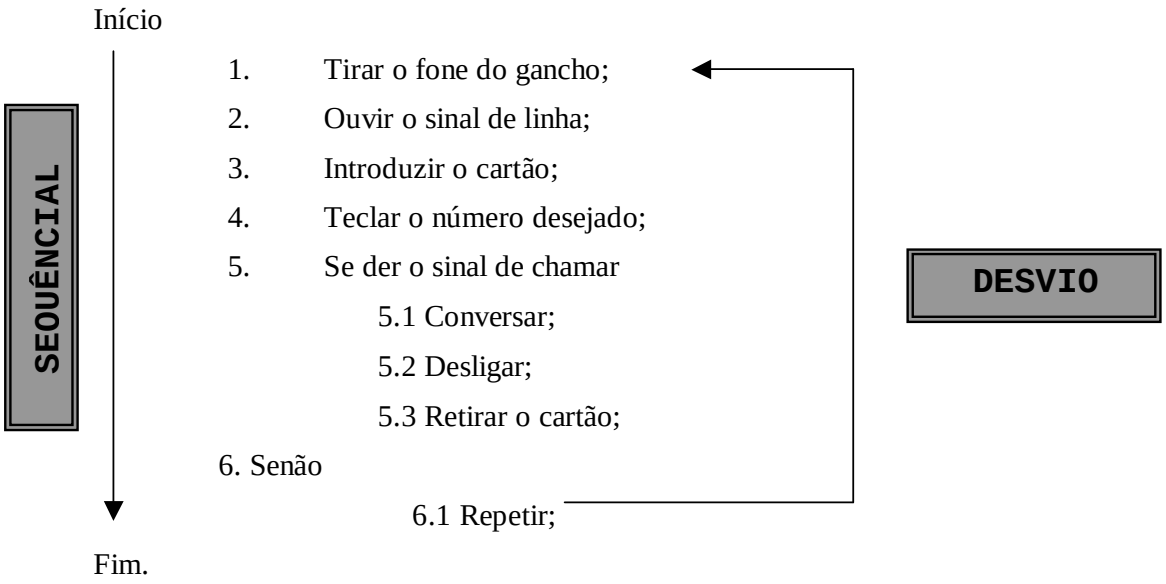
ALGORITMO

Um Algoritmo é uma seqüência de instruções ordenadas de forma lógica para a resolução de uma determinada tarefa ou problema.

ALGORITMO NÃO COMPUTACIONAL

Abaixo é apresentado um Algoritmo não computacional cujo objetivo é usar um telefone público.



PROGRAMA

Um programa é um Algoritmo escrito em uma linguagem computacional.

LINGUAGENS DE PROGRAMAÇÃO

São Softwares que permitem o desenvolvimento de programas. Possuem um poder de criação ilimitado, desde jogos, editores de texto, sistemas empresariais até sistemas operacionais.

Existem várias linguagens de programação, cada uma com suas características próprias. Exemplos:

- Pascal
- Clipper
- C
- Visual Basic
- Delphi e etc.

TÉCNICAS ATUAIS DE PROGRAMAÇÃO

- Programação Seqüencial
- Programação Estruturada
- Programação Orientada a Eventos e Objetos

ALGORITMOS EM "PORTUGOL"

Durante nosso curso iremos aprender a desenvolver nossos Algoritmos em uma pseudo-linguagem conhecida como "Portugol" ou Português Estruturado.

"Portugol" é derivado da aglutinação de Português + Algol. Algol é o nome de uma linguagem de programação estruturada usada no final da década de 50.

OPERADORES ARITMÉTICOS

+	➔	Adição
-	➔	Subtração
*	➔	Multiplicação
/	➔	Divisão

OPERADORES RELACIONAIS

>	➔	Maior que
<	➔	Menor que
>=	➔	Maior ou Igual
<=	➔	Menor ou Igual
=	➔	Igual
<>	➔	Diferente

LINEARIZAÇÃO DE EXPRESSÕES

Para a construção de Algoritmos todas as expressões aritméticas devem ser linearizadas, ou seja, colocadas em linhas.

É importante também ressalvar o uso dos operadores correspondentes da aritmética tradicional para a computacional.

Exemplo:

$$\left[\frac{2}{3}+(5-3)\right]+1=$$

Tradicional

(2/3+(5-3))+1=

Computacional

MODULARIZAÇÃO DE EXPRESSÕES

A modularização é a divisão da expressão em partes, proporcionando maior compreensão e definindo prioridades para resolução da mesma.

Como pode ser observado no exemplo anterior, em expressões computacionais usamos somente parênteses "( )" para modularização.

Na informática podemos ter parênteses dentro de parênteses.

Exemplos de prioridades:

(2+2)/2=2

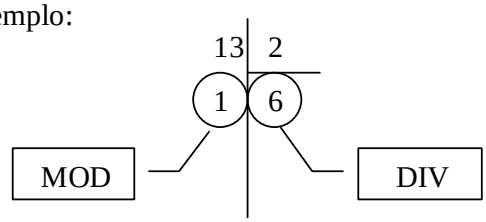
2+2/2=3

OPERADORES ESPECIAIS (MOD e DIV)

MOD ➔ Retorna o resto da divisão entre 2 números inteiros.

DIV ➔ Retorna o valor inteiro que resulta da divisão entre 2 números inteiros.

Exemplo:




```
13 DIV 2 = 6
13 MOD 2 = 1
```

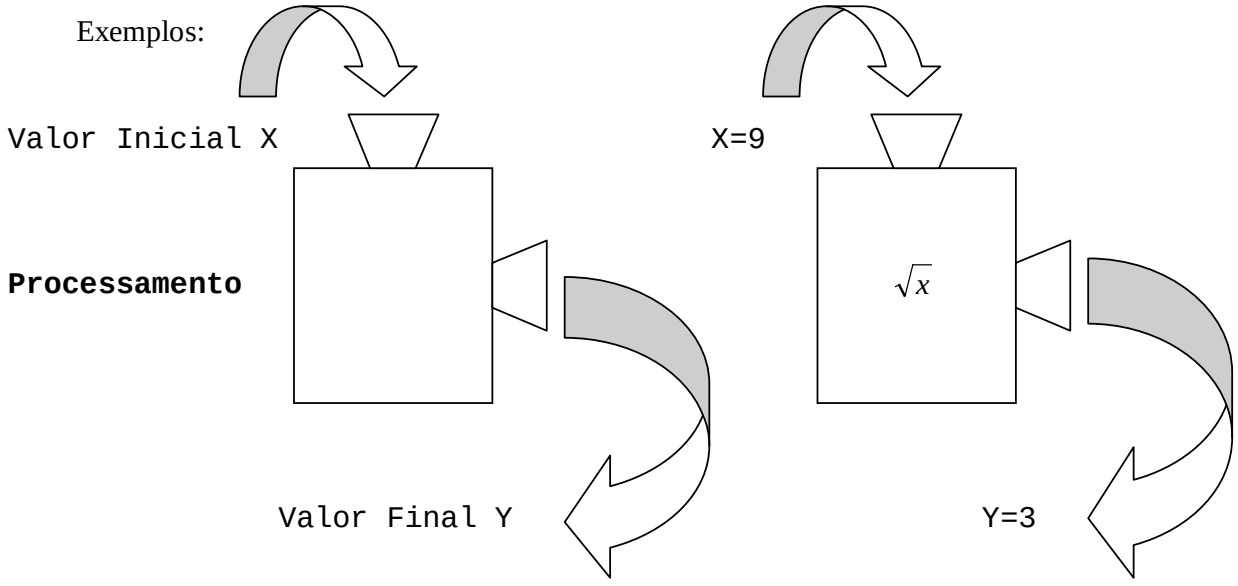
FUNÇÕES

Uma função é um instrumento (Sub–algoritmo) que tem como objetivo retornar um valor ou uma informação.

A chamada de uma função é feita através da citação do seu nome seguido opcionalmente de seu argumento inicial entre parênteses.

As funções podem ser predefinidas pela linguagem ou criadas pelo programador de acordo com o seu interesse.

Exemplos:



BIBLIOTECAS DE FUNÇÕES

Armazenam um conjunto de funções que podem ser usadas pelos programas.

FUNÇÕES PRÉ-DEFINIDAS

ABS( )	VALOR ABSOLUTO
SQRT( )	RAIZ QUADRADA
SQR( )	ELEVA AO QUADRADO
TRUNC( )	VALOR TRUNCADO
ROUND( )	VALOR ARREDONDADO
LOG( )	LOGARITMO
SIN( )	SENO
COS( )	COSENO
TAN( )	TANGENTE

As funções acima são as mais comuns e importantes para nosso desenvolvimento lógico, entretanto, cada linguagem possui suas funções própias. As funções podem ser aritméticas, temporais, de texto e etc.

OPERADORES LÓGICOS

Atuam sobre expressões retornando sempre valores lógicos como Falso ou Verdadeiro.

E	RETORNA VERDADEIRO SE AMBAS AS PARTES FOREM VERDADEIRAS.
OU	BASTA QUE UMA PARTE SEJA VERDADEIRA PARA RETORNAR VERDADEIRO.
NÃO	INVERTE O ESTADO, DE VERDADEIRO PASSA PARA FALSO E VICE-VERSA.

