... : Verifica se um número inteiro digitado é positivo, negativo ou nulo
Data de criação..... : 10/02/96
Data da última alteração.. : 2/4/2003
Escrito por..... : Francisco Carlos Mancin  }
```

```
PROGRAM pos_neg;
USES CRT;
VAR  n      : INTEGER;
     Tecla  : CHAR;

BEGIN
    HIGHVIDEO;                                {ativa vídeo intenso}
    TEXTBACKGROUND(BLUE);                     {ativa fundo azul}
    CLRSCR;
    GOTOXY(10,3); WRITE('Determina se o número digitado é: Positivo, Negativo ou Nulo');
    GOTOXY(10,4); WRITE('-----');
    GOTOXY(15,10); WRITE('Digite um número inteiro qualquer: '); READLN(n);
    TEXTCOLOR(YELLOW); GOTOXY(50,18);         {ativa texto amarelo}
    IF N>0
    THEN
        WRITE('O número é positivo !')
    ELSE
        IF N<0
        THEN
            WRITE('O número é negativo !')
        ELSE
            WRITE('O número é nulo !');
    TEXTCOLOR(2+128);                          {verde piscante}
    GOTOXY(35,25); WRITE('Pressione qualquer tecla para finalizar...', #7); {emite beep}
    tecla:= READKEY;                            {espera pressionar qualquer tecla}
    NORMVIDEO; CLRSCR;                          {ativa vídeo normal e limpa a tela}
END.
```

### Listagem do programa 11

- Escreva um programa em linguagem Pascal que coloque uma cor de fundo (azul claro) na tela do computador que crie uma janela com o fundo branco, limitando a sua utilização às novas coordenadas.

```

{ Nome do programa .....: TS_WINDO.PAS
  Função .....: Utiliza o comando WINDOW para criar janelas na tela
  Ambiente Operacional .....: DOS
  Linguagem .....: PASCAL v. 7.0
  Data de criação .....: 10/02/96
  Data da última alteração ....: 2/4/2003
  Escrito por .....: Francisco Carlos Mancin    }

PROGRAM ts_window;
USES CRT;

BEGIN
    TEXTBACKGROUND(1);  CLRSCR;                { limpa a tela, com fundo azul }

    WINDOW(6, 3, 78, 5);                { limpa uma caixa em preto }
    TEXTBACKGROUND(0);  CLRSCR;

    WINDOW(4, 2, 76, 4);                { limpa uma caixa em amarelo }
    TEXTBACKGROUND(14);  CLRSCR;

    GOTOXY(26,2);
    WRITE('Este recurso é Bárbaro !');    { escreve texto dentro da janela}

    WINDOW(1, 25, 80, 25);                { cria janela p/ mensagens }
    TEXTBACKGROUND(7);  CLRSCR;

    GOTOXY(25, 1); TEXTCOLOR(0 + 128);
    WRITE('Pressione qualquer tecla p/ encerrar...');
    READKEY;

End.

```

### Listagem do programa 12

## Exercícios de fixação

- Escreva um programa em linguagem Pascal que calcule a tabuada de um número real (com 3 casas decimais) qualquer, fornecido pelo usuário. **N.B.:** Apresente na tela a tabuada devidamente centralizada, de acordo com a formatação tradicional. Utilize ainda os recursos de coloração de tela, a seu gosto.
- Escreva um programa em linguagem Pascal que simule um jogo na Loto, sorteando aleatoriamente 5 dezenas (de 0 até 99).
- Escreva um programa em linguagem Pascal que simule um jogo na Sena, sorteando aleatoriamente 6 dezenas (de 0 até 60).
- Escreva um programa em linguagem Pascal que determine se um número inteiro qualquer, digitado, é par ou ímpar, sem utilizar a função ODD.
- Escreva um programa em linguagem Pascal que desenhe um contorno gráfico em volta da tela, com posição inicial em 1,1 e final em 80,25. *Sugestão: utilize os caracteres gráficos da tabela ASCII.*
- Você foi contratado pelos organizadores de uma competição para escrever um programa em linguagem Pascal que efetue a conversão automática dos resultados em diversas unidades métricas. São elas:
  - Conversão dos resultados de salto em altura: os resultados são relatados em metros e deverão ser convertidos para pés (ft) e polegadas (in). Sabe-se que 1 pé vale 12 polegadas e 1 metro vale 39,37 polegadas;
  - Conversão do tempo da corrida de 100 metros: deverá ser calculado o tempo correspondente para 100 jardas. Considere o atleta correndo a uma velocidade constante e que 1 jarda vale 3 pés ou 0,9144 metro.
- Uma lista de taxa de câmbios de 1977, para troca de moedas estrangeiras, fornece a seguinte tabela de equivalência:
 

100	Francos franceses .....	→	21.55	dólares canadenses
1	Dólar americano .....	→	1.06	dólar canadense
100	Marcos alemães .....	→	43.20	dólares canadenses
1	Libra inglesa .....	→	1.84	dólar canadense
100	Coroas suecas .....	→	24.25	dólares canadenses
100	Dracmas gregos .....	→	2.95	dólares canadenses

 Escrever um programa em linguagem Pascal apresentando um menu com as seguintes opções:
  - Ler a quantidade de francos e imprimir o equivalente em dólares canadenses;
  - Ler a quantidade de dólares americanos e imprimir o equivalente em coroas e francos;
  - Ler a quantidade de dracmas e imprimir o equivalente em libras;
  - Ler a quantidade de dólares canadenses e imprimir o equivalente em dólares americanos e marcos alemães.

## 10. Laços de Repetição

Em programação estruturada, os laços de repetição são muito importantes. Praticamente, controlam toda a estrutura. Em Pascal encontramos três tipos de laços de repetição: FOR, REPEAT..UNTIL e WHILE.

### O Comando FOR..

Este comando permite que executemos o bloco de programa nele contido um determinado número de vezes, predeterminado. Sua sintaxe é:

```

FOR <variável> := <valor inicial> TO <valor final> DO
  BEGIN
    <comandos>;
  END;

ou

FOR <variável> := <valor final> DOWNTO <valor inicial> DO
  BEGIN
    <comandos>;
  END;

```

No primeiro exemplo, o valor da *variável* será incrementado automaticamente em 1, até que o *valor final* seja atingido. Se usarmos o **DOWNTO** no caso de **TO** (conforme o exemplo 2), o valor da *variável* será decrementado automaticamente até atingir o *valor inicial*.

### Exercício resolvido usando FOR

1. Desenhe uma moldura de asteriscos em volta do monitor de vídeo, utilizando o comando FOR, com incremento positivo e negativo. Aplique também o comando de atraso para que possamos ver a “montagem” da moldura.

```

{ Nome do programa .....: ASTERISC.PAS
  Função .....: Desenha uma moldura de asteriscos em volta da tela
  Ambiente Operacional ...: DOS
  Linguagem .....: PASCAL v. 7.0
  Data de criação .....: 10/02/96
  Data da última alteração: 2/4/2003
  Escrito por .....: Francisco Carlos Mancin   }

PROGRAM asterisc;
USES CRT;
VAR i      : BYTE;

BEGIN
  CLRSCR;
  FOR i:= 1 TO 80 DO
    BEGIN
      GOTOXY(i,1);   WRITE('*');
      GOTOXY(i,24);  WRITE('*');
      DELAY(100);
    END;
  FOR i:= 23 DOWNTO 2 DO
    BEGIN
      GOTOXY(1,i);   WRITE('*');
      GOTOXY(80,i);  WRITE('*');
      DELAY(100);
    END;
  GOTOXY(12,18); WRITE('Pressione qualquer tecla para terminar...');
  READKEY;
END.

```

*Listagem do programa 13*

### O Comando REPEAT.. UNTIL

Este comando permite-nos que executemos o bloco de comandos nele contido até que uma determinada condição seja satisfeita. Sua sintaxe é:

```

REPEAT
  <bloco de comandos>;
UNTIL <condição>;

```

**Exercício resolvido usando REPEAT.. UNTIL**

1. Escreva um programa em linguagem Pascal que apresente devidamente centralizado na tela os números de 1 a 10 pausadamente, isto é, de acordo com o avanço desejado do usuário. Utilize REPEAT..UNTIL e KEYPRESSED para efetuar tal pausa.

```
{ Nome do programa..... : T_REPEAT.PAS
  Função..... : Testa uma aplicação típica para REPEAT.. UNTIL
  Ambiente Operacional..... : DOS
  Linguagem..... : PASCAL v. 7.0
  Data de criação..... : 10/02/96
  Data da última alteração: 2/4/2003
  Escrito por..... : Francisco Carlos Mancin   }

PROGRAM t_repeat;
USES CRT;
VAR  i: INTEGER;

BEGIN
  CLRSCR;
  GOTOXY(15,25); WRITE('Pressione qualquer tecla para ver mais úmeros...');
  i:= 1;
  REPEAT
    GOTOXY (39,i+8); WRITE (i);
    i:= i + 1;
    KEYPRESSED;
  UNTIL i > 10;
  KEYPRESSED;
  CLRSCR;
END.
```

*Listagem do programa 14*

**O Comando WHILE**

Dentro desta estrutura o bloco de comandos também só é executada enquanto uma condição estiver sendo satisfeita, ou seja, for verdadeira. Sua sintaxe:

```
WHILE <condição> DO
  BEGIN
    <comandos>;
  END;
```

**Exercícios resolvidos usando laços de repetição**

1. Escreva um programa em linguagem Pascal que apresente devidamente centralizado na tela os números de 1 a 10 pausadamente, isto é, de acordo com o avanço desejado do usuário. Utilize o comando WHILE como elemento mestre no processamento.

```
{ Nome do programa ..... : TS_WHILE.PAS
  Função ..... : Testa uma aplicação típica para WHILE
  Ambiente Operacional .... : DOS
  Linguagem ..... : PASCAL v. 7.0
  Data de criação ..... : 10/02/96
  Data da última alteração : 2/4/2003
  Escrito por ..... : Francisco Carlos Mancin   }

PROGRAM ts_while;
USES CRT;
VAR  i: INTEGER;

BEGIN
  CLRSCR;
  GOTOXY(15,25); WRITE('Pressione qualquer tecla para ver mais números...');
  i:= 1;
  WHILE i <= 10 DO
    BEGIN
      GOTOXY (39,i+8); WRITE (i);   i:= i + 1;  READKEY;
    END;
  CLRSCR;
END.
```

*Listagem do programa 15*

2. Escreva um programa em linguagem Pascal que apresente um número inteiro qualquer na tela, maior que 0 e menor que 1000. O número deverá ser apresentado devidamente centralizado e após a apresentação do mesmo, disponibilize a opção de continuar ou não a execução do programa.

```
{ Nome do programa..... : GERA_NUM.PAS
  Função..... : Gera um número aleatório e possibilita continuar ou não
  Data de criação..... : 10/02/96
  Data da última alteração.. : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin   }

PROGRAM gera_num;
USES CRT;
VAR  num      : INTEGER;
     opcao    : CHAR;

BEGIN
  RANDOMIZE;
  opcao := 'S';
  TEXTCOLOR(WHITE);  CLRSCR;
  GOTOXY(22,3);  WRITE('Gera números aleatórios de 0 e 1000');
  GOTOXY(25,11);  WRITE('O número é: ');
  WHILE opcao = 'S' DO
    BEGIN
      num := TRUNC(RANDOM(1000));
      TEXTCOLOR(YELLOW);
      GOTOXY(38,i+8);  WRITE (num);
      TEXTCOLOR(YELLOW + BLINK);
      GOTOXY(27,23);  WRITE('Deseja continuar ? ( S / N ) ');
      opcao := UPCASE (READKEY);
      TEXTCOLOR (WHITE);
      IF opcao < > 'S' THEN CLRSCR;
      GOTOXY (25,23);  CLREOL;
    END;
  END.
```

#### Listagem do programa 16

3. Escreva um programa em linguagem Pascal que emita um som contínuo de frequências contidas na faixa de 40 a 800Hz. A duração de cada frequência deverá ser controlada pelo usuário.

```
{ Nome do programa ..... : GERA_SOM.PAS
  Função..... : Gera um som na frequência de 40 a 800Hz
  Data de criação..... : 10/02/96
  Data da última alteração.. : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin   }

PROGRAM gera_som;
USES CRT;
VAR  i, tempo : INTEGER;

BEGIN
  TEXTCOLOR(LIGHTGREEN);
  CLRSCR;
  GOTOXY(26,3);  WRITE ('Gerando sons no micro...');
  GOTOXY(10,10);  WRITE('Informe o tempo de duração de cada frequência: ');
  READLN(tempo);
  GOTOXY(1,10);  DELLINE;
  FOR i := 40 TO 800 DO                                {aumenta a frequência}
    BEGIN
      SOUND(i);
      DELAY(tempo);
      NOSOUND;
    END;
  FOR i := 800 DOWNT0 40 DO                             {diminui a frequência}
    BEGIN
      SOUND(i);
      DELAY(tempo);
      NOSOUND;
    END;
  END.
```

#### Listagem do programa 17

4. Escreva um programa em linguagem Pascal que reproduza sons correspondentes ao valor numérico da tecla pressionada, até que a tecla ESC não seja pressionada.



```

{ Nome do programa.....: BARULHO.PAS
  Função.....: Produz sons diferentes quando as teclas são pressionadas
  Linguagem.....: PASCAL v. 7.0
  Data de criação.....: 10/02/96
  Data da última alteração..: 2/4/2003
  Escrito por.....: Francisco Carlos Mancin  }

PROGRAM barulho;
USES CRT;

VAR
  tecla : CHAR;
  Fator, Freq, Tempo, Valor: INTEGER;

BEGIN
  TEXTCOLOR (15);
  CLRSCR;
  GOTOXY (15,02); WRITE ('Pressione qualquer tecla e ouça o som correspondente');
  GOTOXY (15,03); WRITE ('-----');
  GOTOXY (23,23); WRITE ('Pressione a tecla <Esc> para encerrar...');
  TEXTCOLOR (YELLOW); GOTOXY (42,23); WRITE ('ESC'); GOTOXY (80,25);
  REPEAT
    tecla := UPCASE (READKEY);
    valor := ORD(tecla);
    freq := (valor - 47) * 80;
    SOUND(freq);
    DELAY(100);
    NOSOUND;
  UNTIL Tecla = #27;           {espera pressionar a tecla ESC}
END.

```

### Listagem do programa 18

5. Escreva um programa em linguagem Pascal que apresente o código ASCII de qualquer tecla pressionada. **Obs.:** O programa deverá ser encerrado ao se pressionar a tecla Esc.