TABELA VERDADE

A	B	A E B	A OU B	NÃO (A)
V	V	V	V	F
V	F	F	V	F
F	V	F	V	V
F	F	F	F	V

EXPRESSÕES LÓGICAS

As expressões compostas de relações sempre retornam um valor lógico.

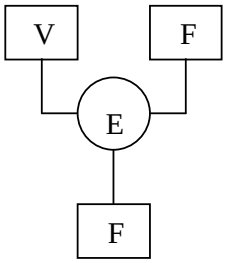
Exemplos:

2+5>4 ➔ Verdadeiro | 3<>3 ➔ Falso

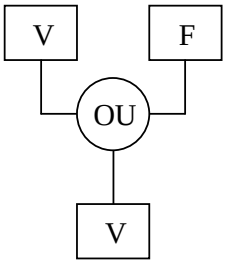
De acordo com a necessidade, as expressões podem ser unidas pelos operadores lógicos.

Exemplos:

2+5>4 E 3<>3 ➔Falso



2+5>4 OU 3<>3 ➔Verdadeiro



NÃO(3<>3)➔Verdadeiro

VARIÁVEIS

Variáveis são endereços de memória destinados a armazenar informações temporariamente.

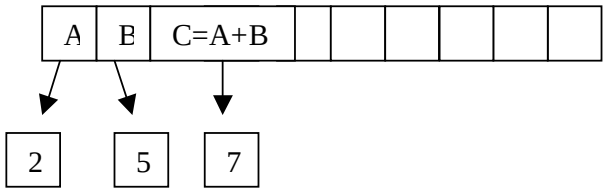
* Todo Algoritmo ou programa deve possuir variáveis!

VARIÁVEIS DE ENTRADA E SAÍDA

Variáveis de Entrada armazenam informações fornecidas por um meio externo, normalmente usuários ou discos.

Variáveis de Saída armazenam dados processados como resultados.

Exemplo:



De acordo com a figura acima A e B são Variáveis de Entrada e C é uma Variável de Saída.

CONSTANTES

Constantes são endereços de memória destinados a armazenar informações fixas, inalteráveis durante a execução do programa.

Exemplo:

PI = 3.1416

IDENTIFICADORES

São os nomes dados a variáveis, constantes e programas.

Regras Para construção de Identificadores:

- Não podem ter nomes de palavras reservadas (comandos da linguagem);
- Devem possuir como 1º caractere uma letra ou Underscore (_);
- Ter como demais caracteres letras, números ou Underscore;
- Ter no máximo 127 caracteres;
- Não possuir espaços em branco;
- A escolha de letras maiúsculas ou minúsculas é indiferente.

Exemplos:

NOME	TELEFONE	IDADE_FILHO
NOTA1	SALARIO	PI
UMNOMEMUITOCOMPRIDOEDIFICILDELER UM_NOME_MUITO_COMPRIDO_E_FACIL_DE_LER		

TIPOS DE DADOS

Todas as Variáveis devem assumir um determinado tipo de informação.

O tipo de dado pode ser:

- Primitivo ➔ Pré-definido pela linguagem;
- Sub-Faixa ➔ É uma parte de um tipo já existente;
- Escalar ➔ Definidos pelo programador.

Exemplos:

A : INTEIRO — PRIMITIVO

TIPO NOTA=[1..10] DE INTEIRO — SUB - FAIXA

TIPO SEMANA = (Segunda-feira, Terça-feira, Quarta-feira, Quinta-feira, Sexta-feira, Sábado, Domingo) — ESCALAR

TIPOS PRIMITIVOS DE DADOS

INTEIRO	ADMITE SOMENTE NÚMEROS INTEIROS. GERALMENTE É UTILIZADO PARA REPRESENTAR UMA CONTAGEM (QUANTIDADE).
REAL	ADMITE NÚMEROS REAIS (COM OU SEM CASAS DECIMAIS). GERALMENTE É UTILIZADO PARA REPRESENTAR UMA MEDIÇÃO.
CARACTERE	ADMITE CARACTERES ALFANUMÉRICOS. OS NÚMEROS QUANDO DECLARADOS COMO CARACTERES TORNAM SE REPRESENTATIVOS E PERDEM A ATRIBUIÇÃO DE VALOR.
LÓGICO	ADMITE SOMENTE VALORES LÓGICOS(VERDADEIRO/FALSO).

COMANDOS DE I/O (INPUT/OUTPUT)

LER ➔ Comando de entrada que permite a leitura de Variáveis de Entrada.

ESCREVER ➔ Comando de saída que exibe uma informação na tela do monitor.

IMPRIMIR ➔ Comando de saída que envia uma informação para a impressora.

SINAL DE ATRIBUIÇÃO

Uma Variável nunca é eternamente igual a um valor, seu conteúdo pode ser alterado a qualquer momento. Portanto para atribuir valores a variáveis devemos usar o sinal de "**:=".**

Exemplos:

A **:=** 2;

B **:=** 3;

C **:=** A + B;

SINAL DE IGUALDADE

As constantes são eternamente iguais a determinados valores, portanto usamos o sinal de "=".

Exemplos:

PI = 3.1416;

Empresa = 'Colégio de Informática L.T.D.A.'

V = Verdadeiro

CORPO GERAL DE UM PROGRAMA

```
PROGRAMA <<identificador>>;
CONST
    <<identificador>> = <<dado>>
VAR
    <<identificador>> :  <<tipo>>;
ÍNICIO
    { COMANDOS DE ENTRADA,PROCESSAMENTO E SAÍDA
    <<comando1>>;
    <<comandoN>>}
FIM.
```

ESTRUTURAS SEQÜÊNCIAIS

Como pode ser analisado no tópico anterior, todo programa possui uma estrutura seqüencial determinada por um ÍNICIO e FIM.

; PONTO E VÍRGULA ;

O sinal de ponto e vírgula “;” indica a existência de um próximo comando (passa para o próximo).

Na estrutura ÍNICIO e no comando que antecede a estrutura FIM não se usa “;”.

{LINHAS DE COMENTÁRIO}

Podemos inserir em um Algoritmo comentários para aumentar a compreensão do mesmo, para isso basta que o texto fique entre Chaves “{}”.

Exemplo:

LER (RAIO); {ENTRADA}

‘ASPAS SIMPLES’

Quando queremos exibir uma mensagem para a tela ou impressora ela deve estar contida entre aspas simples, caso contrário, o computador irá identificar a mensagem como Variável Indefinida.

Exemplo:

ESCREVER (‘AREA OBTIDA =’, AREA) {COMANDO DE SAÍDA}

AREA OBTIDA = X.XX {RESULTADO GERADO NA TELA}

ESTRUTURAS DE DECISÃO

Executa uma seqüência de comandos de acordo com o resultado de um teste.

A estrutura de decisão pode ser Simples ou Composta, baseada em um resultado lógico.

Simples:

```
SE <<CONDIÇÃO>>
        ENTÃO <<COMANDO1>>
```

Composta 1:

```
SE <<CONDIÇÃO>>
        ENTÃO <<COMANDO1>>
SENÃO <<COMANDO1>>
```

Composta 2:

```
SE <<CONDIÇÃO>>
        ENTÃO INICIO
                <<COMANDO1>>;
                <<COMANDON>>
            FIM;
        SENÃO INICIO
                <<COMANDO1>>; <<COMANDON>>
            FIM;
```

NINHOS DE SE

Usados para tomadas de decisões para mais de 2 opções.

Forma Geral:

```
SE <<CONDIÇÃO>>
        ENTÃO <<COMANDO1>>
        SENÃO SE <<CONDIÇÃO>>
                ENTÃO <<COMANDO1>>
                SENÃO <<COMANDO1>>
```

ESTRUTURAS DE CONDIÇÃO

A estrutura de condição eqüivale a um ninho de SE'S.

Forma Geral:

```
FACA CASO
    CASO <<CONDIÇÃO1>>
        <<COMANDO1>>;
    CASO <<CONDIÇÃON>>
        <<COMANDO1>>;
    OUTROS CASOS
        <<COMANDO1>>;
FIM DE CASO
```

ESTRUTURA DE REPETIÇÃO DETERMINADA

Quando uma seqüência de comandos deve ser executada repetidas vezes, tem-se uma estrutura de repetição.

A estrutura de repetição, assim como a de decisão, envolve sempre a avaliação de uma condição.

Na repetição determinada o algoritmo apresenta previamente a quantidade de repetições.

Forma Geral 1:

```
PARA <<VARIAVEL DE TIPO INTEIRO>>:=<<VALOR INICIAL>> ATE <<VALOR FINAL>> FAÇA
    <<COMANDO1>>;
```

Forma Geral 2:

```
PARA <<VARIAVEL DE TIPO INTEIRO>>:=<<VALOR INICIAL>> ATE <<VALOR FINAL>> FAÇA
    ÍNICIO
        <<COMANDO1>>;
        <<COMANDON>>
    FIM;
```

A repetição por padrão determina o passo do valor inicial até o valor final como sendo 1. Determinadas linguagens possuem passo –1 ou permitem que o programador defina o passo.

ESTRUTURA DE REPETIÇÃO INDETERMINADA COM VALIDAÇÃO INICIAL

É usada para repetir N vezes uma ou mais instruções. Tendo como vantagem o fato de não ser necessário o conhecimento prévio do número de repetições.

VALIDAÇÃO INICIAL

Forma Geral 1:

```
ENQUANTO <<CONDIÇÃO>> FACA
        <<COMANDO1>>;
```

Forma Geral 2:

```
ENQUANTO <<CONDIÇÃO>> FACA
        ÍNICIO
                <<COMANDO1>>;
                <<COMANDON>>
                                FIM;
```

TODAS AS VARIÁVEIS QUE ACUMULAM VALORES DEVEM RECEBER UM VALOR INICIAL.

ESTRUTURA DE REPETIÇÃO INDETERMINADA COM VALIDAÇÃO FINAL

Assim como a estrutura ENQUANTO É usada para repetir N vezes uma ou mais instruções. Sua validação é final fazendo com que a repetição seja executada pelo menos uma vez.

Forma Geral;

REPITA

    <<COMANDO1>>;

    <<COMANDON>>

ATE <<CONDIÇÃO>>

EXERCÍCIOS

Observação, antes de tudo:
Os termos "Ler" são referentes à linguagem Pascal e trabalharemos com a linguagem Object Pascal.
Fica sub-entendido então, que o Delphi sabe "Ler" ou em Pascal "Read".

Ler o nome e as 4 notas de um aluno, calcular a média e apresentar o nome e a média:

```
procedimento Media;
var
  Nome: Texto;
  A,B,C,D: Inteiro;
  Media: Real;
inicio
  Ler(Nome);
  Ler(A,B,C,D);
  Media:=(A+B+C+D)/4;
  MostrarMensagem(Nome,Media);
fim;
```

Calcular a área de um círculo:

```
procedimento Calcular_Area;
var
  Raio, Area: Real;
inicio
  Ler(Raio);
  Area:=3.14*Raio*Raio;
  MostrarMensagem(Area);
end;
```

Ler 2 números e mostrar o maior deles:

```
procedimento Maior_de_dois;
var
  A,B: Inteiro;
inicio
  Ler(A,B);
  se A>B então MostrarMensagem('A') senão MostrarMensagem('B');
fim;
```

Ler o nome e as 4 notas de um aluno, calcular a média, se esta for maior ou igual a 6, apresentar que o mesmo passou, senão que foi reprovado:

```
procedimento
var
  Nota1, Nota2, Nota3, Nota4: Inteiro;
  Media: Real;
  Nome: Texto;
inicio
  Ler(Nome);
  Ler(Nota1,Nota2,Nota3,Nota4);
  Media:=(Nota1+Nota2+Nota3+Nota4)/4;
  se Media>=6 então MostrarMensagem('Parabéns, você foi aprovado!') senão
MostrarMensagem('Não consegui, estude mais!');
fim;
```

Ler 3 números e mostrar o maior deles:

```
procedimento Maior_de_tres;
var
  A,B,C: Inteiro;
inicio
  Ler(A,B,C);
  se (A>B) e (A>C) então MostrarMensagem('A')
                   senão se B>C então MostrarMensagem('B')
                            senão MostrarMensagem('A');
fim;
```

... ou poderia ser escrito de outra forma também, pois o ponto e vírgula que indica o final da frase somente.

```
procedimento Maior_de_tres;
var
  A,B,C: Inteiro;
inicio
  Ler(A,B,C);
  se (A>B) e (A>C) então MostrarMensagem('A') senão se B>C então
MostrarMensagem('B') senão MostrarMensagem('A');
fim;
```

... ou

```
procedimento Maior_de_tres;
var
  A,B,C: Inteiro;
inicio
  Ler(A,B,C);
  se (A>B) e (A>C) então MostrarMensagem('A');
  se (B>A) e (B>C) então MostrarMensagem('B');
  se (C>A) e (C>B) então MostrarMensagem('C');
fim;
```

Escrever 10 vezes a palavra FLUMINENSE (2 soluções):

```
procedimento Fluminense;
inicio
var
  X: Inteiro;
inicio
  X:=0;
  enquanto x<10 faça
    inicio
      MostrarMensagem('FLUMINENSE');
      Incrementar(X);
    fim.
fim;
```

```
procedimento Fluminense;
inicio
var
  X: Inteiro;
inicio
  para X:=1 até 10 faça MostrarMensagem('FLUMINENSE');
fim;
```

Escrever os 100 primeiros números pares:

```
procedimento Pares;
var
  X, Par: Inteiro;
inicio
  Par:=0;
  para X:=1 até 100 faça
    inicio
      MostrarMensagem(Par);
      Incrementar(Par,2);
    fim;
fim;
```

... ou

```
procedimento Pares;
var
  X, Par: Inteiro;
inicio
  Par:=0;
  para X:=1 até 100 faça
    inicio
      MostrarMensagem(Par);
      Par:=Par+2;
    fim;
fim;
```

Ler e somar todos os salários de uma empresa até que se entre com o valor zero:

```
procedimento Somatorio;
var
  Total, Salario: Real;
inicio
  Total:=0;
  Salario:=1; //ou qualquer número diferente de zero
  enquanto Salario<>0 faça
    inicio
      Ler(Salario);
      Total:=Total+Salario;
    fim;
  MostrarMensagem(Total);
fim;
```

... ou

```
procedimento Somatorio;
var
  Total, Salario: Real;
inicio
  Total:=0;
  Salario:=1;
  enquanto Salario<>0 faça
    inicio
      Ler(Salario);
      Incrementar(Total,Salario);
    fin;
  MostrarMensagem(Total);
fim;
```

... ou

```
procedimento Somatorio;
var
  Total, Salario: Real;
inicio
  Total:=0;
  repita
    Ler(Salario);
    Total:=Total+Salario;
  até Salário=0;
  MostrarMensagem(Total);
fim;
```

Ler o nome e as 4 notas de um aluno, tendo a 1ª peso 1, a 2ª peso 2, a 3ª peso 3 e a 4ª peso 4, calcular a média ponderada e apresentar o nome e a média:

```
procedimento Media;
var
  Nome: Texto;
  A,B,C,D: Inteiro;
  Media: Real;
inicio
  Ler(Nome);
  Ler(A,B,C,D);
  Media:=(A*1+B*2+C*3+D*4)/10; //10=1+2+3+4
  MostrarMensagem(Nome,Media);
fim;
```

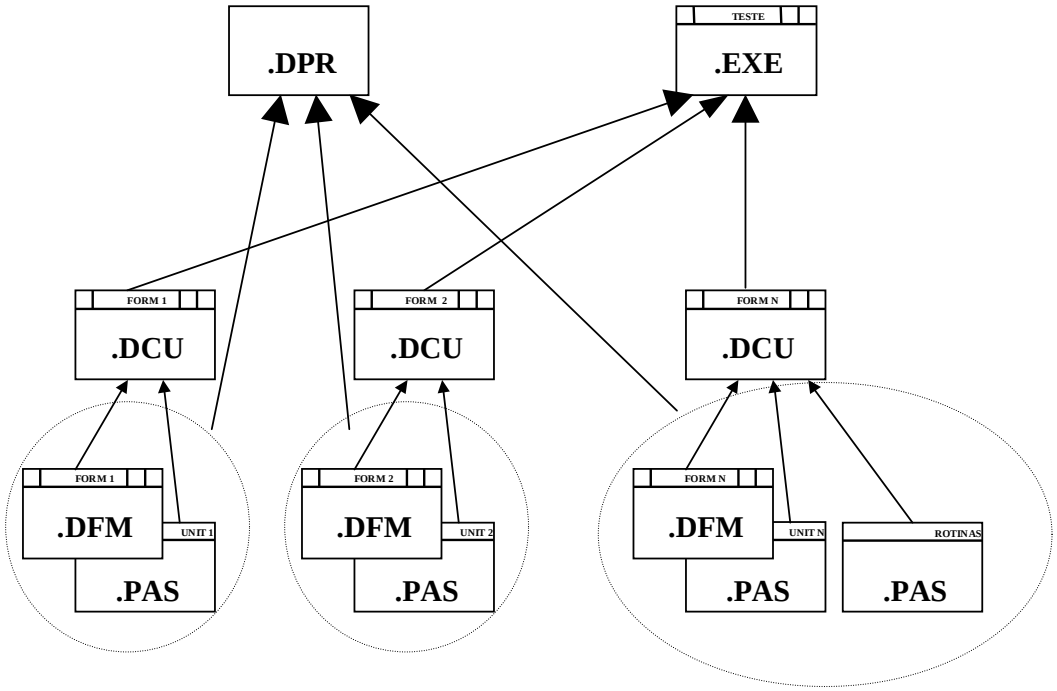
O que é Delphi?

O Delphi é um ambiente de desenvolvimento de aplicações, orientado a objeto, que permite o desenvolvimento de aplicações para os Sistemas Operacionais Windows 3.11, Windows 95/98/Me e Windows 2000, “com pouca codificação”.

Como é formado uma Aplicação em Dephi

Quando você abre um projeto no Delphi, ele já mostra uma Unit com várias linhas de código. Este texto tem como objetivo explicar um pouco desta estrutura que o Delphi usa. Um projeto Delphi tem, inicialmente, duas divisórias: uma Unit, que é associada a um Form, e outra Project, que engloba todos os FORM e UNITs da aplicação.