```

{ Nome do programa .....: Key_Code.PAS
  Função .....: Mostra o código ASCII da tecla pressionada
  Data de criação .....: 10/10/2000
  Data da última alteração : 20/2/2001
  Escrito por .....: Francisco Carlos Mancin  }

PROGRAM ts_window;
USES CRT;

BEGIN
  TextBackground(Blue); Textcolor(Yellow); CLRSCR;
  Gotoxy(22,3); Write ('Exibe código ASCII da tecla pressionada');
  GOTOXY (35,18); WRITE ('Pressione a tecla <Esc> para encerrar...');
  TextColor(White); Window (20,7,60,13); TextBackground(Black); Clrscr;
  REPEAT

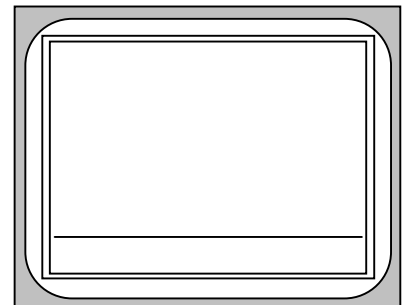
    Gotoxy(13,2); Write ('Tecla      Código ASCII');
    Gotoxy(15,4); tecla:= readkey; Write (tecla, ORD(tecla):13);
  UNTIL Tecla = #27
END.

```

### Listagem do programa 19

### Exercícios de fixação

- Escreva um programa em linguagem Pascal, utilizando FOR, para desenhar uma moldura gráfica em ciano ao redor da tela, segundo apresentado ao lado:
- Escreva um programa em linguagem Pascal que dê palpites para o jogo da Sena, de forma que o programa seja viciado, portanto, nunca deverá apresentar nenhum número no intervalo de 15 a 20, inclusive. Disponibilize a opção de continuar a execução do programa ou não.
- Escreva um programa em linguagem Pascal que desenhe quadrados e retângulos gráficos na tela, em função das coordenadas fornecidas pelo usuário. A cada execução do programa, uma única figura deverá ser gerada. Disponibilize a opção de continuar a execução do programa ou não.
- Escreva um programa em linguagem Pascal que desenhe uma linha de caracter gráfico em qualquer posição da tela. Ao desenhar a linha completamente, o computador deverá emitir um sinal sonoro e perguntar ao usuário se ele



- deseja que mais uma linha seja desenhada. Em função da resposta, a nova linha deverá ser desenhada ou não. Disponibilize a opção de continuar a execução do programa ou não.
5. Você fez um empréstimo de US\$ 2.200,00 para comprar um microcomputador. Considere que, a partir do terceiro mês de empréstimo (inclusive) você começará a pagar prestações correspondentes a 4% do valor do empréstimo. Sabendo-se ainda que será pago juros reais de 2,68% ao mês, pede-se:
    - Escreva um programa em linguagem Pascal que calcule quantas prestações deverão ser pagas até cobrir totalmente o empréstimo;
    - Qual será o valor pago da última parcela ?
    - Considere o valor do empréstimo, os juros e o percentual a ser pago mensalmente como variáveis.
  6. Escreva um programa em linguagem Pascal que efetue a conversão de base decimal para hexadecimal de um número inteiro qualquer contido no intervalo de 0 a 10000. Ao concluir a conversão e apresentá-la em vídeo deverá ser disponibilizada a opção de continuar ou não a execução do programa.
  7. Escreva um programa em linguagem Pascal que desenhe quadrados e retângulos gráficos na tela, em função das coordenadas fornecidas pelo usuário. A cada execução do programa, duas figuras deverão ser geradas, ou seja, o usuário informa as dimensões da primeira e a segunda deverá ser exatamente sua metade. Ao desenhá-las, a menor deverá ser circunscrita na maior. Disponibilize a opção de continuar a execução do programa ou não.
  8. Escreva um programa em linguagem Pascal que faça um objeto gráfico se deslocar pela tela (da coluna 1 até a 80 e vice-versa), na velocidade desejada pelo usuário. Utilize a tecla ESC para encerrar a locomoção do objeto.
  9. Escreva um programa em linguagem Pascal que simule um piano no computador. Para encerrar sua utilização, basta pressionar a tecla ESC.
  10. Escreva um programa em linguagem Pascal que leia todas as notas de todas as disciplinas de um aluno e calcule a média final por disciplina. Apresente em vídeo todas as médias finais e a mensagem: APROVADO ou REPROVADO, de acordo com suas médias. A mensagem deverá ser exibida em amarelo intenso piscante. *Sugestão: Utilize uma tabela para a digitação das médias.*
  11. Escreva um programa em linguagem Pascal que troque a cor de fundo do vídeo, a cada vez que pressionarmos uma tecla qualquer. O programa deverá explorar todas as cores existentes no Turbo e a cada vez que o usuário pressionar qualquer tecla, um *beep* deverá ser emitido pelo programa.
  12. Um fazendeiro que planta soja, algodão, trigo e milho deseja saber quanto vai lhe custar o seguro contra chuvas de granizo em sua propriedade. Sabe-se que o valor do seguro cobrado é de 3,5% do valor total assegurado por colheita. Escreva um programa em linguagem Pascal que calcule o valor total do seguro por cultura e o total geral que o fazendeiro deverá pagar.
  13. Escreva um programa em linguagem Pascal que após colocar uma cor de fundo na tela do computador, divida-a em três pequenas janelas e desenhe pausadamente na primeira a forma de onda da função matemática SIN, na segunda, da função TAN e na última, da função COS.

# 11. Procedimentos (PROCEDURES)

**Procedure** define um conjunto de comandos (rotina) que pode ser executado uma ou mais vezes. Um procedimento pode ter a mesma estrutura de um programa, ter declaração de variáveis e de constantes. Sua principal particularidade é que todo procedimento deve ser definido antes de mencionado no programa, ou seja, deve vir antes do bloco principal. Outra característica importante é que se um procedimento tiver uma chamada de outro procedimento, este deverá ser definido previamente. Sua sintaxe:

```
PROCEDURE nome[<parâmetros:tipo>]; [EXTERNAL;]
                                [INTERRUPT;]
                                [FORWARD;]
                                [TYPE;]
                                [VAR;]
                                [CONST;]

BEGIN
    <comandos;>
END;
```

EXTERNAL	indica se uma PROCEDURE ou uma FUNCTION será compilada em separado.
INTERRUPT	declara uma rotina como sendo de interrupção para o compilador.
FORWARD	informa ao compilador que uma determinada rotina será executada após a sua chamada.
TYPE	declara os tipos de variáveis definidas pelo programador. Se declarado dentro da procedure, as variáveis serão chamadas de variáveis locais.
VAR	área de declaração de variáveis de um programa ou rotina.
CONST	define a área onde são definidas as constantes de um programa.

## Exercícios resolvidos

- Escreva um programa em linguagem Pascal que desenvolva as funções apresentadas no menu abaixo.

Escolha uma opção para visualizar figuras na tela

Menu de Opções

1. Visualizar uma figura...
2. Visualizar duas figuras...
3. Encerrar o programa...

Escolha sua opção --> <--

```
{ Nome do programa ..... : PROCED_1.PAS
  Função..... : Aplica o conceito de procedures
  Data de criação..... : 10/02/96
  Data da última alteração..... : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin  }

PROGRAM proced_1;
USES CRT;

VAR i, opcao : BYTE;
    fim : BOOLEAN;

PROCEDURE tela1; { monta a 1ª tela }
BEGIN
    CLRSCR;
    TEXTCOLOR (15);
    GOTOXY (1,1); WRITE ('-'); GOTOXY (79,1); WRITE ('-');
    GOTOXY (1,25); WRITE ('L'); GOTOXY (79,25); WRITE ('J');
    GOTOXY (1,25); WRITE ('L'); GOTOXY (79,25); WRITE ('J');
```

```

FOR i:= 2 TO 78 DO
BEGIN
    GOTOXY (i,1); WRITE ('-'); GOTOXY (i,5); WRITE ('-');
    GOTOXY (i,25); WRITE ('-');
    IF i<25
    THEN BEGIN
        GOTOXY (1,i); WRITE ('|'); GOTOXY (79,i); WRITE ('|');
        END;
    END;
GOTOXY (15,3); WRITE ('Escolha uma opção para visualizar figuras na tela');
GOTOXY (33,9); WRITE ('Menu de Opções');
GOTOXY (33,10); WRITE ('-----');
GOTOXY (27,12); WRITE ('1. Visualizar uma figura...');
GOTOXY (27,14); WRITE ('2. Visualizar duas figuras...');
GOTOXY (27,16); WRITE ('3. Encerrar o programa...');
GOTOXY (50,20); WRITE ('Escolha sua opção --> <--');
GOTOXY (71,20); opcao:= READKEY;
WHILE opcao > 3 DO
BEGIN
    GOTOXY (71,20); opcao:= READKEY;
    END;
fim:= FALSE;
END;                                     { finaliza a 1ª tela }

PROCEDURE tela2;                         { monta a 2ª tela }
BEGIN
    GOTOXY(15,3); WRITE ('');
    GOTOXY (33,9); WRITE ('');
    GOTOXY (33,10); WRITE ('');
    GOTOXY (27,12); WRITE ('');
    GOTOXY (27,14); WRITE ('');
    GOTOXY (27,16); WRITE ('');
    GOTOXY (50,20); WRITE ('');
    FOR i:= 2 TO 78 DO
    BEGIN
        GOTOXY (i,5); WRITE (' '); GOTOXY (i,21); WRITE ('-');
    END;
    TEXTCOLOR (30);
    GOTOXY (19,23); WRITE ('Pressione barra de espaços para continuar...');
    TEXTCOLOR (15);
END;                                     { finaliza a 2ª tela }

PROCEDURE uma_figura;                     {início da criação de 1 figura}
BEGIN
    FOR i:= 8 TO 15 DO
    BEGIN
        GOTOXY (30,i); WRITE ('██████████████████'); {18x ALT + 219}
    END;
    GOTOXY (65,23);
    REPEAT UNTIL KEYPRESSED;
END;                                     {fim da criação de uma figura}

PROCEDURE duas_figuras;                   {início da criação de 2 figuras}
BEGIN
    FOR i:= 3 TO 11 DO
    BEGIN
        GOTOXY (15,i); WRITE ('██████████████████');
        GOTOXY (50,i+8); WRITE ('██████████████████');
    END;
    GOTOXY (65,23);
    REPEAT UNTIL KEYPRESSED;
END;                                     {fim da criação de 2 figuras}

BEGIN                                   {início do bloco principal}
    REPEAT
        tela1;
        tela2;
        CASE OPCA OF
            1: uma_figura;
            2: duas_figuras;
            3: fim:= true;
        END;
    UNTIL fim= TRUE;
    CLRSCR;

```

END.

{fim do bloco principal}

**Listagem do programa 20**

2. Escreva um programa em linguagem Pascal, utilizando *procedure*, de forma que o programa principal chame a *procedure* para efetuar a leitura da tecla pressionada pelo usuário. Somente as teclas S e N deverão ser consideradas na leitura. Caso o usuário pressione qualquer tecla diferente, o programa deverá emitir um *beep* e acumular quantas vezes o usuário errou na digitação. É solicitado também que o usuário seja informado constantemente sobre o número de vezes que a *procedure* foi executada. Disponibilize ainda a opção de continuar ou não a execução da *procedure* que lê as entradas de dados.

```
{ Nome do programa ..... : PROCED_2.PAS
  Função..... : Aplica o conceito de procedures
  Data de criação..... : 10/02/96
  Data da última alteração.. : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin    }

PROGRAM procedur_2;
USES CRT;

VAR
  tecla : CHAR;
  n      : BYTE;

PROCEDURE le_tecla;
VAR
  n      : BYTE;
BEGIN
  n:= 0;
  REPEAT
    tecla:= UPCASE(READKEY);
    IF (tecla<>'N') AND (tecla<>'S')
      THEN
        BEGIN
          n:= n + 1;
          WRITELN; WRITELN(#7,'Número de erros = ',n:2);
        END;
  UNTIL tecla IN ['S','N'];
END;

BEGIN
  CLRSCR;
  n:= 0;
  REPEAT
    WRITELN; WRITELN;
    WRITELN('Deseja executar esta rotina novamente S/N ???');
    le_tecla;
    n:= n + 1;
    WRITELN; WRITELN('Procedure lê_tecla executada ',n:2,' vez(es)');
  UNTIL tecla = 'N';
END.
```

**Listagem do programa 21**

3. No programa a seguir, encontramos uma outra forma de resolução do problema anterior. Observe entretanto, a passagem de parâmetros a uma *procedure*. Este recurso é muito utilizado na prática sempre que necessitamos efetuar algum cálculo dentro da *procedure*, com valores externos a esta.

```
{ Nome do programa ..... : PROCED_3.PAS
  Função..... : Aplica o conceito de procedures com passagem de parâmetros
  Data de criação..... : 10/02/96
  Data da última alteração.. : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin    }

PROGRAM procedur_3;
USES CRT;

VAR
  tecla : CHAR;
  n      : BYTE;

PROCEDURE le_tecla(m: BYTE);
VAR
  n      : BYTE;
```

```

BEGIN
  n:= 0;
  REPEAT
    tecla:= UPCASE(READKEY);
    IF (tecla<>'N') AND (tecla<>'S')
      THEN
        BEGIN
          n:= n + 1;
          WRITELN; WRITELN(#7,'Número de erros = ',n:2);
        END;
    UNTIL tecla IN ['S','N'];
    WRITELN;
    WRITELN('Procedure lê_tecla executada ',m:2,' vez(es)');
  END;

  BEGIN
    CLRSCR;
    n:= 0;
    REPEAT
      WRITELN; WRITELN; WRITELN;
      WRITELN('Deseja executar esta rotina novamente S/N ???');
      n:= n + 1;
      le_tecla(n);
    UNTIL tecla = 'N';
  END.

```

### Listagem do programa 22

4. A seguir, encontramos uma nova forma de resolução do mesmo problema, ainda com o recurso de passagem de parâmetros a uma procedure.

```

{ Nome do programa ..... : PROCED_4.PAS
  Função..... : Aplica o conceito de procedures com passagem de parâmetros
  Data de criação..... : 10/02/96
  Data da última alteração.. : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin    }

PROGRAM procedur_4;
USES CRT;

VAR
  tecla : CHAR;
  n      : BYTE;

PROCEDURE le_tecla (VAR m: BYTE);
VAR
  n      : BYTE;

BEGIN
  n:= 0;
  REPEAT
    tecla:= UPCASE(READKEY);
    IF (tecla<>'N') AND (tecla<>'S')
      THEN
        BEGIN
          n:= n + 1;
          WRITELN; WRITELN(#7,'Número de erros = ',n:2);
        END;
    UNTIL tecla IN ['S','N'];
    m:= m+ 1;
  END;

  BEGIN
    CLRSCR;
    n:= 0;
    REPEAT
      WRITELN; WRITELN; WRITELN;
      WRITELN('Deseja executar esta rotina novamente S/N ???');
      le_tecla(n);
      WRITELN;
      WRITELN('Procedure lê_tecla executada ',n:2,' vez(es)');
    UNTIL tecla = 'N';
  END.

```

### Listagem do programa 23

5. No exemplo a seguir, observamos uma *procedure* recebendo valores numéricos e uma *string*, tratando-os individualmente e executando seu processamento.

```
{ Nome do programa..... : PROCED_5.PAS
  Função..... : Aplica o conceito de procedures com passagem de parâmetros
  Data de criação..... : 10/02/96
  Data da última alteração.. : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin    }

PROGRAM procedur_5;
USES CRT;

VAR
    tecla : CHAR;

PROCEDURE escreve_em_xy (linha, coluna: BYTE; mensagem: STRING);
BEGIN
    GOTOXY(linha, coluna);
    WRITE(mensagem);
END;

BEGIN
    CLRSCR;
    escreve_em_xy (40,5, 'linha 40 e coluna 5');
    escreve_em_xy (10,12, 'linha 10 e coluna 12');
    escreve_em_xy (20,15, 'linha 20 e coluna 15');
    escreve_em_xy (50,22, 'linha 50 e coluna 22');
    READKEY;
END.
```

#### Listagem do programa 24

6. No exemplo a seguir, observamos a solução do problema anterior através de um operador **BOOLEANO** que será utilizado no controle da execução da *procedure*, ou seja, a *procedure* será executada até que o valor recebido for **TRUE**.

```
{ Nome do programa..... : PROCED_6.PAS
  Função..... : Aplica o conceito de procedures com passagem de parâmetros
  Data de criação..... : 10/02/96
  Data da última alteração ... : 2/4/2003
  Escrito por..... : Francisco Carlos Mancin    }

PROGRAM procedur_6;
USES CRT;

VAR
    tecla : CHAR;
    n      : BYTE;

PROCEDURE escreve_em_xy(linha, coluna: BYTE; mensagem: STRING; repete: BOOLEAN);
BEGIN
    GOTOXY(linha, coluna);
    WRITE(mensagem);
    IF repete
    THEN
        BEGIN
            escreve_em_xy (10,25,'Pressione algo para continuar', FALSE);
            REPEAT UNTIL READKEY <> #0;
            escreve_em_xy (10,25,'', FALSE);
        END;
    END;

BEGIN
    CLRSCR;
    escreve_em_xy (40,5, 'linha 40 e coluna 5', TRUE);
    escreve_em_xy (10,12, 'linha 10 e coluna 12', TRUE);
    escreve_em_xy (20,15, 'linha 20 e coluna 15', TRUE);
    escreve_em_xy (50,22, 'linha 50 e coluna 22', TRUE);
END.
```

#### Listagem do programa 25

7. Escreva um programa em linguagem Pascal que faça um quadradinho (■) se deslocar pela tela através das teclas de direção (setas), até que o usuário pressione a tecla Esc.

```
{ Nome do programa..... : ANDA.PAS
  Função ..... : Faz um quadradinho se deslocar pela tela até pressionar Esc
  Data de criação ..... : 10/02/96
  Data da última alteração ... : 20/2/2001
  Escrito por ..... : Francisco Carlos Mancin  }

PROGRAM ANDA;
USES CRT;
VAR
  tecla : CHAR;
  Lin, Col : Byte;

Procedure Cursor;
Begin
  Clrscr;
  Gotoxy (Col,lin); Write('■');
End;

Procedure Acima;
Begin
  Lin := Lin - 1;   If Lin <= 0 Then Lin := 23;
  Cursor;
End;

Procedure Abaixo;
Begin
  Lin := Lin + 1;   If Lin > 23 Then Lin := 1;
  Cursor;
End;

Procedure Esquerda;
Begin
  Col := Col - 1;   If Col <= 1 Then Col := 79;
  Cursor;
End;

Procedure Direita;
Begin
  Col := Col + 1;   If Col >= 79 Then Col := 1;
  Cursor;
End;

BEGIN
  TextBackground(Blue); Textcolor(Yellow); CLRSCR;
  Gotoxy(19,1); Write ('Use as setas de direção para mover o cursor');
  GOTOXY (25,25); Write ('Pressione a tecla <Esc> para encerrar...');
  TextColor(White); Window (1,2,80,24);
  Lin := 11; Col := 40;
  Cursor;
  REPEAT
    tecla:= readkey;
    Case Ord(tecla) of
      72: Acima;
      75: Esquerda;
      77: Direita;
      80: Abaixo;
    End;
  UNTIL Tecla = #27;
  ClrScr;
END.
```

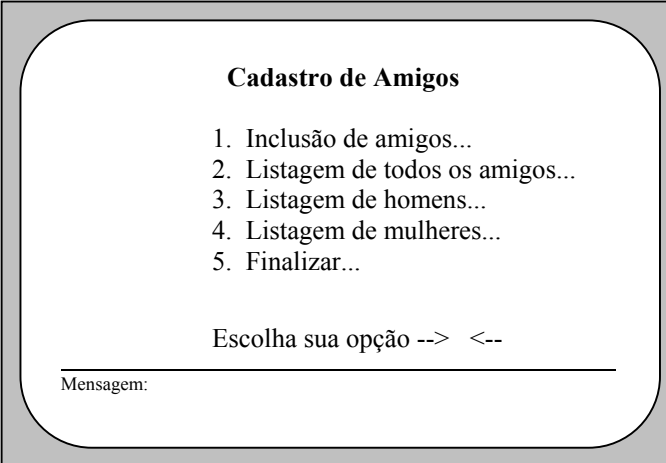
### Listagem do programa 26

## Exercício de fixação

1. Escreva um programa em linguagem Pascal, empregando a utilização de *procedures* de modo que cada uma execute apenas uma função. As *procedures* desejadas são: uma que deseja uma moldura ao redor da tela; uma para escrever o menu principal, uma para inclusão, uma para listagem completa, uma para listagem apenas de pessoas do sexo M; e outra para listagem de pessoas do sexo F.

**Sugestão:** observe o funcionamento do programa com o professor, antes de executá-lo.





**Cadastro de Amigos**

1. Inclusão de amigos...
2. Listagem de todos os amigos...
3. Listagem de homens...
4. Listagem de mulheres...
5. Finalizar...

Escolha sua opção --> <--

Mensagem: \_\_\_\_\_

O programa deverá:

- efetuar a validação dos dados de entrada;
- possibilitar a inclusão de novos amigos;
- dispor de recursos de controle de rolagem de tela.

## 12. Tipos Estruturados (ARRAY)

Permitem a criação de conjuntos, registros ou a definição de arquivos. A declaração ARRAY define a construção de matrizes e sua utilização é certamente indispensável em linguagens de programação. Estas podem apresentar um número fixo de elementos definidos por uma faixa de índices, que por sua vez pode ter mais de um nível (ou seja, podemos encontrar matrizes unidimensionais, bidimensionais, tridimensionais etc.). Os elementos são de tipos predefinidos pelo programador. Uma matriz contém, na maioria das vezes, elementos comuns entre si. Em Pascal, o maior tamanho que uma matriz pode assumir é de 64kbytes, correspondente ao tamanho da área de dados de um programa.

Para exemplificarmos a utilização de matriz, imaginemos a seguinte situação: precisamos armazenar dados ao longo de 12 meses, e para isto poderíamos ter 12 variáveis diferentes e estar em locais diferentes de memória, como:

```
VAR
    mes01, mes02, mes03, mes04, mes05, mes06,
    mes07, mes08, mes09, mes10, mes11, mes12 : STRING;
```

ou ainda, ser guardados em uma área contínua de memória, permitindo um acesso mais fácil:

```
VAR
    meses : ARRAY [1..12] OF STRING;
```

A referência a qualquer um dos elementos da matriz é efetuada através do índice da matriz, o qual representa a posição do elemento desejado. Ex.: *para visualizarmos o conteúdo da quarta posição da matriz, basta solicitarmos: **WRITE (meses[4]);***.

### Exercícios resolvidos

1. Escreva um programa em linguagem Pascal, utilizando vetor, que leia quantos números inteiros o usuário deseje e calcule o valor médio de todos os números. Após calcular, o valor médio deverá ser mostrado no vídeo.

```
{ Nome....: Vetor
  Função... Implementar a utilização de vetores
  Data....: 14/08/97
  Autor...: Prof. Xico
}

Program VETOR;
Uses crt;
Const maximo = 500;
Var numero      : array[1..maximo] of longint;
    i, num_tot, val_tot : integer;
    valor_md      : real;
    opcao         : char;

Begin
    window(1,1,80,25); textbackground(black); CLRSCR;
    textcolor (15);
    gotoxy (1,1);   write ('P'); gotoxy (79,1);   write ('Q');
    gotoxy (1,25);  write ('L'); gotoxy (79,25);  write ('J');
    for i:= 2 TO 78 DO
        begin
            gotoxy (i,1); write ('='); gotoxy (i,5); write ('-');
            gotoxy (i,25); write ('=');
            if i<25 then
                begin
                    gotoxy (1,i); write ('|'); gotoxy (79,i); write ('|');
                end;
        end;
    gotoxy (17,3); write ('Lê uma relação de números e calcula o valor médio');
    gotoxy (19,8); write ('Digite o total de números desejado (1-500): ');
    readln(num_tot);
    window(20,10, 60, 18); textbackground (blue); clrscr;
    gotoxy(12, 2); write('Digitando os valores');
    for i:= 1 to num_tot do
        begin
            gotoxy(10,7); write('Digite o ', i:3,'º número: '); readln(numero[i]);
            gotoxy(10,4); delLine;
            val_tot:= val_tot + numero[i];
        end;
end;
```

```

    end;
    valor_md:= val_tot / num_tot;
    clrscr; gotoxy(8,6); write('O valor médio é: ', valor_md:8:3);
    textcolor(yellow + blink); gotoxy (5,9); write('Pressione qq tecla p/ encerrar');
    repeat until keypressed;
End.

```

### Listagem do programa 27

2. Escreva um programa em linguagem Pascal, utilizando vetor, que leia o nome, sexo e a idade de 5 pessoas e, disponibilize ao usuário a opção de ver somente os dados das pessoas de um respectivo sexo.

```

{ Nome...: Pessoas.Pas
  Função.: Ler o nome, a idade e o sexo de 5 pessoas
  Data...: 10/11/99
  Autor...: Francisco Carlos Mancin
}

Program Pessoas;
Uses Crt;
Const max = 5;

Var Nome : Array [1..max] Of String;
    Sexo : Array [1..max] Of Char;
    Idade: Array [1..max] Of Byte;
    i    : Byte;
    Resp : Char;

Begin
  TextColor(White); TextBackground(Blue); ClrScr;
  Gotoxy(20,5); Write('Exemplifica a utilização de vetores');
  Gotoxy(30,8); Write('Vamos digitar dados de ', max:2, ' pessoas');
  Gotoxy(15,10); Write('Sexo Idade Nome');
  For i:= 1 to max do
    Begin
      Gotoxy(12,11+i); Write(i, '.');
      Gotoxy(15,11+i); Sexo[i] := Upcase(Readkey); Write(Sexo[i]);
      Gotoxy(21,11+i); Readln(Idade[i]);
      Gotoxy(29,11+i); Readln(Nome[i]);
    End;
  ClrScr;
  Gotoxy(15,10); Write('Deseja ver a listagem dos homens(M) ou mulheres(F)? ');
  Resp := UpCase(Readkey); ClrScr;
  For i:= 1 to max Do
    If Sexo[i] = Resp Then
      Begin
        Gotoxy(15,11+i); Write(Sexo[i]:5, Idade[i]:3, ' ', Nome[i]);
      End;
  Readkey;
End.

```

### Listagem do programa 28

3. Escreva um programa em linguagem Pascal, utilizando matriz, que gere automaticamente 45 números aleatórios contidos entre 0 e 100 e os coloque numa matriz. Após a geração dos 45 números, eles deverão ser mostrados na tela, tabulados em 3 colunas. Disponibilize ainda ao usuário a possibilidade de executar novamente o programa.

```

{ Nome..... : MATRIZ_1.PAS
  Função ..... : Gera uma matriz de 45 elementos e os apresenta em vídeo
  Data ..... : 2/4/2003
  Autor ..... : Francisco Carlos Mancin }

PROGRAM MATRIZ_1;
USES CRT;
CONST tot_num = 45;

VAR numeros      : ARRAY[1..tot_num] OF BYTE;
    fim          : BOOLEAN;
    i, opcao, lin : BYTE;
    tecla        : CHAR;

BEGIN
  REPEAT
    HIGHVIDEO; TEXTCOLOR(WHITE); CLRSCR;
    GOTOXY (31,3); WRITE ('Trabalhando com Matrizes');
    GOTOXY (33,7); WRITE ('Menu de Opções');

```

```

GOTOXY (33,08); WRITE ('-----');
GOTOXY (32,10); WRITE ('1. Gerar números...');
GOTOXY (32,12); WRITE ('2. Listar números...');
GOTOXY (32,14); WRITE ('3. Fim...');
GOTOXY (50,19); WRITE ('Escolha sua opção -->  <--');
GOTOXY (72,19); READLN (opcao);
WHILE opcao > 3 DO
  BEGIN
    GOTOXY (71,19); WRITE ('  <--');
    GOTOXY (72,19); READLN (opcao);
  END;
FIM:= FALSE;

CASE opcao OF
  1: BEGIN
    RANDOMIZE;
    GOTOXY(24,24); WRITE ('Os números estão sendo gerados...', #7);
    DELAY(1000);
    FOR i:= 1 TO tot_num DO
      BEGIN
        numeros[i]:= TRUNC(RANDOM(100));
      END;
    TEXTCOLOR(LIGHTCYAN+BLINK);
    GOTOXY(17,24); WRITE ('Processamento concluído...  Pressione a tecla
      ESC');
    REPEAT tecla:=READKEY UNTIL tecla= #27;
    GOTOXY(1,24); DELLINE;
  END;
  2: BEGIN
    CLRSCR;
    lin:= 6;
    GOTOXY(28,3); WRITE('Os números da tabela são:');
    FOR i:= 1 TO 45 DO
      BEGIN
        IF i < 16
          THEN
            BEGIN
              GOTOXY(20,lin); WRITE(i:2,'º') ',numeros[i]);
              lin:= lin + 1;
            END
          ELSE IF i < 31
            THEN
              BEGIN
                IF i= 16 THEN lin:= 6;
                GOTOXY(40,lin); WRITE(i,'º') ',numeros[i]);
                lin:= lin + 1;
              END
            ELSE
              BEGIN
                IF i= 31 THEN lin:= 6;
                GOTOXY(60,lin); WRITE(i,'º') ',numeros[i]);
                lin:= lin + 1;
              END;
            END;
        GOTOXY(28,24); WRITE('Pressione a tecla <ESC>');
        REPEAT tecla:=READKEY UNTIL tecla= #27;
      END;
    3: FIM:= TRUE;
  END;
UNTIL FIM= TRUE;
CLRSCR;
END.

```

### Listagem do programa 29

4. Escreva um programa em linguagem Pascal que gere uma matriz bidimensional de alunos, contendo suas notas em cada bimestre. O programa deverá oferecer a opção ao usuário de inserir novos bimestres ou não.

```

{ Nome..... : MATRIZ_2.PAS
  Função ..... : Gera uma matriz bidimensional, contendo alunos e notas
  Data ..... : 2/4/2003
  Autor ..... : Francisco Carlos Mancin }

```

```

PROGRAM matriz_2;
USES CRT;

CONST
    maximo = 5;

VAR
    notas           : ARRAY[1..4, 1..maximo] OF BYTE;
    opcao           : CHAR;
    aluno, bimestre : BYTE;

BEGIN
    REPEAT
        CLRSCR;
        GOTOXY(10,1); WRITE('Digitação de notas bimestrais');
        GOTOXY(1,3); WRITE('Digite o bimestre para as notas: ');
        READLN(bimestre); WRITELN;
        FOR aluno:= 1 TO maximo DO
            BEGIN
                WRITE('Nota do aluno nº ',aluno:2,': ');
                READLN(notas[bimestre,aluno]);
            END;
        WRITE('Digite algo para continuar...');
        REPEAT UNTIL READKEY <> #0;
        CLRSCR;
        GOTOXY(10,1); WRITE('As notas do ',bimestre:2,'º Bimestre');
        WRITELN;
        FOR aluno:= 1 TO maximo DO
            BEGIN
                GOTOXY(15,aluno+1);
                WRITELN('Nota do aluno nº ',aluno:2,' --> ',notas[bimestre,aluno]);
            END;
        WRITELN('Deseja um novo bimestre S/N ?? ');
        REPEAT opcao:= UPCASE(READKEY) UNTIL opcao IN ['S','N'];
        UNTIL opcao = 'N';
    END.

```

### Listagem do programa 30

5. Escreva um programa em linguagem Pascal que gere uma matriz bidimensional que armazene as notas de 4 bimestres de quantos alunos eu desejar e, após a leitura, mostre-as na tela.