Em Delphi temos: o Project, os Forms e as Units. Para todo Form temos pelo menos uma Unit (Código do Form), mas temos Units sem Form (códigos de procedures, funções, etc.).



Arquivos Gerados no Desenvolvimento

Extensão	Definição	Função
.DPR	Arquivo do Projeto	Código fonte em Pascal do arquivo principal do projeto. Lista todos os formulários e units no projeto, e contém código de inicialização da aplicação. Criado quando o projeto é salvo.
.PAS	Código fonte da Unit (Object Pascal)	Um arquivo .PAS é gerado por cada formulário que o projeto contém. Seu projeto pode conter um ou mais arquivos .PAS associados com algum formulário. Contem todas as declarações e procedimentos incluindo eventos de um formulário.
.DFM	Arquivo gráfico do formulário	Arquivo que contém as propriedades do desenho de um formulário contido em um projeto. Um .DFM é gerado em companhia de um arquivo .PAS para cada formulário do projeto.
.OPT	Arquivo de opções do projeto	Arquivo texto que contém a situação corrente das opções do projeto. Gerado com o primeiro salvamento e atualizado em subsequentes alterações feitas para as opções do projeto.
.RES	Arquivo de Recursos do Compilador	Arquivo que contém o ícone, mensagens da aplicação e outros recursos usados pelo projeto.
.~DP	Arquivo de Backup do Projeto	Gerado quando o projeto é salvo pela segunda vez.
.~PA	Arquivo de Backup da Unit	Se um .PAS é alterado, este arquivo é gerado.
.~DF	Backup do Arquivo gráfico do formulário	Se você abrir um .DFM no editor de código e fizer alguma alteração, este arquivo é gerando quando você salva o arquivo.
.DSK	Situação da Área de Trabalho	Este arquivo armazena informações sobre a situação da área de trabalho especifica para o projeto em opções de ambiente (Options Environment).

Obs.: .~DF, .~PA , .~DP são arquivos de backup (Menu Options, Enviroment, Guia Editor Display, Caixa de Grupo Display and file options, opção Create Backup Files, desativa o seu salvamento).

Devido a grande quantidade de arquivos de uma aplicação, cada projeto deve ser montado em um diretório específico.

Arquivos Gerados pela Compilação

Extensão	Definição	Função
.EXE	Arquivo compilado executável	Este é um arquivo executável distribuível de sua aplicação. Este arquivo incorpora todos os arquivos .DCU gerados quando sua aplicação é compilada.
.DCU	Código objeto da Unit	A compilação cria um arquivo .DCU para cada .PAS no projeto.

Neste arquivo está escrito o código de criação da aplicação e seus formulários. O arquivo Project tem apenas uma seção.

Esta seção é formada pelo seguinte código:

PROGRAM - Define o Projeto;
USES Cláusula que inicia uma lista de outras unidades.
Forms = É a unidade do Delphi que define a forma e os componentes do aplicativo
in = A clausula indica ao compilador onde encontrar o arquivo Unit.
Unit1 = A unidade que você criou
*{$R *.RES}* - Diretiva compiladora que inclui o arquivo de recursos.

Abaixo veja como fica o Project quando você abre um projeto novo:

```
program Project1;

uses
  Forms,
  Unit1 in 'UNIT1.PAS' {Form1};

{$R *.RES}

begin
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.
```

Nesta divisória serão escritos os códigos dos seus respectivos forms (Unit1 = Form1). Aqui serão definidos os códigos de cada procedimento dos componentes que você colocar no form.

Seção Unit

Declara o nome da unit.

Seção Uses

Contém as units acessadas por este arquivo.

Seção Interface

Nesta seção estão as declarações de constantes, tipos de variáveis, funções e procedures gerais da Unit/Form.

INTERFACE Palavra que inicia a seção;

USES Cláusula que inicia uma lista de outras unidades compiladas (units) em que se basea:

SysUtils = utilitários do sistema (strings, data/hora, gerar arquivos)

WinProcs = acesso a GDI, USER e KERNEL do Windows

Wintypes = tipos de dados e valores constantes

Messages = constantes com os números das mensagens do Windows e tipos de dados das Mensagens

Classes = elementos de baixo nível do sistema de componentes

Graphics = elementos gráficos

Controls = elementos de nível médio do sistema de componentes

Forms = componentes de forma e componentes invisíveis de aplicativos

Dialogs = componentes de diálogo comuns

Seção Type

Declara os tipos definidos pelo usuário. Subseções: Private, declarações privativas da Unit. Public, declarações publicas da Unit.

Seção Var

Declara as variáveis privadas utilizadas.

Seção Implementation

Contém o corpo das funções e procedures declaradas nas seções **Interface** e **Type**. Nesta seção também estão definidos todos os procedimentos dos componentes que estão incluídos no Form. As declarações desta seção são visíveis apenas por ela mesma. Esta seção é formada pelo seguinte código:

{$R.DFM}* - Diretiva compiladora que inclui toda a **interface**, propriedades da forma e componentes do arquivo *.DFM

Seção uses adicional

Serve para declarar Units que ativam esta.

Inicialization

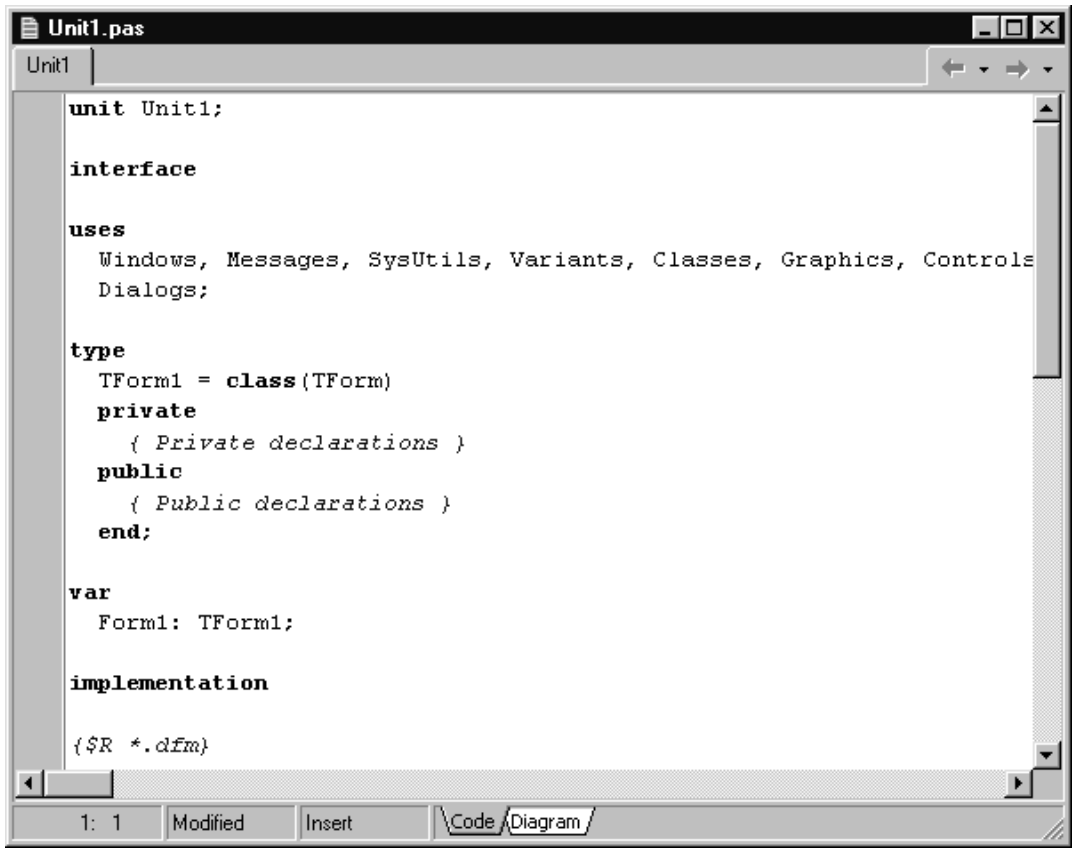
Nesta seção, que é opcional, pode ser definido um código para proceder as tarefas de inicialização da Unit quando o programa começa. Ela consiste na palavra reservada inicialization seguida por uma ou mais declarações para serem executadas em ordem.

Abaixo veja como fica a unit quando você abre um projeto novo:

```
unit Unit1;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics,
Controls, Forms, Dialogs;

type
  TForm1 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

{Uses Adicional}

{Initialization}

end.
```

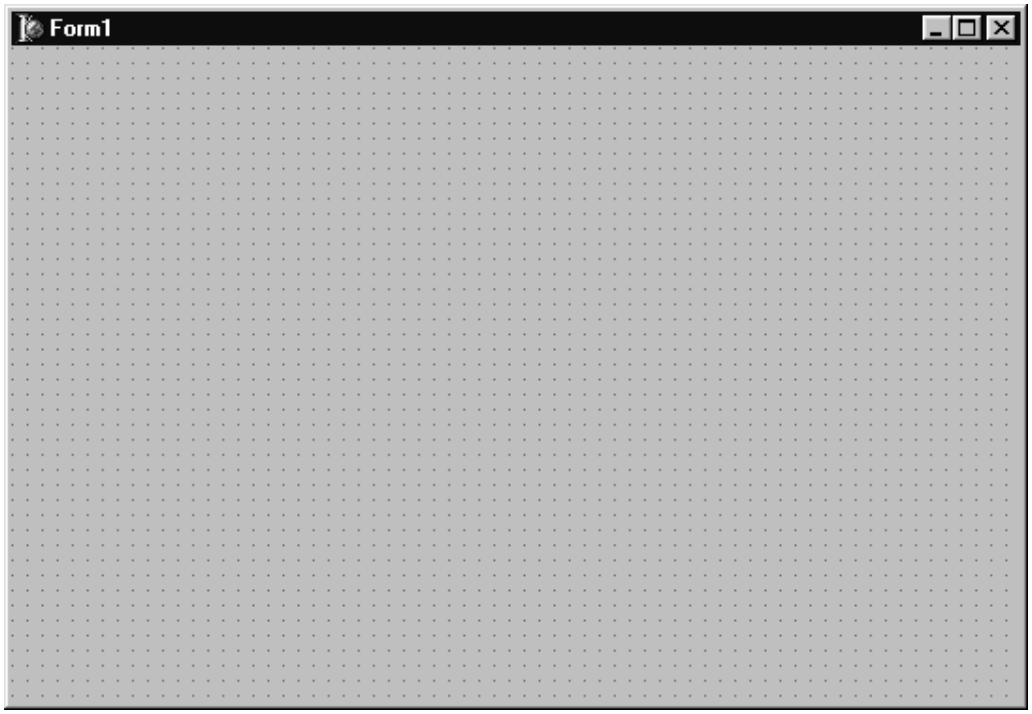
Janelas

O Code Editor funciona como um arquivo texto qualquer, onde serão implementadas os procedimentos e eventos do programa. O próprio Delphi efetua as mudanças de formatação na fonte (negrito, cores, etc.), exceto maiúsculas e minúsculas.



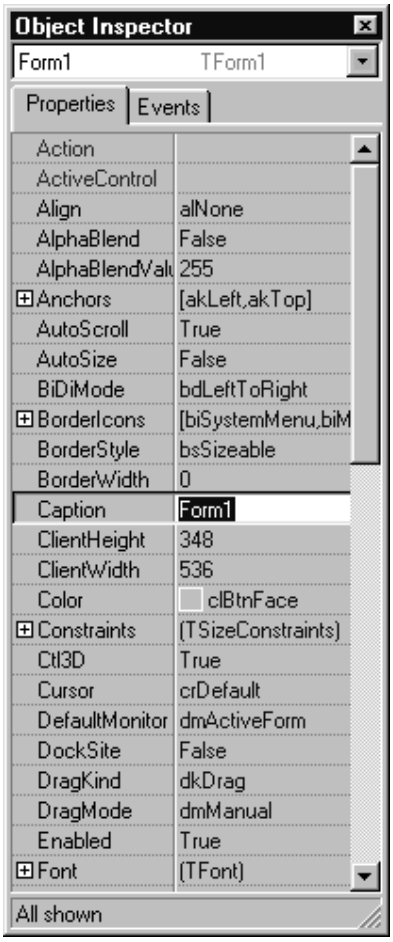
Formulários

Você usa formulários para fazer **interface** com o usuário, nele são inseridos os componentes. O formulário é uma janela, e portanto, possui os atributos de uma janela (menu de controle, botões de maximizar e minimizar, barra de título, bordas redimensionáveis).



Object Inspector

É uma ferramenta composta de duas páginas: Properties (Propriedades) e Events (Eventos).



A página Properties (Propriedades) permite que você estabeleça parâmetros de formulários e componentes. Estes parâmetros especificam os valores iniciais de características como nome do componente e sua posição no formulário.

A páginas Events (Eventos) permite associar os componentes com ações do usuário.

Component Palette

É composta de várias páginas contendo cada uma grupos de componentes. Para mudar de página na palheta de componentes, clique sobre a guia da página que você quiser acessar.

Orientação a Objetos

Por ser baseado no Pascal Object, o Delphi permite que se construa aplicações orientadas a objetos. Em linhas gerais, aplicações orientadas a objetos se baseiam no conceito de classe. A classe é um tipo de dados, contendo atributos e serviços. O objeto é uma variável de determinada classe. Por exemplo, um formulário nada mais é do que um objeto da classe Formulário (Tform). Contém atributos e serviços.

Os programas feitos em Delphi são orientados a eventos. Eventos são ações normalmente geradas pelo usuário. Ex.: Clicar o mouse, pressionar uma tecla, mover o mouse, etc. Os eventos podem ser também gerados pelo Windows.

Existem eventos associados ao formulário e cada componente inserido neste.

Exemplos

Ao formulário está ligado *on create*, que ocorre quando mostramos o formulário na tela.

Ao componente botão está ligado o evento *on click*, que ocorre quando damos um click com o mouse sobre o botão.

Rotinas que Respondem a Eventos

Cada evento gera uma **procedure**, aonde você deve inserir as linhas de código que envolvem este evento. Por exemplo, o evento OnClick, que é gerado ao clicarmos em um botão chamado BTNSair, cria a **procedure**:

```
procedure TForm1.BTNSairClick(Sender: Tobject);
```

Onde TForm1 é o objeto Tform que contém o botão BTNSair, e Sender é um objeto Tobject que representa o componente que deu origem ao evento.

Se você quiser inserir uma rotina que trate um determinado evento de um componente, faça o seguinte:

- clique sobre o componente;
- no Object Inspector, seleciona a página Events;
- dê um duplo clique sobre o evento para o qual quer inserir o código;
- entre no editor de código e escreva as linhas de código.

```
procedure TForm1.BTNSairClick(Sender: Tobject);
begin
  Form1.Close;
end;
```

Obs.: Escreva seu código entre o **begin** e o **end**, se por acaso você quiser retirar o evento e o componente, retire primeiro os eventos do componente removendo somente o código que você colocou e depois o componente; os resto dos procedimentos o Delphi retira para você.

Como vimos, eventos podem estar associados a modificações em propriedade de componente e formulário, ou seja, você pode modificar propriedades de formulários e componentes durante a execução do sistema. Para isto você deverá usar a sintaxe:

<componente>.<propriedade>;

Por exemplo, para modificar a propriedade *text* de uma caixa de edição Edit1 para "Bom Dia" faça:

Edit1.Text := 'Bom Dia';

Se a propriedade do componente tiver subpropriedades, para acessa-lá, utilize a seguinte sintaxe:

<componente>.<propriedade>.<subpropriedade>

Por exemplo, para modificar a subpropriedade Name referente a propriedade fonte, de uma caixa de edição Edit1, para 'Script', faça:

Edit1.Font.name := 'Script';

Obs.: Verifique o tipo da propriedade para antes de mandar o valor, consultando no Objetc Inspector.

Métodos

São procedures ou funções embutidas nos componentes e formulários, previamente definidas pelo Delphi.

Alguns métodos são descritos a seguir:

- Show : Mostra um formulário;
- Hide : Esconde um formulário mais não o descarrega;
- Print : Imprime um formulário na impressora;
- SetFocus : Estabelece o foco para um formulário ou componente;
- BringtoFront: Envia para frente.

Chamado de métodos como resposta a eventos

Um evento pode gerar a chamada para um método, ou seja, uma subrotina previamente definida para um componente.

No código, use a seguinte sintaxe: <nome do objeto>.<método>

Por exemplo, clicar em um botão pode dar origem ao evento Show de um outro formulário, mostrando este novo formulário na tela: Form2.show;

Trabalhando com Banco de Dados

O Borland Database Engine é o coração do Delphi e suas aplicações com banco de dados usando o mesmo database engine usado pelo Paradox e dBase. Paradox e dBase é claro trazem capacidades adicionais além do database engine (como o Delphi também o faz), mas isso é longe de dizer que o valor agregado destes recursos adicionais é maior de 2 % das capacidades - os outros 98% são providos pelo database engine e estão disponíveis para todos os usuários deste engine.

O Borland Database Engine (BDE) é uma coleção de DLLs que as aplicações de banco de dados irão fazer chamadas. Cada estação de trabalho que tiver a aplicação de banco de dados instalada deverá ter, também, o BDE instalado (o Delphi vem com a instalação do BDE para você adicionar a sua aplicação).

O BDE permite a você usar tabelas dBase, Paradox ou ODBC em modo multi-usuário. A versão Cliente/Servidor do Delphi também vem com links para servidores de banco de dados como Oracle, Sybase, MS SQL Server, Informix, e InterBase.

A Linguagem SQL

A linguagem SQL (Structured Query Language - Linguagem Estruturada de Pesquisa) foi criada para ser uma linguagem padrão para consulta, atualização e manipulação de banco de dados em um banco de dados relacional.

Comercialmente implementada pela IBM, se tornou um padrão de linguagem de acesso a dados em vários bancos de dados relacionais, como Oracle, DB2, SQL Server, Sybase, Interbase, etc.

Usaremos as Declarações em SQL para extrair/atualizar registros de uma ou mais tabelas que atendam as condições especificadas, manipulando, assim, somente os dados que sejam de nosso interesse.