```

{ Nome      : MATRIZ_3.PAS
  Função    : Gera uma matriz bidimensional, contendo alunos e notas
  Data      : 2/4/2003
  Autor     : Francisco Carlos Mancin  }

Program Matriz_3;
Uses CRT;

Const bimestres = 4;
      alunos    = 2;

Var notas           : Array[1..alunos, 1..bimestres] of real;
    media           : real;
    col, lin, aux    : byte;
    opcao           : char;

Begin
    opcao:= 'S';
    while opcao = 'S' do
        Begin
            window (1,1,80,25); textcolor (white); textbackground (0); clrscr;
            gotoxy (31,2); write ('Trabalhando com Matrizes');
            window (5,3,75,23); textbackground (0); clrscr;
            gotoxy (18,2); write ('Aluno  1ºBim   2ºBim   3ºBim   4ºBim');

            for lin:= 1 to alunos do
                begin
                    aux := 28;
                    gotoxy (20,lin+4); write (lin,'. ');
                    for col := 1 to bimestres do
                        begin
                            gotoxy (aux,lin+4); readln (notas[lin,col]);

```

```

        aux:= aux + 10;
    end;
end;

clrscr; gotoxy (20,10); write ('Digitação de notas concluídas...');
readkey;
gotoxy (20,10); write ('Deseja visualizar a média (S/N) ?');
opcao := upcase (readkey);
if opcao = 'S' then
begin
    clrscr;
    gotoxy (13,3); write ('Aluno    1ºBim    2ºBim    3ºBim    4ºBim    Média');
    for lin := 1 to alunos do
        begin
            aux:= 23;
            gotoxy (15,lin+4); write (lin, '.');
            for col := 1 to bimestres do
                begin
                    media := media + notas[lin, col];
                    gotoxy (aux,lin+4); write (notas[lin,col]:4:1);
                    aux:= aux+8;
                end;
            gotoxy (aux,lin+4); write (media / bimestres:5:2);
            media := 0;
        end;
    end;
    readkey;
    clrscr; gotoxy (20,10); write ('Deseja continuar (S/N) ?');
end;
clrscr;
end.

```

### Listagem do programa 31

## Exercícios de fixação

- Escreva um programa em linguagem Pascal que nos possibilite trabalhar com uma matriz 5x5 e que disponibilize todas as opções apresentadas no menu a seguir:

**Cálculos com uma Matriz 5 x 5**

1. Incluir elementos na matriz
2. Listar matriz
3. Calcular determinante
4. Exibir matriz transposta
5. Finalizar...

Escolha sua opção --> <--

---

Mensagem:

O programa deverá:

- efetuar a validação dos dados de entrada (digitação de valores e menu);
- possibilitar a inclusão de novos elementos na matriz;
- apresentação das mensagens de erro em área reservada.

## 13. Ordenação de dados

Existem 2 métodos de ordenação de dados que podem ser aplicados na programação: método bolha e método binário. Não iremos entrar em detalhes em relação às vantagens e características de cada um dos métodos. Nosso objetivo é fazer uma classificação de dados que seja simples e funcional.

Quando trabalhamos com ordenação, precisamos trabalhar com duas variáveis indexadas simultaneamente e que as duas olhem para o mesmo vetor. Dessa forma, comparamos cada elemento dentro do vetor com todos os outros e, quando encontramos um elemento menor que aquele que está sendo comparado, o trazemos para a posição do atual, sempre com o auxílio de uma terceira variável indexada.

O processo parece ser um tanto estranho e complicado, mas através do programa a seguir poderemos entendê-lo facilmente.

### Exercício resolvido

1. Escreva um programa em linguagem pascal que leia quantos números o usuário desejar (até 100 números) reais e os armazene em um vetor. Após ler todos os números, o programa deverá classificá-los (ordem crescente) e os apresentar no vídeo em forma de 4 colunas.

```
{ Nome.....: ORDENA.PAS
  Função.....: Ordenar até 100 números reais
  Autor.....: Francisco Carlos Mancin
  Data.....: 12/09/97
}

PROGRAM ORDENA;
USES CRT;
CONST
  Max_Elem = 100;
VAR
  Vetor      : ARRAY[1..Max_Elem] OF REAL;
  Temp       : REAL;
  Qtde, I, J : INTEGER;
BEGIN
  CLRSCR;
  REPEAT
    GOTOXY(5,5);
    WRITE('Quantidade de elementos a ordenar : '); READLN(Qtde);
    IF Qtde > Max_Elem THEN
      BEGIN
        GOTOXY(10,10); CLREOL;
        WRITE('O Número Máximo Permitido é ',Max_Elem, ' números. ');
        READKEY;
        CLRSCR;
      END;
    UNTIL Qtde <= Max_Elem;

    FOR I := 1 TO Qtde DO
      BEGIN
        GOTOXY(10,15); CLREOL;
        GOTOXY(10,15); WRITE('Entre com o ',I,'º elemento : '); READ(Vetor[I]);
      END;

    FOR I := 1 TO Qtde-1 DO
      BEGIN
        FOR J := I+1 TO Qtde DO
          BEGIN
            IF Vetor[I] > Vetor[J] THEN
              BEGIN
                Temp      := Vetor[I];
                Vetor[I] := Vetor[J];
                Vetor[J] := Temp;
              END;
          END;
        END;
      END;
  END;
```

```
END;  
END;  
  
CLRSCR;  
FOR I := 1 TO Qtde DO  
BEGIN  
  CASE I OF  
    1..25:  
      BEGIN  
        GOTOXY(1,I);  
        WRITE(I:3,' ') ',Vetor[I]:13:3);  
      END;  
    26..50:  
      BEGIN  
        GOTOXY(20,I-25);  
        WRITE(I:3,' ') ',Vetor[I]:13:3);  
      END;  
    51..75:  
      BEGIN  
        GOTOXY(40,I-50);  
        WRITE(I:3,' ') ',Vetor[I]:13:3);  
      END;  
    76..100:  
      BEGIN  
        GOTOXY(60,I-75);  
        WRITE(I:3,' ') ',Vetor[I]:13:3);  
      END;  
  END  
END;  
REPEAT UNTIL KEYPRESSED;  
END.
```

### Listagem do programa 32

## Exercícios de fixação

1. Escreva um programa em linguagem pascal que gere 100 números inteiros (0 → 1000) e os guarde em um vetor. O programa deverá executar todas as funções representadas no menu a seguir. Utilize *procedures* em todos os processamentos.

**Trabalhando com Números Inteiros**

1. Gerar 100 números (1 → 1000)
2. Classificar números
3. Listar números pares
4. Listar números ímpares
5. Imprimir relatório
6. Finalizar...

Escolha sua opção --> <--

Mensagem: \_\_\_\_\_

O programa deverá:

- apresentação das mensagens de advertência em área reservada..



## 14. Funções (FUNCTIONS)

Uma *function* é uma rotina que nos retorna um determinado valor. A definição e sintaxe de uma *function* é bastante parecida com a de uma *procedure*. Assim como uma *procedure*, a *function* deve ser declarada antes de sua chamada. Cada *function* retorna apenas um resultado, o qual será de acordo com o seu tipo, ou seja, se a *function* for REAL, o retorno será um valor REAL; se for BOOLEAN, o retorno será um BOOLEAN e assim sucessivamente.

### Exercícios resolvidos

1. Escreva um programa em linguagem Pascal, utilizando *function*, onde sua tarefa básica é a de verificação se o usuário digitou S ou N, retornando ao programa principal o correspondente ao conteúdo da digitação. **N.B.**: esta função nos retorna apenas S ou N e pode ser utilizada em qualquer aplicação.

```
{ Nome do programa .....: FUNCAO_1.PAS
  Função .....: Aplica o conceito de functions
  Data de criação .....: 10/02/96
  Data da última alteração : 2/4/2003
  Escrito por .....: Francisco Carlos Mancin  }

PROGRAM funcao_1;
USES CRT;

FUNCTION resposta: CHAR;
VAR
  tecla: CHAR;
BEGIN
  REPEAT
    tecla:= UPCASE(READKEY);
  UNTIL tecla IN ['S', 'N'];
  resposta:= tecla;
END;

BEGIN
  REPEAT
    CLRSCR; GOTOXY(20,10);
    WRITE('Você é do sexo masculino ? (S/N) ?? ');
    GOTOXY(20,15);
    IF resposta = 'S'
    THEN
      WRITE('Você disse que é homem !')
    ELSE
      WRITE('Você disse que é mulher !');
    GOTOXY(30,24); WRITE('Deseja continuar ? (S/N) ?? ');
  UNTIL resposta = 'N';
END.
```

### Listagem do programa 33

2. Escreva um programa em linguagem Pascal que após receber uma *string* qualquer converta-a em maiúscula, através de uma *function*.

```
{ Nome do programa .....: FUNCAO_2.PAS
  Função .....: Converte qualquer string em sua equivalente maiúscula
  Data de criação .....: 10/02/96
  Data da última alteração : 2/4/2003
  Escrito por .....: Francisco Carlos Mancin  }

PROGRAM funcao_2;
USES CRT;

VAR
  x: STRING;

FUNCTION maiuscula(s: STRING) : STRING;
VAR
  indice: BYTE;
BEGIN
  FOR indice:= 1 TO ORD(s[0]) DO
```

```

        s[indice]:= UPCASE(s[indice]);
        maiuscula:= s;
    END;

    BEGIN
        CLRSCR;
        GOTOXY(10,5);  WRITE('Digite uma frase: ');  READLN(x);
        GOTOXY(10,15);  WRITE('A frase convertida é: ', maiuscula(x));
    END.

```

#### Listagem do programa 34

3. Escreva um programa em linguagem Pascal que após ler dois números inteiros quaisquer, determine se são iguais ou não. Utilize *function* para efetuar a comparação. **N.B.**: Preste atenção que neste caso, o tipo da função independe do parâmetro passado.

```

{ Nome do programa.....: FUNCAO_3.PAS
  Função.....: Utiliza function para verificar se dois números são iguais
  Ambiente Operacional.....: DOS
  Linguagem.....: PASCAL v. 7.0
  Data de criação.....: 10/02/96
  Data da última alteração..: 2/4/2003
  Escrito por.....: Francisco Carlos Mancin  }

PROGRAM funcao_3;
USES CRT;

VAR
    n1, n2 : LONGINT;

FUNCTION compara(a, b : LONGINT): BOOLEAN;

BEGIN
    compara := a = b;
END;

BEGIN
    CLRSCR;
    GOTOXY(20,5);  WRITE('Digite o primeiro número: ');  READLN(n1);
    GOTOXY(20,8);  WRITE('Digite o segundo número: ');  READLN(n2);
    GOTOXY(30,18);  TEXTCOLOR(YELLOW);
    IF compara(n1, n2)
    THEN
        Writeln('Os números são iguais !')
    ELSE
        Writeln('Os números são diferentes !');
    READKEY;
END.

```

#### Listagem do programa 35

4. Escreva um programa em linguagem Pascal que apresente uma função cujo objetivo é calcular o valor de  $X^3$  e, após o cálculo mostre o resultado na tela. **N.B.**: Preste atenção que neste caso, o tipo da função independe do parâmetro passado.

```

{ Nome do programa.....: FUNCAO_4.PAS
  Função.....: Utiliza function para calcular o valor de X elevado a n
  Ambiente Operacional.....: DOS
  Linguagem.....: PASCAL v. 7.0
  Data de criação.....: 10/02/98
  Data da última alteração..: 2/4/2003
  Escrito por.....: Francisco Carlos Mancin  }

PROGRAM funcao_4;
USES CRT;

VAR
    numero      : Integer;

FUNCTION Cubo(valor : LongInt): LongInt;

BEGIN
    Cubo := valor * valor * valor;
END;

```

```
BEGIN
  CLRSCR;
  GOTOXY(20,5); WRITE('Digite o número a ser elevado ao cubo: ');
  READLN(numero);
  GOTOXY(30,18); TEXTCOLOR(YELLOW);
  WRITE ('O resultado de ', numero, '^3 é = ', Cubo(numero));
  READKEY;
END.
```

### Listagem do programa 36

## Exercícios de fixação

1. Escreva um programa em linguagem Pascal, utilizando FUNCTION, que ordene uma matriz unidimensional de  $n$  elementos. Após a ordenação, o resultado deverá ser mostrado na tela.
2. Escreva um programa em linguagem Pascal que leia 20 números inteiros quaisquer e apresente uma relação separada em duas colunas: uma só com números pares e outra só com os números ímpares. No final da relação deverá aparecer a somatória dos números pares e a dos números ímpares. **Obs.:** Utilize procedures para:
  - uma procedure para ler os números e armazenar em um vetor;
  - outra para separar os números em dois novos vetores (PAR e ÍMPAR);
  - outra para totalizar os pares e ímpares;
  - outra para mostrar os números e seus respectivos totais na tela.

Disponibilize ainda a opção de continuar sua execução ou não.

3. Escreva um programa em linguagem Pascal que disponibilize uma **função** (FATORIAL) que calcule o fatorial de um número inteiro qualquer. Disponibilize a opção de continuar sua execução ou não. Sabe-se que:

$$5! = 5 * 4 * 3 * 2 * 1 = 120 \qquad 7! = 7 * 6 * 5 * 4 * 3 * 2 * 1 = 5040$$

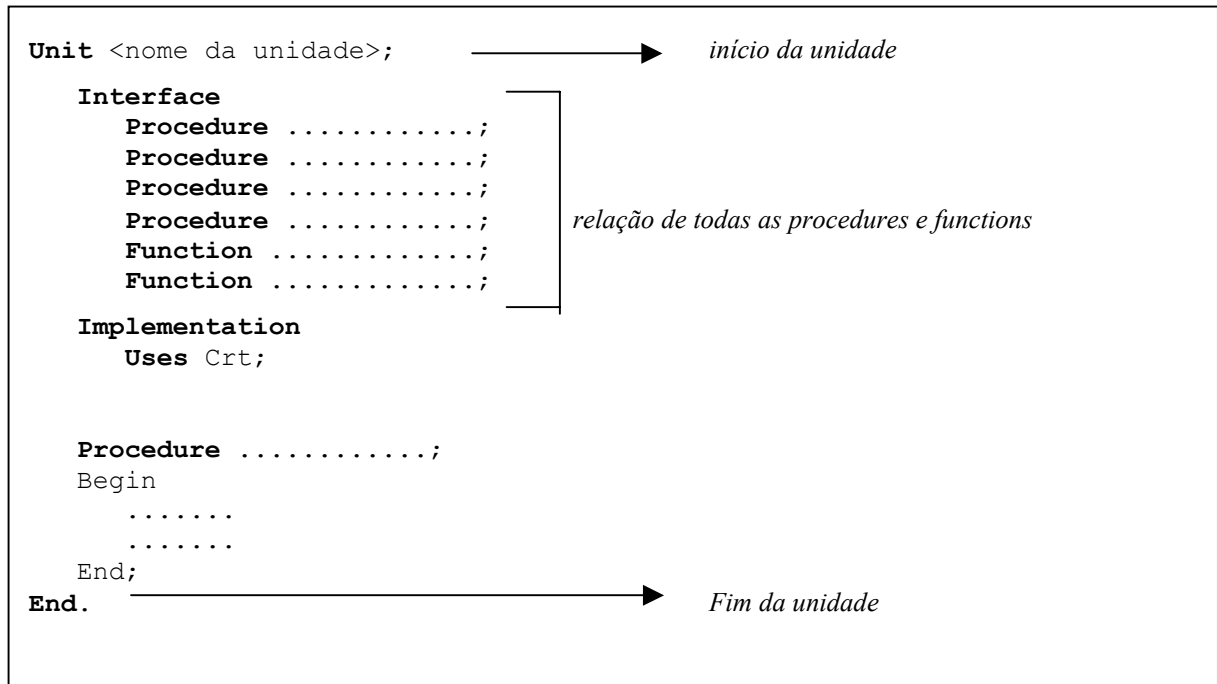
4. Escreva um programa em linguagem Pascal que disponibilize uma função (COLORS) que receba uma *string* qualquer e a devolva ao programa principal escrita contendo cada caracter de uma cor, sorteadas aleatoriamente.
5. Escreva um programa em linguagem Pascal após ler um número inteiro qualquer, escreva-o por extenso.

## 15. Criando sua própria Unit

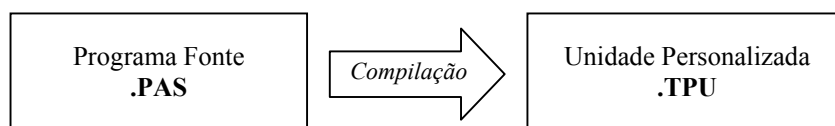
Uma das maiores facilidades encontradas no ambiente de programação do Pascal é a possibilidade que programador encontra em criar suas próprias bibliotecas de funções. Essas bibliotecas são chamadas de unidades e serão incorporadas aos futuros programas através da instrução **Uses**.

Toda unidade personalizada é composta por dois tipos de elementos previamente definidos que são as *procedures* e as *functions*. Um conjunto de *procedures* e *functions* são partes integrantes de uma unidade personalizada que poderá ser incorporada a qualquer programa que o programador deseje.

### Estrutura de uma unit



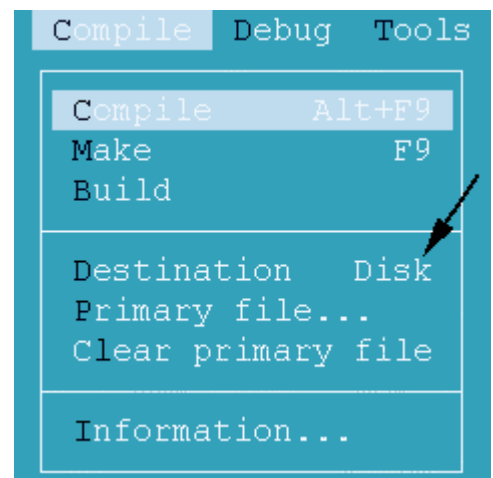
### Fazes na criação de uma unidade



A criação do código fonte deve seguir a mesma linha e ordem que a criação de um programa Pascal. A única diferença é que pelo fato de se tratar de uma *unit* você deverá respeitar sua estruturação apresentada acima.

Após a edição de todo o código fonte, chegou a hora da compilação (ALT + F9). É nesse momento que o compilador Pascal vai gerar uma unidade (.TPU) e não um programa executável (.EXE). Isso é obtido automaticamente devido a declaração **Unit** <nome da unidade>; que aparece na primeira linha do código fonte.

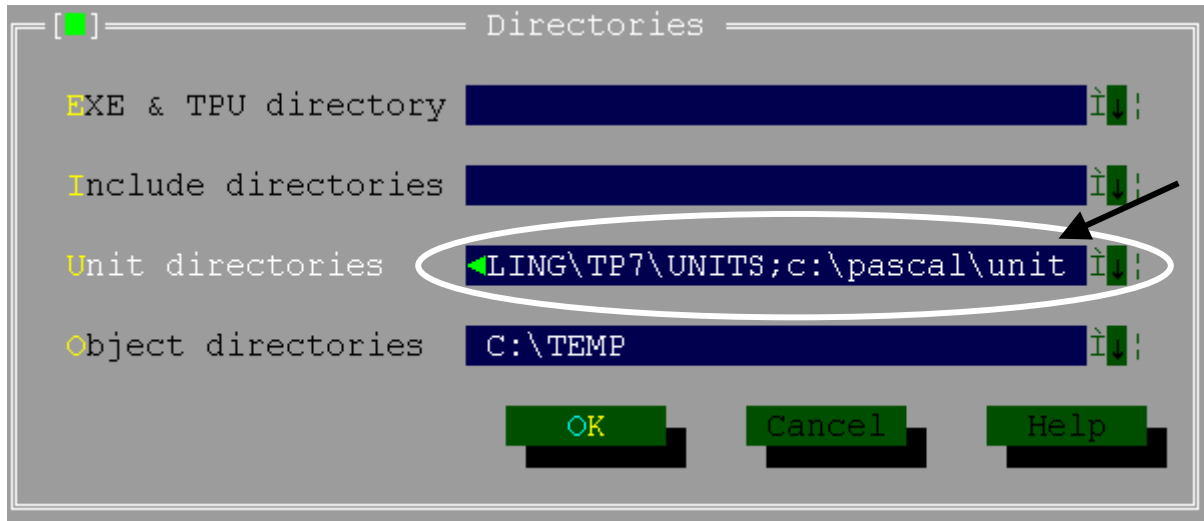
Como a nova unidade é um arquivo .TPU e que deverá estar presente no disco, a compilação deverá ser direcionada para o disco e não para a memória, que normalmente é feita com os programas em Pascal (apenas para verificar seu funcionamento). Para direcionar a compilação para o disco, basta “avisar” o Turbo Pascal, na opção **Compile ... Destination... Disk**, conforme ilustrado ao lado.



### Ajustando o diretório de busca da sua unidade

Certamente, por uma questão de organização, todas as suas unidades deverão e estarão reunidas em um único diretório. Este seu diretório deverá ser informado ao Turbo Pascal para que ele possa “encher” suas unidades e incorporá-las em seus programas.

Caso você se esqueça de informar onde a unidade (.TPU) se encontra, ao compilar se programa .PAS, o compilador mandará uma mensagem de erro informando-o que ele não encontrou sua referida unidade. O ajuste desse diretório deverá ser feito em **Options... Directories... Unit directories**, conforme ilustra a figura abaixo.



### Exercício Resolvido

1. Escreva uma unidade personalizada (My\_Unit.PAS) que execute algumas funções automaticamente que possa auxiliá-lo no processo de programação.