Por exemplo, a declaração permite que somente os registros cujo o campo Nome começando pela letra A da tabela de Clientes sejam exibidos na tela:

```
Select *
From Clientes
Where Nome="A*"
```

Podemos dividir os comandos da linguagem SQL em três categorias distintas:

- Comandos de Definição de Dados: permitem definir ou alterar tabelas em um banco de dados.
- Comandos de Controle de Dados: servem para gerenciar o acesso dos usuários a determinadas tabelas.
- Comandos para a Manipulação de Dados: servem para manipular os dados contidos nas tabelas.

Exemplos

1)
```
Select *
From Clientes
```

Seleciona todos(*) os campos de todos registros da tabela clientes.

2)
```
Select Codigo,Nome
From Clientes
Where Codigo > 10 And Codigo < 200
```

Seleciona os campos Código e Nome da tabela Clientes para os registros que tenham os campo Código > (Maior) que 10 e <(Menor) que 200.

3)
```
Select *
From Clientes
Group By Cidade
```

Seleciona todos os campos e registros da tabela Clientes agrupada pelo campo Cidade.

4)
```
Select *
From Clientes
Order By Codigo
```

Seleciona todos os campos e registros da tabela Clientes ordenada pelo campo Codigo.

5)
```
Select *
From Fornecedores, Produtos
```

```
     Where Fornecedores.Codigo = Produtos.Codigo
```

Seleciona todos os campos e registros das tabelas de Fornecedores e Produtos que tenham o campo codigo de Fornecedores igual ao campo Codigo de Produtos.

```
6)   Select *
     From Clientes
     Where Nome Like “S*”
```

Seleciona todos os campos e registros da tabela de Clientes cujo o campo Nome comece pela letra S.

```
7)   Update Funcionarios
     Set Salario = Salario * 1.2
```

Atualiza o campo Salario de todos os registros da tabela de Funcionarios para o conteúdo atual multiplicado por 1.2 (aumento de 20%).

```
8)   Update Funcionarios
     Set Salario = Salario * 1.2
     Where Cargo = “Diretor”
```

Atualiza o campo Salario de todos os registros da tabela de Funcinarios que campo Cargo seja igual a “Diretor” para o conteúdo atual multiplicado por 1.2 (aumento de 20%).

```
9)   Delete From Produtos
     Where Codigo > 5 And Codigo < 20
```

Apaga todos os registros da tabela Produtos para Codigo >(maior) que 5 e < (menor) que 20.

Construindo uma Consulta Dinamicamente

Procedimento para criar construir uma consulta em SQL dinamicamente.

```
Query1.Close; {Fecha a query}
Query1.SQL.Clear; {Limpa o conteúdo da propriedade SQL}
Query1.SQL.Add(‘Select * From Country Where Name = “Argentina” ’);
Query1.Open; {Abre e executa a Query}
```

QuickReport

O Presente trabalho tem como objetivo demostrar os procedimentos para criar relatórios em programas feitos em Delphi utilizando o QuickReport. O Delphi 1.0 já possui um gerador de relatórios o ReportSmith que perde em desempenho e desenvoltura em relação ao QuickReport, além de necessitar de sua instalação prévia quando da sua utilização. Por ser um programa a parte torna-se muito lento em relação a o QuickReport que é uma biblioteca de componentes que passa a fazer parte da aplicação quando da sua utilização após a compilação.

O QuickReport é um gerador de relatórios por banda. Seu relatório é construído em cima de vários tipos de banda, onde são colocados componentes visíveis como campos de banco de dados, rótulos, imagens e outros componentes imprimíveis.

QuickReport pode criar vários tipos de relatórios, de fato com um breve conhecimento de Delphi e um pouco de criatividade pode ser usado para criar qualquer tipo de relatório.

Lembrar ainda que se os relatórios são criados como as forms, mas não com a intenção de ser mostrado na tela como forms. Em uma Form relatório deve ser incluído um componente TQuickReport, e você usa neste componente os métodos PRINT e PREVIEW para criar seu relatório. Não tente usar em um relatório os métodos de Form SHOW ou SHOWMODAL. Uma Form relatório nunca deverá ser em sua aplicação o Form principal.

Exceções

A Estrutura Try…Finally…End

Seu código precisa garantir que, mesmo ocorrendo um erro, os recursos alocados sejam desalocados. Entre estes recursos estão: arquivos, memória, recursos do Windows, objetos.

Para garantir a desalocação dos recursos, usamos a estrutura abaixo:

```
{Aloca os recursos}
try
     {Comandos que usam os recursos}
finally
     {Libera os recursos}
end;
```

A aplicação sempre executará os comandos inseridos na parte Finally do bloco, mesmo que uma execeção ocorra. Quando um erro ocorre no bloco protegido, o programa pula para a parte finally, chamada de código limpo, que é executado. Mesmo que não ocorra um erro, estes comandos são executados.

No código a seguir, alocamos memória e geramos um erro, ao tentarmos uma divisão por 0. Apesar do erro o programa libera a memória alocada:

```
procedure TForm1.Button1Click(Sender:TObject);
var
   Apointer : Pointer;
   AnInteger, Adividend : Integer;
begin
   Adividend := 0;
   GetMem(Apointer, 1024);
   try
     Aninteger := 10 Div Adividend;
   finally
      FreeMem(Apointer, 1024);
   end;
end;
```

A Estrutura Try…Except…End

Um tratamento de exceção é um código que trata erros que ocorrem dentro de blocos protegidos. Para definir um tratamento de exceção, utilize a seguinte construção:

```
try
     {Comandos que você deseja proteger}
except
     {Comandos de tratamento de erros}
end;
```

A aplicação irá executar os comandos na parte except somente se ocorrer um erro. Se na parte try você chamar uma rotina que não trata erros, e um erro ocorrer, ao voltar para este bloco a parte except será executada. Uma vez que a aplicação localiza um tratamento para exceção ocorrida, os comandos são executados, e o objeto exceção é destruído. A execução continua até o fim do bloco.

Dentro da parte except definimos um código a ser executado para manipular tipos específicos de exceção. Por exemplo, o código abaixo trata o erro de divisão por zero, através da exceção EDivByZero:

```
function Divisão (Soma, Numero : Integer) : Integer;
begin
   try
      Result := Soma div Numero;
   except
      on EDivByZero do Result 0;
   end;
end;
```

A palavra reservada *on* define respostas para uma exceção. *On* está sempre junto de *do*, para manipular a exceção.

Para ler informações específicas sobre o erro ocorrido, você usa uma variação da estrutura *on…do*, que provê uma variável temporária que engloba a exceção. Nesse caso, você poderá criar seu próprio quadro de mensagem contendo a string da mensagem da exceção:

```
try
   {Comandos}
except
   on E:EInvalidOperation do
       MessageDlg('Ígnorando a exceção : '+E.Message,
mtinformation,[mbOk],0);
end;
```

Onde a variável temporária *E* é do tipo *EInvalidOperation.*

Você pode prover um tratamento padrão de erro para tratar exceções que não tem tratamentos especificados. Para isto, adicione uma parte *else* na parte except do bloco:

```
try
   {Comandos}
except
   on EPrimeiroTipo do
     {Código especifico para o primeiro tipo de erro]
   on ESegundoTipo do
     {Código específico para o segundo tipo de erro}
   else
     {Código padrão de tratamento de erros}
end;
```

Além disso, você pode utilizar as exceções genéricas para tratar um erro, em vez de uma exceção específica. Por exemplo, se você quer tratar um erro relacionado a uma operação com inteiros, mas não sabe exatamente o erro, poderá utilizar a exceção *EIntError*, que é a exceção genérica da qual derivam outras exceções relacionadas a inteiros:

```
try
   {Comandos}
except
   on EIntError do
   {Código de tratamento de erros}
end;
```

Exceções Silenciosas

Você pode definir exceções que não mostram um quadro de mensagem para o usuários quando aparecem. São chamadas exceções sileciosas.

O caminho mais curto para criar esta exceção é através da **procedure** *Abort.* Esta **procedure** automaticamente gera uma exceção do tipo *EAbort*, que abortará a operação sem mostrar uma mensagem.

O exemplo abaixo aborta a operação de inclusão de itens em um ListBox quando tentamos inserir o terceiro elemento:

```
begin
   for i := 1 to 10 do
      begin
         ListBox1.items.Add(IntToStr(i));
         if i = 3 then Abort;
      end;
end;
```

Algoritmo – Prova

1ª Fase

```
procedimento Passar_em_Delphi;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Recuperacao;
fim;

procedimento FazerProva;
inicio
  Nota:=0;
  Pergunta1;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta2;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta3;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta4;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta5;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta6;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta7;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta8;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta9;
  se Resposta=Certa então Incrementar(Nota);
  Pergunta10;
  se Resposta=Certa então Incrementar(Nota);
fim;

procedimento Aprovado;
inicio
  MostrarMensagem('Parabéns, você foi aprovado!);
fim;

procedimento Recuperacao;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Reprovado;
fim;

procedimento Reprovado;
inicio
  MostrarMensagem('Você não conseguiu, estude mais!);
fim;
```

2ª Fase

```
procedimento Passar_em_Delphi;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Recuperacao;
fim;

procedimento FazerProva;
inicio
  Nota:=0;
  FazerPergunta('1 + 1 é igual a 2?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Um triângulo tem 4 lados?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('O MSWord é um sistema operacional?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('A água do mar é salgada?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('A letra A é uma vogal?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Usamos crase antes de substantivo masculino?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('Em Sto A de Pádua temos CETEP?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Um quadrado possui lados iguais?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('5 * 3 = 16?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('O ano tem 12 meses?');
  se Resposta='SIM' então Incrementar(Nota);
fim;

procedimento Aprovado;
inicio
  MostrarMensagem('Parabéns, você foi aprovado!);
fim;

procedimento Recuperacao;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Reprovado;
fim;

procedimento Reprovado;
inicio
  MostrarMensagem('Você não conseguiu, estude mais!);
fim;
```

3ª Fase

```
var
  Nota: Inteiro;
  Resposta: Texto;

procedimento Passar_em_Delphi;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Recuperacao;
fim;

procedimento FazerProva;
inicio
  Nota:=0;
  FazerPergunta('1 + 1 é igual a 2?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Um triângulo tem 4 lados?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('O MSWord é um sistema operacional?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('A água do mar é salgada?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('A letra A é uma vogal?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Usamos crase antes de substantivo masculino?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('Em Sto A de Pádua temos CETEP?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Um quadrado possui lados iguais?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('5 * 3 = 16?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('O ano tem 12 meses?');
  se Resposta='SIM' então Incrementar(Nota);
fim;

procedimento Aprovado;
inicio
  MostrarMensagem('Parabéns, você foi aprovado!);
fim;

procedimento Recuperacao;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Reprovado;
fim;

procedimento Reprovado;
inicio
  MostrarMensagem('Você não conseguiu, estude mais!);
fim;
```

4ª Fase

```
var
  Nota: Inteiro;
  Resposta: Texto;

procedimento Passar_em_Delphi;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Recuperacao;
fim;

procedimento FazerProva;
inicio
  Nota:=0;
  FazerPergunta('1 + 1 é igual a 2?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Um triângulo tem 4 lados?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('O MSWord é um sistema operacional?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('A água do mar é salgada?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('A letra A é uma vogal?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Usamos crase antes de substantivo masculino?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('Em Sto A de Pádua temos CETEP?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('Um quadrado possui lados iguais?');
  se Resposta='SIM' então Incrementar(Nota);
  FazerPergunta('5 * 3 = 16?');
  se Resposta='NÃO' então Incrementar(Nota);
  FazerPergunta('O ano tem 12 meses?');
  se Resposta='SIM' então Incrementar(Nota);
fim;

procedimento Aprovado;
inicio
  MostrarMensagem('Parabéns, você foi aprovado!);
fim;

procedimento Recuperacao;
inicio
  FazerProva;
  se Nota>=7 então Aprovado senão Reprovado;
fim;

procedimento Reprovado;
inicio
  MostrarMensagem('Você não conseguiu, estude mais!);
fim;

procedimento FazerPergunta(Questao: Texto);
inicio
  Mostrar_Formulario; //Com 2 botões, um SIM e um NÃO, aguardar implementação em
Delphi.
fim;
```

5ª Fase

```
var
  Nota: Integer;
  Resposta: String;

procedure Passar_em_Delphi;
begin
  FazerProva;
  if Nota>=7 then Aprovado else Recuperacao;
end;

procedure FazerProva;
begin
  Nota:=0;
  FazerPergunta('1 + 1 é igual a 2?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('Um triângulo tem 4 lados?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('O MSWord é um sistema operacional?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('A água do mar é salgada?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('A letra A é uma vogal?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('Usamos craif antes de substantivo masculino?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('Em Sto A de Pádua temos CETEP?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('Um quadrado possui lados iguais?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('5 * 3 = 16?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('O ano tem 12 meifs?');
  if Resposta='SIM' then Inc(Nota);
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado!);
end;

procedure Recuperacao;
begin
  FazerProva;
  if Nota>=7 then Aprovado else Reprovado;
end;

procedure Reprovado;
begin
  ShowMessage('Você não consiguiu, estude mais!);
end;

procedure FazerPergunta(Questao: String);
begin
  Form2.ShowModal;
end;
```

Complementação para formulário de respostas (Form2):

Acrescentar 2 botões, ao clicar no botão SIM: Resposta:='SIM' e fecha, ao clicar no botão NÃO: Resposta:='NÃO' e fecha;

```
procedure Botao_SimClick;
begin
  Resposta:='SIM';
  Close;
end;

procedure Botao_NaoClick;
begin
  Resposta:=NÃO';
  Close;
end;
```

Projeto – Prova

1ª Fase

```
unit ProvaU;

//1ª Fase
//Inicia a prova
//Faz 10 perguntas
//Verifica nota e resultado

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs;

type
  TForm1 = class(TForm)
    Button1: TButton;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Nota: Integer;
  Resposta: String;

implementation

uses PerguntaU;

{$R *.dfm}

procedure FazerPergunta(Questao: String);        //Abre o formulário com a
pergunta
begin
  Form2.Label1.Caption:=Questao;
  Form2.ShowModal;
end;

procedure FazerProva;
begin
  Form1.Hide;
  Nota:=0;
  FazerPergunta('1 + 1 é igual a 2?');
  if Resposta='SIM' then Inc(Nota);              //Verifica se a resposta está
correta ou não, se estiver, incrementa a nota
  FazerPergunta('Um triângulo tem 4 lados?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('O MSWord é um sistema operacional?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('A água do mar é salgada?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('A letra A é uma vogal?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('Usamos crase antes de substantivo masculino?');
  if Resposta='NÃO' then Inc(Nota);
```

```
  FazerPergunta('Em Sto A de Pádua temos CETEP?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('Um quadrado possui lados iguais?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('5 * 3 = 16?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('O ano tem 12 meses?');
  if Resposta='SIM' then Inc(Nota);
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;                          //Refaz a prova
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);  //Início
begin
  FazerProva;
  if Nota>=7 then Aprovado else Recuperacao;
end;

end.
```

2ª Fase

```
unit ProvaU;

//2ª Fase
//Relógio
//Botão DESISTIR
//Imagem
//Menus

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    BitBtn2: TBitBtn;
    Image1: TImage;
    Configurao1: TMenuItem;
    Sobre1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure BitBtn2MouseMove(Sender: TObject; Shift: TShiftState; X, Y:
Integer);
    procedure BitBtn2Click(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Nota: Integer;
  Resposta: String;

implementation

uses PerguntaU;

{$R *.dfm}

procedure FazerPergunta(Questao: String);        //Abre o formulário
begin
  Form2.Label1.Caption:=Questao;
```

```
  Form2.ShowModal;
end;

procedure FazerProva;
begin
  Form1.Hide;
  Nota:=0;
  FazerPergunta('1 + 1 é igual a 2?');
  if Resposta='SIM' then Inc(Nota);              //Verifica se a resposta está
correta ou não, se estiver, incrementa a nota
  FazerPergunta('Um triângulo tem 4 lados?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('O MSWord é um sistema operacional?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('A água do mar é salgada?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('A letra A é uma vogal?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('Usamos crase antes de substantivo masculino?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('Em Sto A de Pádua temos CETEP?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('Um quadrado possui lados iguais?');
  if Resposta='SIM' then Inc(Nota);
  FazerPergunta('5 * 3 = 16?');
  if Resposta='NÃO' then Inc(Nota);
  FazerPergunta('O ano tem 12 meses?');
  if Resposta='SIM' then Inc(Nota);
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;                           //Refaz a prova
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);  //Início
begin
  FazerProva;
  if Nota>=7 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);   //Mostra as horas, de um em um
segundo
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.BitBtn2MouseMove(Sender: TObject; Shift: TShiftState; X, Y:
Integer); //Botão desistir
```

```
begin
  if BitBtn2.Left=184 then BitBtn2.Left:=284 else BitBtn2.Left:=184;
end;

procedure TForm1.BitBtn2Click(Sender: TObject);
begin
  ShowMessage('Desiste não!');
end;

procedure TForm1.Sair1Click(Sender: TObject);    //Fecha o programa
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);  //Troca a imagem
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);  //Imagem normal
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject); //Imagem esticada
begin
  Image1.Stretch:=True;
end;

end.
```