```
{ Nome.....: My_Unit.PAS
  Função...: Implementar a criação de uma unidade personalizada
  Autor...: Francisco Carlos Mancin
  Data....: 1/3/98
}

Unit My_Unit;

interface          {relação das procedures e funções contidas na unit}
  Procedure Writeln (lin, col: Byte; cor: Byte; texto: String);
  Procedure Janela  (col_ini, lin_ini, col_fin, lin_fin, fundo: Byte);
  Procedure Janelapadiao;
  Procedure Centerx (lin : Byte; texto : String);
  Procedure Linhasimples (lin, col_ini, col_fin : Byte);
  Procedure Linhadupla  (lin, col_ini, col_fin : Byte);
  Procedure Moldurasimples (col_ini, lin_ini, col_fin, lin_fin : Word);
  Procedure Molduradupla  (col_ini, lin_ini, col_fin, lin_fin : Word);

implementation      {inicia a unit propriamente dita, é encerrada com END.}
  uses crt;

  procedure Writeln (lin, col: Byte; cor: Byte; texto: String);
  begin
    GotoXY (col,lin);
    TextColor (cor);
    Write (texto);
  end;

  procedure Janela (col_ini, lin_ini, col_fin, lin_fin, fundo: Byte);
  begin
    Window (col_ini,lin_ini,col_fin,lin_fin);
    TextBackground (fundo);
    ClrScr;
  end;

  Procedure Janelapadiao;
  Begin
    Window (1,1,80,25);
```

```
End;

Procedure CenterX (lin : Byte; texto : String);
Var  coluna : Integer;
Begin
  coluna := 40 - length(texto) Div 2;
  GotoXY (coluna, lin); Write (texto)
End;

Procedure LINHASIMPLES (lin, col_ini, col_fin : Byte);
Var  i : Byte;
Begin
  For i:= col_ini To col_fin Do
    Begin
      GotoXY (i,lin); Write (#196);
    End;
  End;

Procedure LINHADUPLA (lin, col_ini, col_fin : Byte);
Var  i : Byte;
Begin
  For i:= col_ini To col_fin Do
    Begin
      GotoXY (i,lin); Write (#205);
    End;
  End;

Procedure MOLDURASIMPLES (col_ini, lin_ini, col_fin, lin_fin : Word);
Var i: Word;
Begin
  gotoxy(col_ini,lin_ini); write(#218);
  gotoxy(col_fin, lin_ini); write(#191);
  gotoxy(col_ini, lin_fin); write(#192);
  gotoxy(col_fin, lin_fin); write(#217);
  for i:=col_ini+1 to col_fin-1 do
    begin
      gotoxy(i, lin_ini); write(#196);
      gotoxy(i, lin_fin); write(#196);
    end;
  for i:=lin_ini+1 to lin_fin-1 do
    begin
      gotoxy(col_ini, i); write(#179);
      gotoxy(col_fin, i); write(#179);
    end;
  End;

Procedure MOLDURADUPLA (col_ini, lin_ini, col_fin, lin_fin : Word);
Var i: Word;
Begin
  gotoxy(col_ini,lin_ini); write(#201);
  gotoxy(col_fin, lin_ini); write(#187);
  gotoxy(col_ini, lin_fin); write(#200);
  gotoxy(col_fin, lin_fin); write(#188);
  for i:=col_ini+1 to col_fin-1 do
    begin
      gotoxy(i, lin_ini); write(#205);
      gotoxy(i, lin_fin); write(#205);
    end;
  for i:=lin_ini+1 to lin_fin-1 do
    begin
      gotoxy(col_ini, i); write(#186);
      gotoxy(col_fin, i); write(#186);
```

```

end;
End;
End.

```

### Listagem do programa 37

## Utilizando a Unit personalizada My\_Unit

Para utilizarmos a unit personalizada My\_Unit, basta incorporá-la ao programa desejado, da mesma forma que incorporamos a unit CRT. Para exemplificar sua aplicação, vejamos o exemplo a seguir e observe como ficou fácil desenhar linhas, molduras, janelas e centralizar textos em nossos programas.

```

{ Nome.....: UNIDADES.PAS
  Função....: Exemplificar aplicações práticas da minha unidade personalizada
  Data.....: 25/02/98
  Autor.....: Francisco Carlos Mancin
}

Program Unidades;
Uses Crt, MyUnit;

Var
  fundo, lin, col, cor, col_ini, lin_ini, col_fin, lin_fin : Byte;
  texto                                                    : string;

Begin
  TextColor (White); TextBackground (Black); ClrScr;

  {Aplicação de WRITEXY}
  Writeln ('Aplicação de WRITEXY'); Writeln; Writeln;
  Write ('Digite um texto qualquer.....: '); Readln (texto);
  Write ('Digite a cor desejada para o texto..: '); Readln (cor);
  Write ('Informe a coluna desejada para v^-lo: '); Readln (col);
  Write ('Informe a linha desejada.....: '); Readln (lin);
  WRITEXY (lin, col, cor, texto);
  Readln; TextColor (White); TextBackground (Black); ClrScr;

  {Aplicação de CENTERX}
  Writeln ('Aplicação de CENTERX'); Writeln; Writeln;
  Write ('Digite o a centralizar: '); Readln (texto);
  Write ('Em que linha deseja que o texto apareça ? '); Readln (lin);
  CENTERX (lin, texto);
  Readln; ClrScr;

  {Aplicação de JANELA}
  CENTERX (2, 'Aplicação de JANELA'); Writeln; Writeln;
  Write ('Digite a coluna inicial.: '); Readln (col_ini);
  Write ('Digite a coluna final...: '); Readln (col_fin);
  Write ('Digite a linha inicial...: '); Readln (lin_ini);
  Write ('Digite a linha final....: '); Readln (lin_fin);
  Write ('Informe a cor de fundo..: '); Readln (fundo);
  JANELA (col_ini, lin_ini, col_fin, lin_fin, fundo);
  JANELAPADRAO;
  Readln; TextColor (White); TextBackground (Black); ClrScr;

  {Aplicação de LINHASIMPLES}
  CENTERX (2, 'Aplicação de LINHASIMPLES'); Writeln; Writeln;
  Write ('Digite a coluna inicial.: '); Readln (col_ini);
  Write ('Digite a coluna final...: '); Readln (col_fin);
  Write ('Digite a linha desejada.: '); Readln (lin);
  LINHASIMPLES(lin, col_ini, col_fin);
  Readln; ClrScr;

  {Aplicação de LINHADUPLA}
  CENTERX (2, 'Aplicação de LINHADUPLA'); Writeln; Writeln;
  Write ('Digite a coluna inicial.: '); Readln (col_ini);
  Write ('Digite a coluna final...: '); Readln (col_fin);
  Write ('Digite a linha desejada.: '); Readln (lin);

```

```
LINHADUPLA(lin, col_ini, col_fin);
Readln; ClrScr;

{Aplicação de MOLDURASIMPLES}
CENTERX (2, 'Aplicação de MOLDURASIMPLES'); Writeln; Writeln;
Write ('Digite a coluna inicial.: '); Readln (col_ini);
Write ('Digite a coluna final...: '); Readln (col_fin);
Write ('Digite a linha inicial..: '); Readln (lin_ini);
Write ('Digite a linha final....: '); Readln (lin_fin);
MOLDURASIMPLES (col_ini, lin_ini, col_fin, lin_fin);
Readln; ClrScr;

{Aplicação de MOLDURADUPLA}
CENTERX (2, 'Aplicação de MOLDURADUPLA'); Writeln; Writeln;
Write ('Digite a coluna inicial.: '); Readln (col_ini);
Write ('Digite a coluna final...: '); Readln (col_fin);
Write ('Digite a linha inicial..: '); Readln (lin_ini);
Write ('Digite a linha final....: '); Readln (lin_fin);
MOLDURADUPLA (col_ini, lin_ini, col_fin, lin_fin);
Readln; ClrScr;
```

End.

### Listagem do programa 38

## Exercícios de fixação

1. Escreva um programa em linguagem Pascal que possua uma função que desenvolva o cálculo de  $X$  elevado a  $n$ . Após a conclusão e teste do programa, edite a My\_Unit e incremente nela mais esta função.
2. Escreva um programa em linguagem Pascal que possua uma função que converta uma string qualquer em maiúscula e a devolva ao programa principal. Após a conclusão e teste do programa, edite a My\_Unit e incremente nela mais esta função.
3. Escreva um programa em linguagem Pascal que possua uma função que converta uma string qualquer em **minúscula** e a devolva ao programa principal. Após a conclusão e teste do programa, edite a My\_Unit e incremente nela mais esta função.
4. Escreva um programa em linguagem Pascal que possua uma função que calcule o Fatorial de um número qualquer. Após a conclusão e teste do programa, edite a My\_Unit e incremente nela mais esta função.



## 16. Trabalhando com interrupções

Uma das grandes vantagens da linguagem Pascal é possibilitar o controle do hardware da máquina através da utilização de interrupções de hardware. Existe uma quantidade bastante grande de interrupções disponíveis para o controle do hardware e, cada uma delas é destinada a uma aplicação específica. Como exemplo, utilizaremos interrupções de hardware para obter a data do sistema e para controlar o mouse na tela do computador.

Muitas outras aplicações poderão ser testadas: resete do sistema, desaparecimento do cursor, formatação de disco, obtenção do espaço livre em disco e em memória e outros. Não exploraremos estas interrupções por não se tratar de uma necessidade específica neste 1º ano.

Maiores informações poderão ser obtidas em livros específicos sobre Interrupções ROM-BIOS, ou até mesmo na Internet.

### Exercícios Resolvidos

1. Escreva um programa em linguagem Pascal que disponibilize na tela a hora e a data do sistema.

```
{ Nome.....: DataHora.PAS
  Função.....: Exemplifica a visualização da data e hora do sistema
  Data.....: 25/01/00
  Autor.....: Francisco Carlos Mancin
}
PROGRAM DataHora;
USES Dos,crt;

CONST
  dias : array [0..6] of String[13] = ('Domingo','Segunda-feira','Terça-feira',
                                       'Quarta-feira','Quinta-feira','Sexta-feira',
                                       'Sábado');

VAR   regs           : registers;
      h, m, s, hund, col, lin : word;
      ano, mes, dia, dow   : Word;
      op              : char;

procedure Mouse;
begin
  regs.ax:=1;           {0 desliga o mouse}
  intr($33,regs);
end;

function AjustaZero(w : Word) : String;
var
  s : String;
begin
  Str(w:0,s);
  if Length(s) = 1 then s := '0' + s;
  AjustaZero := s;
end;

begin
  clrscr;
  mouse;
  GetTime(h,m,s,hund);
  GotoXY(25,5);
  Writeln('A hora do sistema é: ', AjustaZero(h),':', AjustaZero(m),':', AjustaZero(s));
  GetDate(ano,mes,dia,dow);
  GotoXY(25,6); Write('Hoje ', dias[dow],', ', dia:0, '/', mes:0, '/', ano:0);
  Gotoxy(20,15); Write('Movimente o mouse e <click> em seu botão');
  repeat
    regs.ax:=6;   regs.bx:=0;   intr($33,regs);
    if regs.bx <> 0 then
      begin
        op := 'S';
        Gotoxy(20,15); Write('Pressione qualquer tecla para encerrar...');
      end;
  until (OP='N') OR (OP='S');
  readkey;
end.
```

2. Escreva um programa em linguagem Pascal que exemplifique na prática a utilização do mouse como controle de eventos na tela.

```
{ Nome.....: Mouse.PAS
  Função.....: Exemplifica a utilização do mouse e data do sistema
  Data.....: 25/01/00
  Autor.....: Francisco Carlos Mancin
}

program mouse;
uses crt, DOS;

CONST
  dias : array [0..6] of String[13] = ('Domingo', 'Segunda-feira', 'Terça-feira',
                                       'Quarta-feira', 'Quinta-feira', 'Sexta-feira',
                                       'Sábado');

var
  a      : Real;
  CALC   : EXTENDED;
  op     : char;
  e      : INTEGER;
  cont   : byte;
  REGS   : REGISTERS;
  AA,D,M, Col,Lin, dow : word;

function expo(num:real;expoente:INTEGER):EXTENDED;
var
  c : EXTENDED;
  i : BYTE;
begin
  c:=1;
  for i := 1 to expoente do
  begin
    c:=c * num;
    expo:=c;
  end;
end;

function center (mensagem:string):string;
var
  tamanho : integer;
begin
  tamanho:= 38 + length(mensagem) div 2;
  write(mensagem:tamanho);
end;

Begin
  REGS.AX:=1;
  INTR($33,REGS);
  TEXTBACKGROUND(7);CLRSCR;

  WINDOW(3,3,79,24);
  TEXTBACKGROUND(0);CLRSCR;
  textcolor(8);
  gotoxy(1,22);center('Função Exponencial');

  WINDOW(2,2,78,23);
  TEXTBACKGROUND(1);CLRSCR;
  textcolor(14);

  op:='S';
  while op = 'S' do
  begin
    clrscr;
    for cont := 1 to 77 do
    begin
      gotoxy(cont,1);write('=');
      gotoxy(cont,21);write('=');
    end;

    for cont := 1 to 21 do
    begin
      gotoxy(1,cont);write('||'); gotoxy(77,cont);write('||');
```

```

end;
gotoxy(1,1); write('F'); gotoxy(1,21); write('L');
gotoxy(77,1); write('J'); gotoxy(77,21);write('J');

GOTOXY(3,2);WRITE(' C:\ : ',DISKFREE(3) DIV 1024,' KB LIVRES');

GETDATE(AA,M,D,dow); TEXTCOLOR(LIGHTGREEN);
GOTOXY(59,2); WRITE('Data: ',D,'/',M,'/',AA);
GOTOXY(62,3); WRITE('<',dias[dow], '>');

gotoxy(10,4);write('Digite um nº: ');readln(a);
gotoxy(10,6);write('Entre c/ um expoente p/ esse nº: '); readln(e);

TEXTCOLOR(LIGHTGREEN);
GETDATE(AA,M,D,dow); GOTOXY(59,2); WRITE('DATA: ',D,'/',M,'/',AA);

IF E = 0
Then
begin
gotoxy(15,10); Write('O resultado de ',A:1:0,' elevado a 0 é: 1');
end
Else
Begin
CALC:=expo(a,e); gotoxy(15,10);
Write('O resultado de ',A:1:0,' elevado a ',E,' é: ');
end;

TextColor(white); WRITE(CALC:1:0); Textcolor(Yellow);
gotoxy(30,16); write('DESEJA CONTINUAR???');
gotoxy(30,17); write('');
gotoxy(30,18); write('');
gotoxy(30,19); write('Sim Não');
gotoxy(30,20); write('');
textcolor(lightgreen+BLINK);
GOTOXY(34,19); write(''); gotoxy(45,19); write(''); {2 x ALT + 219}
textcolor(14);

col:=1;
repeat
regs.ax:=6;
regs.bx:=0;
intr($33,regs);
if regs.bx <> 0 then
begin
col:=(regs.cx div 8)+1;
lin:=(regs.dx div 8)+1;
end;
OP:='A';
IF ((COL= 35) OR (COL=36)) AND (LIN=20)
THEN OP:='S';
IF ((COL= 46) AND (LIN=20)) OR ((COL= 47) AND (LIN=20))
THEN OP:='N';
until (OP='N') OR (OP='S');
end;
op:='a';
TEXTBACKGROUND(Black); ClrScr;
end.

```

### Listagem do programa 40

## 17. Definindo Constantes e Tipos Definidos pelo Usuário

Constantes são expressões que têm valores inalterados durante o processo do programa, sendo seu tipo definido por seus conteúdos. A vantagem na utilização de constantes é que estas são incorporadas no início de um programa e torna-se fácil fazer a manutenção do mesmo, porque basta alterar um único local e todo o programa estará atualizado. Sintaxe:

```
CONST
kbyte = 1024;
mbyte = kbyte * kbyte;
mens1 = 'Eu estou adorando a linguagem PASCAL!';
estqmin = 500;
estqmax = 10000;
```

Os tipos que podem ser definidos pelo usuário também apresentam diversas vantagens, pois possibilitam limitar determinados componentes que integrarão um grupo. Para defini-los, basta utilizarmos a declaração TYPE, como segue:

```
TYPE
estacao = ('Verão', 'Outono', 'Inverno', 'Primavera');
naipe = ('Ouro', 'Espadas', 'Copas', 'Paus');
dia_util = ('Segunda', 'Terça', 'Quarta', 'Quinta', 'Sexta');
fim_de_semana = ('Sábado', 'Domingo');
cor_basica = ('Azul', 'Vermelha', 'Verde');
nota = 0..10;
letra = 'A'..'Z';
```

### Exercícios resolvidos

1. Elabore um programa em linguagem Pascal que após ler uma palavra de  $n$  letras, mostre-a invertida na tela.

```
{ Nome do programa .....: INVERTE.PAS
Função .....: Lê uma palavra qualquer e exibe sua inversa
Ambiente Operacional ....: DOS
Linguagem .....: PASCAL v. 7.0
Data de criação .....: 10/02/96
Data da última alteração   : 2/4/2003
Escrito por .....: Francisco Carlos Mancin   }

PROGRAM inverte;
USES CRT, My_UNIT;

CONST
    comprimento = 20;

VAR
    palavra : STRING[comprimento];
    indice  : BYTE;

BEGIN
    CLRSCR;
    MOLDURADUPLA (1,1,80,24);
    CENTERX (10, 'Digite a palavra que será invertida: ');
    READLN(palavra);
    GOTOXY(35,15);
    FOR indice:= ORD(palavra[0]) DOWNT0 1 DO
        WRITE(palavra[indice]);
    ReadKey;
END.
```

#### Listagem do programa 41

2. Escreva um programa em linguagem Pascal, utilizando matriz, que informe por extenso o mês correspondente ao número digitado pelo usuário. Incorpore também o teste de validação de dados.

```
{ Nome do programa.....: MES.PAS
Função.....: Informa por extenso o correspondente ao mês numérico
Data de criação.....: 10/02/96
Data da última alteração..: 2/4/2003
```

```
Escreito por..... : Francisco Carlos Mancin  }  
  
PROGRAM Mes;  
  
USES CRT;  
CONST  
    meses : ARRAY[1..12] OF STRING[9] = ('Janeiro', 'Fevereiro', 'Março', 'Abril',  
                                           'Maio', 'Junho', 'Julho', 'Agosto',  
                                           'Setembro', 'Outubro', 'Novembro',  
                                           'Dezembro');  
  
VAR  
    mes : BYTE;  
  
BEGIN  
    CLRSCR; GOTOXY(10,5);  
    WRITE('Entre com um mês entre 1 e 12: ');  
    REPEAT  
        GOTOXY(41,5); CLREOL;  
        GOTOXY(41,5); READLN(mes);  
    UNTIL mes IN [1.. 12];  
    GOTOXY(20,10); WRITE('O mês número ', mes:2, ' é o: ',meses[mes]);  
    READKEY;  
END.
```

### Listagem do programa 42

## Exercícios de fixação

1. Escreva um programa em linguagem Pascal que após ler 30 números quaisquer fornecidos pelo usuário, determine quantos são os pares e quantos são os ímpares. Desenhe uma matriz de caracteres gráficos na tela e utilize técnica tabular para seu preenchimento.
2. Rescreva o programa anterior, incorporando mais duas matrizes gráficas de modo que uma apresente todos os elementos pares e outra apresente todos os elementos ímpares da matriz original.
3. Escreva um programa em linguagem Pascal que crie uma matriz com 20 números aleatórios (de 0 a 100) e os classifique em ordem crescente.
4. Próximo a um grande evento esportivo, você foi contratado para desenvolver um programa em linguagem Pascal que seja capaz de calcular a idade e o peso médio de todos os jogadores de um time individualmente e a idade e o peso médio de todos os jogadores participantes. Sabe-se que cada time tem 23 jogadores e que 40 times estão participando. Tem-se também o nome de cada um dos jogadores. Disponibilize ainda a oportunidade de continuar a execução ou não do programa.

## 18. Trabalhando com o Modo Gráfico

O ambiente gráfico do Turbo Pascal é capaz de transformá-lo em um ambiente de programação bastante interessante e fácil de programar.

Podemos desenvolver qualquer programa de forma gráfica, e, assim torná-lo mais interessante e amigável. No ambiente gráfico dispomos de até 640 x 480 na tela para utilizar independente da aplicação desejada. Nele podemos também efetuar a aplicação do mouse em nossos programas, tarefa esta que será estudada posteriormente.

### Entendendo a tela no modo gráfico



### Funções gráficas disponíveis no Turbo Pascal

Arc	GetFillSettings	ImageSize	SetAspectRatio
*Bar	GetGraphMode	*InitGraph	*SetBkColor
*Bar3D	GetImage	InstallUserDriver	*SetColor
*Circle	GetLineSettings	InstallUserFont	*SetFillPattern
*ClearDevice	GetMaxColor	*Line LineRel	*SetFillStyle
ClearViewPort	GetMaxMode	LineTo	SetGraphBufSize
*CloseGraph	*GetMaxX	MoveRel	SetGraphMode
DetectGraph	*GetMaxY	*MoveTo	*SetLineStyle
Drawpoly	GetModeName	*OutText	SetPalette
Ellipse	GetModeRange	*OutTextXY	SetRGBPalette
FillEllipse	GetPalette	PieSlice	SetTextJustify
FillPoly	GetPaletteSize	PutImage	*SetTextStyle
FloodFill	GetPixel	*PutPixel	SetUserCharSize
GetArcCoords	GetTextSettings	*Rectangle	SetViewPort
GetAspectRatio	GetViewSettings	RegisterBGldriver	SetVisualPage
GetBkColor	GetX	RegisterBGIfont	SetWriteMode
GetColor	GetY	RestoreCrtMode	TextHeight
GetDefaultPalette	GraphDefaults	Sector	TextWidth
GetDriverName	GraphErrorMsg	SetActivePage	
GetFillPattern	GraphResult	SetAllPalette	

### Resumo das funções gráficas mais utilizadas

1) *InitGraph (nGrDriver: Integer, nGrMode:Integer, cGrPath:String);*

Descrição: Inicializa o trabalho em modo gráfico.

nGrDriver pode ser:

Constante	Valor/comentário	Constante	Valor/comentário
CurrentDriver	-128/Para GetModeRange	EGAMono	5
<b>Detect</b>	<b>0/Requer autodeteção</b>	IBM8514	6
CGA	1	HercMono	7
MCGA	2	ATT400	8
EGA	3	VGA	9
EGA64	4	PC3270	10

**nGrMode** retorna um valor que indica o modo de resolução atual.

**cGrPath** é uma string que indica o caminho (path) de procura dos drivers de vídeo.

## 2) CloseGraph;

**Descrição:** Encerra o trabalho em modo gráfico.

## 3) ClearDevice;

**Descrição:** Limpa a tela gráfica e posiciona o ponteiro do cursor na coordenada 1,1. É equivalente a procedure ClrScr do modo alfa.

## 4) GetMaxX e GetMaxY

**Descrição:** São funções que retornam os números máximos de pontos em X e em Y, respectivamente, do modo gráfico atual em uso.

## 5) SetTextJustify (nHorizontal: Integer, nVertical: Integer);

**Descrição:** Parametriza os valores de justificação do texto usado pelos comandos OutText e OutTextXY.

nHorizontal pode assumir:		nVertical pode assumir:	
Constante	Valor	Constante	Valor
LeftText	0	BottomText	0
CenterText	1	CenterText	1
RightText	2	TopText	2

## 6) SetTextStyle (nFont, nDirection, nSize:Integer);

**Descrição:** Define o estilo de texto que será escrito.

**nFont** pode assumir os seguintes valores:

Constante	Valor	Significado
DefaultFont	0	8x8 bit mapped font
TriplexFont	1	Stroked font
SmallFont	2	Stroked font
SansSerifFont	3	Stroked font
GothicFont	4	Stroked font

**nDirection** pode assumir os seguintes valores:

Constante	Valor	Significado
HorizDir	0	Orient left to right
VertDir	1	Orient bottom to top

nSize pode assumir os seguintes valores:

Constante	Valor	Significado
UserCharSize	0	Tamanho definido pelo usuário.
	> 0	Tamanho escalonado.

**7) MoveTo (nX:Integer, nY:Integer);**

**Descrição:** Posiciona o ponteiro do cursor em uma coordenada estabelecida pelos parâmetros X e Y.

**8) OutText (cText:String)**

**Descrição:** Coloca a cadeia de caracteres cText em uma posição do vídeo que corresponde a coordenada atual do ponteiro do cursor.

**9) OutTextXY (nX:Integer, nY:Integer, cText:String);**

**Descrição:** Coloca a cadeia de caracteres cText em uma posição do vídeo estabelecida por nX e nY.

**10) SetBkColor (nColor:Word);**

**Descrição:** Seleciona a cor de fundo usando a paleta.

**11) SetColor(nColor: Word);**

**Descrição:** Seleciona a cor de frente usando a paleta.

Cores Escuras: (Foreground & Background)		Cores Claras: (Foreground)	
Constante	Valor	Constante	Valor
Black	0	DarkGray	8
Blue	1	DarkBlue	9
Green	2	LightGreen	10
Cyan	3	LightCyan	11
Red	4	LightRed	12
Magenta	5	LightMagenta	13
Brown	6	Yellow	14
LightGray	7	White	15

Obs.: Para frente piscante, Blink = 128

**12) PutPixel (nX:Integer, nY:Integer, nColor:Word);**

**Descrição:** Coloca um ponto na tela na posição especificada por nX e n Y com a cor estabelecida por nColor.

**13) Line (nX1:Integer, n Y1:Integer, nX2:Integer, n Y2:Integer);**

**Descrição:** Traça uma linha na tela ligando dois pontos especificados por [nX1, nY1] e [nX2, nY2].

**14) Rectangle (nX1:Integer, nY1:Integer, nX2:Integer, nY2:Integer);**

**Descrição:** Desenha um retângulo usando as definições atuais de linha e estilo nas coordenadas indicadas pelos pontos [nX1, nY1] e [nX2, nY2].

**15) Circle(nX:Integer; n Y:Integer; nR:Word);**

**Descrição:** Desenha um círculo tendo como centro as coordenadas nX e nY. O tamanho é determinado pelo valor de nR, ou seja, o valor do raio do círculo.



**16) SetLineStyle (nLineStyle:Integer, nPattern:Integer, nThickness:Integer);**

**Descrição:** Define um estilo de linha para ser usado em Line(), Rectangle(), etc.

**nLineStyle** é o estilo da linha e pode ser:

SolidLn	0	(solida)
DottedLn	1	(pontilhada)
CenterLn	2	(tracejada 1)
DashedLn	3	(tracejada 2)
UserBitLn	4	(definida pelo usuário)

**nPattern** é um padrão de bits para a linha e só tem validade quando o valor de nLineStyle é igual a UserBitLn, ou seja, 4.

**nThickness** é a largura de linha e pode ser:

NormWidth	1	(normal)
ThickWidth	3	(grossa)

**17) SetFillStyle (nPattern : Word, nColor : Word);**

**Descrição:** Seleciona um padrão de preenchimento e cor para as funções Fillpoly(), Bar(), Bar3D() e PieSlice().

**nPattern** pode ser:

Constante	Valor	Significado
EmptyFill	0	Uses background color
SolidFill	1	Uses draw color
LineFill	2	--- fill
LtSlashFill	3	/// fill
SlashFill	4	III thick fill
BkSlashFill	5	\ thick fill
LtBkSlashFill	6	\ fill
HatchFill	7	Light hatch fill
XHatchFill	8	Heavy cross hatch
InterleaveFill	9	Interleaving Line
WideDotFill	10	Widely spaced dot
CloseDotFill	11	Closely spaced dot
UserFiil	12	User-defined fill

**18) SetFillPattern (nPattern:FillPattern Type, n Color: Word);**

**Descrição:** Define um padrão de preenchimento e cor para as funções Fillpoly(), Bar(), Bar3D() e PieSlice().

**19) Bar (nX1:Integer, nY1:Integer, nX2:Integer, nY2:Integer);**

**Descrição:** Desenha uma barra na tela nas coordenadas indicadas pelos pontos [nX1, nY1] e [nX2, n Y2].

**20) Bar3D (nX1, nY1, nX2, nY2:Integer, nDepth:Word, ITop:Boolean);**

**Descrição:** Desenha uma barra tridimensional na tela nas coordenadas indicadas pelos pontos [nX1, nY1] e [nX2, nY2]. A profundidade da projeção é definida pelo nDepth. O parâmetro ITop define se a projeção do topo será mostrada.

**Exercícios resolvidos**

Para exemplificar a aplicação dos recursos gráficos na programação em Pascal estaremos vendo alguns exemplos práticos de programas que exploram parte dos recursos gráficos disponíveis.

1. Escreva um programa gráfico, em linguagem Pascal, que escreva mensagens em volta da tela.

```

{Arquivo.....: Text_01.pas
 Função.....: Exemplificar a escrita de textos no ambiente gráfico
 Autor.....: Francisco Carlos Mancin
 Data.....: 2/4/2003
}
program Text_01;
uses Crt, Graph;
var
  grDriver, grMode, ErrCode: Integer;
begin
  clrscr;
  write('Modo Alfa');
  readkey;

  grDriver := Detect;
  InitGraph(grDriver, grMode, 'R:\ling\tp7\bgi');
  ErrCode := GraphResult;

                                     {superior/esquerdo}
  setcolor (red);
  settextstyle (triplexfont, horizdir,5);
  settextjustify (lefttext, toptext);
  outtextxy (0,0, 'Escrita');
  readkey;

                                     {superior/centro}
  setcolor (blue);
  settextjustify (centertext, toptext);
  outtextxy (320,0, 'Escrita');
  readkey;

                                     {superior/direito}
  setcolor (green);
  settextjustify (righttext, toptext);
  outtextxy (639,0, 'Escrita');
  readkey;

                                     {centro/esquerdo}
  setcolor (yellow);
  settextjustify (lefttext, centertext);
  outtextxy (0,239, 'Escrita');
  readkey;

                                     {centro/centro}
  setcolor (magenta);
  settextjustify (centertext, centertext);
  outtextxy (320,240, 'Escrita');
  readkey;

                                     {centro/direito}
  setcolor (lightgreen);
  settextjustify (righttext, centertext);
  outtextxy (639,240, 'Escrita');
  readkey;

                                     {inferior/esquerdo}
  setcolor (lightgray);
  settextjustify (lefttext, bottomtext);
  outtextxy (0,479, 'Escrita');
  readkey;

                                     {inferior/centro}
  setcolor (lightred);
  settextjustify (centertext, bottomtext);
  outtextxy (320,479, 'Escrita');
  readkey;

                                     {inferior/direito}
  setcolor (lightcyan);
  settextjustify (righttext, bottomtext);
  outtextxy (639,479, 'Escrita');
  readkey;

                                     {centro/centro/vertical}
  setcolor (magenta);
  settextjustify (centertext, centertext);
  outtextxy (320,240, 'Escrita');
  readkey;
  CloseGraph;
end.

```

2. Escreva um programa gráfico, em linguagem Pascal, que escreva mensagens em vários ângulos em volta da tela.

```
{ Nome do Programa.....: Text_02.pas
  Função.....: Inicialização do Modo Gráfico
  Autor.....: Francisco Carlos Mancin
  Data.....: 9/8/97
}

Program Text_02;
Uses Crt, Graph;
Var
  GrDriver, Grmode : Integer;
  GrPath : String;
Begin
  ClrScr;
  Gotoxy(32,12); Write('Estou no modo Alfa...');
  Textcolor (Yellow + Blink);
  Gotoxy(25,18); Write('Pressione qq. tecla e veja o modo gráfico.');
```

Readkey;

GrPath := 'R:\LING\TP7\BGI';  
GrDriver := Detect;  
GrMode := Detect;  
InitGraph(GrDriver, Grmode, GrPath);

SetColor(Green);  
SetTextJustify(CenterText,CenterText);  
OutTextxy(Getmaxx div 2,Getmaxy div 2,'Estou no Modo Gráfico');

SetColor(Yellow);  
SetTextStyle(3,Horizdir,3);  
SetTextJustify(LeftText,TopText);  
OutTextxy(1,1,'Modo Gráfico');

SetColor(White);  
SetTextStyle(5,Horizdir,3);  
SetTextJustify(RightText,TopText);  
OutTextxy(Getmaxx,1,'Modo Gráfico');

SetColor(Blue);  
SetTextStyle(7,Horizdir,6);  
SetTextJustify(LeftText,BottomText);  
OutTextxy(1,Getmaxy,'Modo Gráfico');

SetColor(Red);  
SetTextStyle(9,Horizdir,3);  
SetTextJustify(RightText,BottomText);  
OutTextxy(Getmaxx,Getmaxy,'Modo Gráfico');

SetColor(LightGray);  
SetTextStyle(10,Vertdir,3);  
SetTextJustify(CenterText,BottomText);  
OutTextxy(Getmaxx Div 2,Getmaxy,'Modo Gráfico');

SetColor(LightMagenta);  
SetTextStyle(12,Vertdir,5);  
SetTextJustify(CenterText,topText);  
OutTextxy(Getmaxx Div 2,1,'Modo Gráfico');

SetColor(Magenta);  
SetTextStyle(13,Vertdir,9);  
SetTextJustify(CenterText,CenterText);  
OutTextxy(Getmaxx Div 2,Getmaxy div 2,'Modo Gráfico');

SetColor(Cyan);  
SetTextStyle(6,Horizdir,5);  
SetTextJustify(LeftText, CenterText);  
OutTextxy(1,Getmaxy div 2,'Modo Gráfico');

```

SetColor(lightBlue);
SetTextStyle(4, Horizdir, 5);
SetTextJustify(RightText, CenterText);
OutTextxy(Getmaxx, Getmaxy div 2, 'Modo Gráfico');

Readkey;
CloseGraph;
End.

```

**Listagem do Programa 44**

3. Escreva um programa gráfico, em linguagem Pascal, que exemplifique a utilização de linhas gráficas.

```

{ Nome.....: Lines_1.PAS
  Função...: Desenhando linhas diagonais coloridas
  Autor....: Francisco Carlos Mancin
  Data.....: 2/1/98
}

PROGRAM Lines_1;
USES CRT, GRAPH;
VAR nGrDriver, nGrMode, X, Y, F: INTEGER;
    cGrPath      : STRING;

BEGIN
  cGrPath := 'R:\LING\TP7\BGI';
  nGrDriver := DETECT;
  nGrMode := 3;
  InitGraph (nGrDriver, nGrMode, cGrPath);
  X:= GetMaxX;
  Y:= GetMaxY;
  FOR F:=0 TO GetMaxX DO
    BEGIN
      SETCOLOR (F);
      LINE (0, 480, F, 0);
    END;
  FOR F:=GetMaxX DOWNT0 0 DO
    BEGIN
      SETCOLOR (F);
      LINE (640, 0, F, 480);
    END;
  ReadKey;
  CloseGraph;
END.

```

**Listagem do Programa 45**

4. Escreva um programa gráfico, em linguagem Pascal, que exemplifique a criação de barras gráficas na tela.

```

{ Nome.....: Bar_3D.PAS
  Função...: Mostrar valores num, ricos em barras 3D verticais
  Autor....: Francisco Carlos Mancin
  Data.....: 1/3/98
}

Program Barras;
Uses Crt, Graph;
Const
  Profundidade = 10;
  CoefY = 44;

Var
  nGrDriver, nGrMode      : Integer;
  M, N1, N2, N3, N4, GN1, GN2, GN3, GN4 : Integer;

Begin
  N1:= 5;  N2:= 2;  N3:= 8;  N4:= 6;           {números para compor gráfico}

```

```

nGrDriver:=detect;
InitGraph(nGrDriver,nGrMode,'R:\LING\TP7\bgi');
SetTextStyle(8,Horizdir,3);
SetColor(14); SetBkColor (9);
Rectangle (0,0,639,479);
OutTextXY (250,05,'Gráfico 3D');
SetColor(1);
GN1 := N1 * CoefY;      GN2 := N2 * CoefY;
GN3 := N3 * CoefY;      GN4 := N4 * CoefY;

Line (120,044,400,044);   Line (120,089,400,089);
Line (120,133,400,133);   Line (120,176,400,176);
Line (120,220,400,220);   Line (120,265,400,265);
Line (120,310,400,310);   Line (120,353,400,353);
Line (120,395,400,395);   Line (120,440,400,440);

BAR3D(120,GN1,170,450,Profundidade,TRUE);
BAR3D(200,GN2,250,450,Profundidade,TRUE);
BAR3D(280,GN3,330,450,Profundidade,TRUE);
BAR3D(360,GN4,410,450,Profundidade,TRUE);
Line (100,450,600,450);
Line (100,40,100,450);

SetTextStyle(2,horizdir,6);
Line (88,44,100,44);   OutTextXY (70,42, '10');
Line (88,89,100,89);   OutTextXY (80,85, '9');
Line (88,133,100,133); OutTextXY (80,129,'8');
Line (88,176,100,176); OutTextXY (80,172,'7');
Line (88,220,100,220); OutTextXY (80,216,'6');
Line (88,265,100,265); OutTextXY (80,261,'5');
Line (88,310,100,310); OutTextXY (80,306,'4');
Line (88,353,100,353); OutTextXY (80,349,'3');
Line (88,395,100,395); OutTextXY (80,391,'2');
Line (88,440,100,440); OutTextXY (80,436,'1');

SetTextStyle (2,HorizDir,5);   OutTextXY (140,460,'1º');
SetTextStyle (2,HorizDir,5);   OutTextXY (220,460,'2º');
SetTextStyle (2,HorizDir,5);   OutTextXY (300,460,'3º');
SetTextStyle (2,HorizDir,5);   OutTextXY (380,460,'4º');

readln;
CloseGraph;
End.

```

### Listagem do Programa 46

5. Escreva um programa gráfico, em linguagem Pascal, que apresente na tela uma textura.

```

{ Nome....: TEXTURA.PAS
  Função...: Criar uma textura em modo gráfico
  Autor....: Francisco Carlos Mancin
  Data.....: 1/3/98
}
Program Textura;
Uses Crt, graph;
Const path = 'R:\ling\tp7\bgi';
Var i, j, drive, mode : Integer;

Begin
  drive := detect;
  InitGraph(Drive, Mode, Path);

  For i := 1 to getmaxx do
  Begin
    For j := 1 to Getmaxy do
    begin

```

```

    PutPixel(i,j, Random(GetmaxColor));
End;
End;

SetFillStyle(1, Black);
Bar(Getmaxx div 2-250,Getmaxy div 2-20, Getmaxx div 2+250, Getmaxy div 2+30);
SetColor(Green);
SetTextStyle(3, horizdir, 5);
SetTextJustify(CenterText,CenterText);
OutTextXy(Getmaxx div 2, Getmaxy div 2,'Francisco Carlos Mancin');
Readkey;
End.

```

#### Listagem do Programa 47

6. Escreva um programa gráfico, em linguagem Pascal, que desenhe a bandeira do Brasil.

```

{Arquivo....: B_Brasil.pas
Função.....: Desenha uma bandeira do Brasil
Autor.....: Francisco Carlos Mancin
Data.....: 27/08/97
}

program B_Brasil;
uses crt, graph;
const path = 'R:\ling\tp7\bgi';
var driver, mode, i : integer;

Begin
  Driver := Detect;
  Mode := Detect;
  Initgraph(Driver, Mode, Path);
  Setcolor(Green);
  Setfillstyle(1,Green);
  Bar(30,30, Getmaxx -30 , Getmaxy -30);
  Setcolor(Yellow);
  For i:= 50 To GetMaxX - 50 Do
  begin
    line(i,getmaxy div 2, getmaxx div 2, 50);
    line(i,getmaxy div 2, getmaxx div 2, getmaxy -50);
  end;
  setcolor(blue);
  setfillstyle(1,blue);
  fillellipse(getmaxx div 2, getmaxy div 2, 85,85);
  readln;
  closegraph;
end.

```

#### Listagem do Programa 48

7. O programa a seguir tem por objetivo explorar a utilização de recursos gráficos de maneira mais ampla.

```

{ Nome.....: DEMO.PAS
Função.....: Demonstrar aplicação em modo gráfico
Autor.....: Francisco Carlos Mancin
Data.....: 25/02/98
}
Program Demo;
Uses Crt, Graph;
Const kGrPathDefauIt = 'R:\ling\tp7\bgi';
Var nGrOriver, nGrMode : Integer;
    CGrPath           : String;

Procedure Title (cTitleText:String);
Begin
  SetTextJustify (CenterText, TopText);
  SetTextStyle (TriplexFont, HorizDir,5);
  OutTextXY (GetMaxX Div 2, 1, cTitleText);

```

```
End;

Procedure Message (cMessage Text: String);
Begin
  SetTextJustify (CenterText, BottomText);
  SetTextStyle (SansSerifFont, HorizDir,4);
  OutTextXY (GetMaxX Div 2, GetMaxY, cMessageText);
End;

Procedure ShowFonts;
Begin
  ClearDevice;  SetTextJustify (CenterText, TopText);
  SetTextStyle (DefaultFont, HorizDir,3);  {0}
  OutTextXY (GetMaxX Div 2, GetMax Y Div 15, 'DefaultFont - 0');

  SetTextStyle (TriplexFont, HorizDir,3);  {1}
  OutTextXY (GetMaxX Div 2, GetMax Y Div 15 * 2, 'TriplexFont - 1');

  SetTextStyle (SmallFont, HorizDir,4);      {2}
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 3, 'SmallFont - 2');

  SetTextStyle (SansSerifFont, HorizDir,3); {3}
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 4, 'SansSerifFont - 3');

  SetTextStyle (GothicFont, HorizDir,3);  {4}
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 5, 'GothicFont - 4');

  SetTextStyle (5, HorizDir,3);
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 6, 'Desconhecido - 5');

  SetTextStyle (6, HorizDir,3);
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 7, 'Desconhecido - 6');

  SetTextStyle (8, HorizDir,2);
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 8, 'Desconhecido - 7');
  SetTextStyle (9, HorizDir,2);
  OutTextXY (GetMaxX Div 2, GetMax Y Div 15 * 9, 'Desconhecido - 9');

  SetTextStyle(10, HorizDir,2);
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 10, 'Desconhecido - 10');

  SetTextStyle (15, HorizDir,2);
  OutTextXY (GetMaxX Div 2, GetMaxY Div 15 * 12, 'Desconhecido - 11 a 15');
  Message('Pressione ENTER para continuar ...'); Readln;
End;

Procedure ShowColors;
Var Frente: Word;
    nColors: Longint;
Begin
  ClearDevice;
  nColors:= GetMaxColor;
  For Frente:= 0 To 30 Do
  Begin
    SetColor(Frente); OutTextXY(Frente * 10 + 10, Frente * 10 + 10, '|');
  End;
  Message ('Pressione ENTER para continuar ...'); Readln;
End;

Procedure DrawPoints;
Begin
  ClearDevice; Randomize;
  Repeat
    PutPixel(Random(GetMaxX), Random(GetMaxY), Random(GetMaxColor));
  until KeyPressed;
  Message('Pressione ENTER para continuar ...'); Readln;
End;
```

```

Procedure DrawLines;
Begin
  ClearDevice;
  Randomize;
  Repeat
    SetColor(Random(GetMaxColor)); SetLineStyle(Random(4),0, Random(3));
    Line(Random(GetMaxX), Random(GetMaxY), Random(GetMaxX), Random(GetMaxY));
  until KeyPressed;
  Message('Pressione ENTER para continuar ...'); Readln;
End;

Procedure DrawRectangles;
Var i: Integer;
Begin
  ClearDevice;
  Randomize;
  Repeat
    For i:= 0 To 3 Do
      Begin
        SetColor(Random(GetMaxColor));
        SetLineStyle(i,0, NormWidth);
        Rectangle(Random(GetMaxX), Random(GetMaxY), Random(GetMaxX),
          Random(GetMaxY));
      End;
    Delay(500); ClearDevice;
  until KeyPressed;
  Message('Pressione ENTER para continuar ...'); ReadKey;
End;

Procedure DrawCircles;
Var i: Integer;
Begin
  ClearDevice; Randomize;
  Repeat
    For i:= 0 to 3 Do
      Begin
        SetColor(Random(GetMaxColor));
        SetLineStyle(i,0, NormWidth);
        Circle(Random(GetMaxX), Random(GetMaxY), Random(GetMaxY DIV 2));
      End;
    Delay(500); ClearDevice;
  until KeyPressed;
  Message('Pressione ENTER para continuar ...');
  Readkey;
End;

Procedure DrawBars;
Var FPT: FillPatternType; {array[1..8] of byte}
Begin
  ClearDevice; Randomize;
  FPT[1]:= $1; FPT[2]:= $3; FPT[3]:= $7; FPT[4]:= $F;
  FPT[5]:= $1F; FPT[6]:= $3F; FPT[7]:= $7F; FPT[8]:= $FF;
  Repeat
    SetFillPattern(FPT,Random(GetMaxColor));
    Bar(Random(GetMaxX),Random(GetMaxY), Random(GetMaxX),Random(GetMaxY));
    Delay(500); ClearDevice;
  until KeyPressed;
  Message('Pressione ENTER para continuar ...'); Readln;
End;

Procedure DrawBars3D;
Var FPT: FillPatternType; { array[1..8] of byte }
    nX1, nX2, n Y1, nY2: Integer;
Begin
  ClearDevice; Randomize;
  FPT[1]:= $1; FPT[2]:= $3; FPT[3]:= $7; FPT[4]:= $F;

```



```

FPT[5]:= $1F; FPT[6]:= $3F; FPT[7]:= $7F; FPT[8]:= $FF;
Repeat
  SetFillPattern(FPT, Random(GetMaxColor));
  nX1:= Random(GetMaxX); nY1:= Random(GetMaxY); nX2:= Random(GetMaxX);
  nY2:= Random(GetMaxY);
  Bar3D(nX1, nY1, nX2, nY2, Random(GetMaxY DIV 10), TopOn);
  Delay(500); ClearDevice;
until KeyPressed;
Message('Pressione ENTER para continuar ...'); Readln;
End;

Procedure ShowPolygons;
const Triangle: array[1..4] of PointType = ((X: 50; Y: 100), (X: 100; Y:100),
                                             (X: 150; Y:150), (X: 50; Y:100));
      Star: array[1..11] of PointType = ((X: 100; Y:50), (X:125; Y:50),
                                           (X: 137; Y:37), (X: 150; Y: 50),
                                           (X: 175; Y:50), (X: 155; Y: 70),
                                           (X: 170; Y:90), (X: 137; Y: 75),
                                           (X: 105; Y:90), (X: 120; Y: 70),
                                           (X: 100; Y: 50));
Begin
  DrawPoly(SizeOf(Triangle) div SizeOf(PointType), Triangle); Readln;      {4}
  FillPoly(SizeOf(Triangle) div SizeOf(PointType), Triangle); Readln;      {4}
  DrawPoly(SizeOf(Star) div SizeOf(PointType), Star); Readln;              {11}
  FillPoly(SizeOf(Star) div SizeOf(PointType), Star); Readln;
End;

{ - - - Inicio do Programa Principal --- }

Begin
  ClrScr;
  Write('Diretorio de Drivers: '); Readln(cGrPath);
  If cGrPath = '?' Then cGrPath:= kGrPathDefault;
  InitGraph(nGrDriver,nGrMode,cGrPath);
  SetBkColor(Blue); SetColor(Yellow); Title('Testes com funções gráficas');
  Message('Pressione ENTER para continuar ...'); Readln;
  SetBkColor(Black);
  ShowFonts;      {Demonstração dos fontes}
  ShowColors;     {Demonstração das cores }
  DrawPoints;     {Demonstração dos pontos}
  DrawLines;      {Demonstração das linhas}
  DrawRectangles; {Demonstração dos retângulos}
  DrawCircles;    {Demonstração dos círculos}
  DrawBars;       {Demonstração de barras}
  DrawBars3D;     {Demonstração de barras 3D}
  ShowPolygons;   {Demonstração de polígonos}
  CloseGraph;
End.

```

## 19. Estruturas de *RECORD*

A declaração RECORD permite-nos definir um registro. Seu conceito e sua sintaxe podem ser observados a seguir:

```
<identificador> = RECORD
    <campo1> : tipo;
    <campo2> : tipo;
    :
    <campon> : tipo;
```

---

**Registro:** é um grupo de informações relativas a uma mesma entidade. Estas informações podem ter características diferentes, como exemplo: RG, nome, endereço, telefone, CEP, código etc...

---

A diferença básica entre uma matriz e um registro é que uma matriz possui elementos de um mesmo tipo e, no caso do registro, podemos ter elementos de diversos tipos. A definição de um registro no Turbo Pascal só pode ser realizada através da área de tipos (TYPE) e, a cada registro definido, estamos na realidade criando um novo tipo. Para sua utilização, basta que o declaremos na área das variáveis do Pascal.

### Exercício resolvido

1. Escreva um programa em linguagem Pascal, utilizando a estrutura de registros, que implemente uma agenda eletrônica, segundo as funções apresentadas a seguir:

```
{ Nome do programa ..... : AGENDA.PAS
  Função ..... : Utiliza records para implementar uma agenda eletrônica
  Ambiente Operacional..... : DOS
  Linguagem ..... : PASCAL v. 7.0
  Data de criação ..... : 10/02/96
  Data da última alteração ..... : 2/4/2003
  Escrito por ..... : Francisco Carlos Mancin }
```

```
PROGRAM agenda;
USES CRT;
TYPE
    registro = RECORD
        nome      : STRING[30];
        ende      : STRING[30];
        cep       : LONGINT;
        fone      : STRING[8];
    END;
CONST
    max = 50;
VAR
    ind      : BYTE;
    tab_reg  : ARRAY [1..max] OF registro;
```

```

FUNCTION s_n : BOOLEAN; {Retorna TRUE se a resposta for S}
VAR
  ch : CHAR;
BEGIN
  REPEAT
    ch := UPCASE(READKEY);
    IF NOT (ch IN ['S', 'N']) THEN WRITE(#7);
  UNTIL ch IN ['S', 'N'];
  s_n := ch = 'S';
END;

PROCEDURE inicio;
BEGIN
  CLRSCR;
  ind := 1;
END;

PROCEDURE inc;
VAR
  x : STRING[6];
  ce : INTEGER;
BEGIN
  REPEAT
    CLRSCR;
    IF ind <= max
      THEN
        BEGIN
          GOTOXY(1,5); WRITE('Incluindo número ',ind:2);
          GOTOXY(1,7); WRITE('Nome.....: ');
          READLN(tab_reg[ind].nome);
          GOTOXY(1,9); WRITE('Endereço.....: ');
          READLN(tab_reg[ind].ende);
          REPEAT
            GOTOXY(1,11); WRITE('C.E.P.....: '); CLREOL; READLN(x);
            VAL(x, tab_reg[ind].cep, ce);
          UNTIL ce = 0;
          GOTOXY(1,13); WRITE('Fone.....: ');
          READLN(tab_reg[ind].fone);
        END
      ELSE
        BEGIN
          GOTOXY(1,5); WRITE('Não pode incluir mais dados. Utilize
arquivos...');
        END;
        ind := ind + 1;
        GOTOXY(25,25); WRITE('Deseja incluir mais algum (S/N) ??? ');
      UNTIL NOT s_n;
    END;
  END;

PROCEDURE classifica;
VAR
  aux : registro;
  ct : BYTE;
BEGIN
  FOR ct := 1 TO ind-2 DO
    BEGIN
      IF tab_reg[ct].nome > tab_reg[ct + 1].nome
        THEN
          BEGIN
            aux := tab_reg[ct];
            tab_reg[ct] := tab_reg[ct + 1];
            tab_reg[ct + 1] := aux;
          END;
        END;
    END;

  END;

PROCEDURE list;
VAR

```

```

ct : BYTE;

BEGIN
  classifica;
  FOR ct := 1 TO ind - 1 DO
    BEGIN
      CLRSCR;
      GOTOXY(10,5);  WRITE('Nome.....: ', tab_reg[ct].nome);
      GOTOXY(10,7);  WRITE('Endereço....: ', tab_reg[ct].ende);
      GOTOXY(10,9);  WRITE('C.E.P.....: ', tab_reg[ct].cep);
      GOTOXY(10,11); WRITE('Fone.....: ', tab_reg[ct].fone);
      GOTOXY(25,25); WRITE('Pressione algo para continuar...');
      REPEAT UNTIL READKEY < > #0;
    END;
  END;

  PROCEDURE fim;
  BEGIN
    CLRSCR;
  END;

  PROCEDURE menu;
  VAR
    fim : BOOLEAN;
    ch   : CHAR;
  BEGIN
    fim := FALSE;
    REPEAT
      CLRSCR;
      GOTOXY(24,2); WRITE('A G E N D A   E L E T R Ô N I C A');
      GOTOXY(30,6); WRITE('I - Inclusão');
      GOTOXY(30,8); WRITE('L - Listagem');
      GOTOXY(30,10); WRITE('F - Fim');
      GOTOXY(40,16); WRITE('Sua opção... ');
      REPEAT
        ch := UPCASE(READKEY);
      UNTIL ch IN ['I', 'L', 'F'];
      CASE ch OF
        'I' : inc;
        'L' : list;
        'F' : fim := TRUE;
      END;
    UNTIL fim;
  END;

  BEGIN
    inicio;
    menu;
    fim;
  END.

```

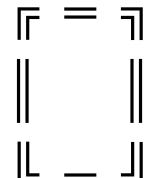
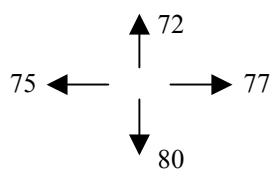
**Listagem do programa 50.**

## Exercício de fixação

1. Escreva um programa em linguagem Pascal que calcule quantos pontos você precisará em cada disciplina no 4º Bimestre. Veja o programa de exemplo com o professor Xico.
2. Escreva um programa em linguagem Pascal que simule uma calculadora que apresente as quatro operações elementares da matemática. Esta calculadora deverá operar em sistema de pilha, ou seja, você vai efetuando a digitação dos dados até o instante em que digitar o valor zero. Nesse instante, o programa deverá retornar ao menu principal e oferecer as opções de cálculos. Pede-se também que a calculadora apresente em tela todos os números nela digitados e em ordem crescente; só os números pares ou só os números ímpares, de acordo com a necessidade do usuário. Disponibilize também a oportunidade de trocar algum(ns) valor(es) em qualquer posição da pilha.

## 20. Tabela ASCII (American National Standard Code for Information Interchange)

32		70	F	108	l	146	Æ	184	©	222	ì
33	!	71	G	109	m	147	ô	185	¶	223	■
34	"	72	H	110	n	148	ö	186		224	Ó
35	#	73	I	111	o	149	ò	187	⌋	225	β
36	\$	74	J	112	p	150	û	188	⌋	226	Ô
37	%	75	K	113	q	151	ù	189	ç	227	Ò
38	&	76	L	114	r	152	ÿ	190	¥	228	ø
39	'	77	M	115	s	153	Ö	191	⌋	229	Õ
40	(	78	N	116	t	154	Ü	192	⌋	230	μ
41	)	79	O	117	u	155	φ	193	⌋	231	þ
42	*	80	P	118	v	156	£	194	⌋	232	Ɔ
43	+	81	Q	119	w	157	Ø	195	⌋	233	Ú
44	,	82	R	120	x	158	x	196	—	234	Û
45	-	83	S	121	y	159	f	197	⌋	235	Ü
46	.	84	T	122	z	160	á	198	ã	236	ý
47	/	85	U	123	{	161	í	199	Ä	237	Ý
48	0	86	V	124		162	ó	200	ℓ	238	—
49	1	87	W	125	}	163	ú	201	ℓ	239	'
50	2	88	X	126	~	164	ñ	202	ℓ	240	-
51	3	89	Y	127	À	165	Ñ	203	ℓ	241	±
52	4	90	Z	128	Ç	166	ä	204	ℓ	242	=
53	5	91	[	129	ü	167	ö	205	=	243	¾
54	6	92	\	130	é	168	ç	206	ℓ	244	¶
55	7	93	]	131	â	169	®	207	α	245	§
56	8	94	^	132	ä	170	¬	208	ð	246	÷
57	9	95	_	133	à	171	½	209	Ð	247	,
58	:	96	`	134	å	172	¼	210	Ê	248	°
59	;	97	a	135	ç	173	ı	211	Ë	249	¨
60	<	98	b	136	ê	174	«	212	È	250	·
61	=	99	c	137	ë	175	»	213	ı	251	¹
62	>	100	d	138	è	176	▒	214	Í	252	³
63	?	101	e	139	ï	177	▒	215	Î	253	²
64	@	102	f	140	î	178	▒	216	İ	254	■
65	A	103	g	141	ì	179		217	⌋	255	
66	B	104	h	142	Ä	180	⌋	218	⌋		
67	C	105	i	143	Å	181	Á	219	■		
68	D	106	j	144	É	182	Â	220	■		
69	E	107	k	145	æ	183	À	221	ı		



## 21. Bibliografia

---

1. RINALDI, Roberto. Turbo Pascal 7.0: Comandos e Funções. - Editora Érica.
2. HERGERT, Douglas. Dominando o Turbo Pascal 5.0. - Editora Ciência Moderna.
3. PASAHOW, Edward J. Turbo Pascal para Eletrônica versão 5.0. - Editora McGraw Hill.
4. SCHMITZ, Antonio Anibal de Souza Teles. Pascal e Técnicas de Programação. - Editora Livros Técnicos e Científicos.
5. Dicionário de informática inglês-português / Sociedade dos Usuários de Computadores e Equipamentos Subsidiários - SUCESU. 4ª edição. - Editora LTC.
6. MANZANO, José Augusto N. G. Programando em Turbo Pascal 7.0. – Editora Érica.
7. BRÁS, Eurico Soalheiro. Interrupções ROM BIOS MS-DOS. – Editora McGraw-Hill.