3ª Fase

```
unit ProvaU;


//3ª Fase
//Rotinas para chamar Calculadora, Paint e WordPad
//Conceito de matriz:
        //Matriz é uma variável que possui várias posições, neste caso, 15
posições
        //Total de 15 perguntas e 15 respostas certas, cada pergunta tem sua
própria resposta

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Nota: Integer;
  Resposta: String;
  Perguntas, Respostas: array [1..15] of String; //As perguntas e respostas
agora estão em matrizes de 15 posições
```

```
implementation

uses PerguntaU, SobreU;

{$R *.dfm}

procedure FazerPergunta(Questao: String);        //Abre o formulário
begin
  Form2.Label1.Caption:=Questao;
  Form2.ShowModal;
end;

procedure FazerProva;
var
  X: Integer;
begin
  Form1.Hide;
  Nota:=0;
  for X:=1 to 15 do                              //Loop para fazer as 15
perguntas, da 1ª a 15ª
    begin
      FazerPergunta(Perguntas[X]);
      if Resposta=Respostas[X] then Inc(Nota);   //Verifica se a resposta está
correta ou não, se estiver, incrementa a nota
    end;
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;                           //Refaz a prova
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);  //Início
begin
  //Aqui carraga-se a matriz com as 15 perguntas, da 1ª a 15ª
  Perguntas[1]:='O CorelDraw trabalha com desenhos vetoriais?';
  Respostas[1]:='SIM';
  Perguntas[2]:='A ferrramenta Mão-Livre cria linhas e curvas?';
  Respostas[2]:='SIM';
  Perguntas[3]:='A ferramenta Elipse move os nós de um objeto?';
  Respostas[3]:='NÃO';
  Perguntas[4]:='Mudamos a cor do contorno de um objeto com o duplo clique
neste?';
  Respostas[4]:='NÃO';
  Perguntas[5]:='O tecla de atalho F4 corresponde ao Zoom?';
  Respostas[5]:='SIM';
  Perguntas[6]:='Para colocar um texto em um caminho, digitamos diretamente
neste caminho?';
```

```
  Respostas[6]:='SIM';
  Perguntas[7]:='O atalho da ferramenta Forma é a tecla F11?';
  Respostas[7]:='NÃO';
  Perguntas[8]:='Para apagar parte de um objeto usamos a ferramenta Faca?';
  Respostas[8]:= 'NÃO';
  Perguntas[9]:='Para trabalhar corretamente com linhas-guia devemos acionar o
botão "Alinhar pelas linhas-guia"?';
  Respostas[9]:='SIM';
  Perguntas[10]:='A tecla CTRL auxilia a criar objetos perfeitos?';
  Respostas[10]:='SIM';
  Perguntas[11]:='Com o efeito Gradiente colorimos um objeto selecionado?';
  Respostas[11]:='SIM';
  Perguntas[12]:='Para girar um objeto usamos a ferramenta Mistura Interativa?';
  Respostas[12]:='NÃO';
  Perguntas[13]:='O comando desfazer volta ações efetuadas?';
  Respostas[13]:='SIM';
  Perguntas[14]:='Um modelo de papel tipo A4 deitado é tipo paisagem?';
  Respostas[14]:='SIM';
  Perguntas[15]:='Usamos o Texto Artístico para digitar grandes quantidades de
texto?';
  Respostas[15]:='NÃO';
  //Fim da matriz
  FazerProva;
  if Nota>=7 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);   //Mostra as horas, de um em um
segundo
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);    //Fecha o programa
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);  //Troca a imagem
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);  //Imagem normal
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject); //Imagem esticada
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject); //Chama a calculadora do
Windows
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;

procedure TForm1.Paint1Click(Sender: TObject);    //Chama o Paint
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;
```

```
procedure TForm1.WordPad1Click(Sender: TObject); //Chama o WordPad
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;

procedure TForm1.Sobre1Click(Sender: TObject);   //Mostra o formulário SOBRE
begin
  Form3.ShowModal;
end;

end.
```

4ª Fase

```
unit ProvaU;

//4ª Fase
//Acerto das variáveis, para facilitar o entendimento
//Mudança nas perguntas:
        //Cada pergunta agora possui 3 opções de respostas (antes era SIM ou
NÃO)
        //Manteve a matriz de perguntas
        //Criou-se 3 matrizes com as opções, de 1 a 15, pois são 15 perguntas
        //A matriz de resposta não possui mais SIM ou NÃO, possui a opção
correta agora, A, B ou C.
//Retirado os botões de SIM e NÃO do Form2 e acrescentado 3 RadioButton, para
receber as opções

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Nota: Integer;
```

```
  Minha_Resposta: String;
  Perguntas, Resposta_Certa,Opcao_A,Opcao_B,Opcao_C: array [1..15] of String;
//As perguntas e respostas agora estão em matrizes de 15 posições, sendo que
existe 3 opções de respostas

implementation

uses PerguntaU, SobreU;

{$R *.dfm}

procedure FazerPergunta(Questao,Primeira_Opcao,Segunda_Opcao,Terceira_Opcao:
String); //Mostra a pergunta e 3 opções de resposta
begin
  Form2.Label1.Caption:=Questao;
  Form2.RadioButton1.Caption:=Primeira_Opcao;
  Form2.RadioButton2.Caption:=Segunda_Opcao;
  Form2.RadioButton3.Caption:=Terceira_Opcao;
  Form2.ShowModal;
end;

procedure FazerProva;
var
  X: Integer;
begin
  Form1.Hide;
  Nota:=0;
  for X:=1 to 15 do                              //As perguntas e respostas
agora estão em matrizes de 15 posições
    begin
      FazerPergunta(Perguntas[X],Opcao_A[X],Opcao_B[X],Opcao_C[X]);
      if Minha_Resposta=Resposta_Certa[X] then Inc(Nota);//Verifica se a
resposta está correta ou não, se estiver, incrementa a nota
    end;
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;                           //Refaz a prova
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);  //Início
begin
  //Aqui carraga-se a matriz com as 15 perguntas, da 1ª a 15ª
  //Para cada pergunta existem 3 opções de respostas
  //Para cada pergunta e 3 opções, existe a opção correta
  //Temos agora 5 matrizes, todas de 1 a 15, com pergunta, 3 opções e resposta
  Perguntas[1]:='O CorelDraw trabalha principalmente com?';
  Opcao_A[1]:='Desenhos vetoriais e gráficos.';
```

```
  Opcao_B[1]:='Imagens.';
  Opcao_C[1]:='Fotos.';
  Resposta_Certa[1]:='A';

  Perguntas[2]:='A ferrramenta Mão-Livre, cria rapidamente?';
  Opcao_A[2]:='Elipses perfeitas.';
  Opcao_B[2]:='Retângulos perfeitos.';
  Opcao_C[2]:='Linhas e curvas.';
  Resposta_Certa[2]:='C';

  Perguntas[3]:='Qual ferramenta é usada para criar, apagar ou mover os nós dos
objeto?';
  Opcao_A[3]:='Elipse.';
  Opcao_B[3]:='Retângulo.';
  Opcao_C[3]:='Forma.';
  Resposta_Certa[3]:='C';

  Perguntas[4]:='Como mudamos a cor do contorno de um objeto selecionado?';
  Opcao_A[4]:='Clicando com o botão esquerdo do mouse na cor desejada.';
  Opcao_B[4]:='Clicando com o botão direito do mouse na cor desejada.';
  Opcao_C[4]:='Efetuando um duplo clíque no objeto.';
  Resposta_Certa[4]:='B';

  Perguntas[5]:='Para que servem os atalhos CTRL+D, CTRL+G e F4,
respectivamente?';
  Opcao_A[5]:='Duplicar, agrupar e zoom para todos.';
  Opcao_B[5]:='Agrupar, duplicar e zoom para todos.';
  Opcao_C[5]:='Zoom para todos, agrupar e duplicar.';
  Resposta_Certa[5]:='A';

  Perguntas[6]:='Qual a maneira mais fácil de colocar um texto a um caminho?';
  Opcao_A[6]:='Utilizando a ferramenta Forma.';
  Opcao_B[6]:='Digitando o texto diretamente ao caminho.';
  Opcao_C[6]:='Agrupando o texto ao caminho.';
  Resposta_Certa[6]:='B';

  Perguntas[7]:='Qual o atalho da ferramenta Forma?';
  Opcao_A[7]:='F10.';
  Opcao_B[7]:='F11.';
  Opcao_C[7]:='F12.';
  Resposta_Certa[7]:='A';

  Perguntas[8]:='Para apagar parte de um objeto devemos utilizar a ferramenta?';
  Opcao_A[8]:='Faca.';
  Opcao_B[8]:='Borracha.';
  Opcao_C[8]:='Selecão.';
  Resposta_Certa[8]:='B';

  Perguntas[9]:='Para trabalhar corretamente com linhas-guia, faz-se necessário
acionar o botão "Alinhar pelas linhas-guia"?';
  Opcao_A[9]:='Sim.';
  Opcao_B[9]:='Não.';
  Opcao_C[9]:='Qualquer uma das alternativa é correta.';
  Resposta_Certa[9]:='A';

  Perguntas[10]:='Qual a tecla que auxilia a criacão de objetos perfeitos?';
  Opcao_A[10]:='SHIFT.';
  Opcao_B[10]:='CTRL.';
  Opcao_C[10]:='ENTER.';
  Resposta_Certa[10]:='B';

  Perguntas[11]:='Os efeitos de cor como preenchimento Gradiente e a Textura são
feitos com as ferramentas?';
  Opcao_A[11]:='Mistura interativa.';
```

```
  Opcao_B[11]:='Preenchimento interativo.';
  Opcao_C[11]:='Extrusão interativa.';
  Resposta_Certa[11]:='B';

  Perguntas[12]:='Para girar um objeto, usamos qual ferramenta para selecioná-
lo?';
  Opcao_A[12]:='Forma.';
  Opcao_B[12]:='Selecão.';
  Opcao_C[12]:='Mistura.';
  Resposta_Certa[12]:='B';

  Perguntas[13]:='O comando desfazer serve para?';
  Opcao_A[13]:='Criar novos objetos.';
  Opcao_B[13]:='Voltar as ações.';
  Opcao_C[13]:='Nehuma das opções.';
  Resposta_Certa[13]:='B';

  Perguntas[14]:='Quando quermos usar uma folha de papel modelo A4 deitada, por
exemplo, devemos mudar?';
  Opcao_A[14]:='A orientação da folha para retrato.';
  Opcao_B[14]:='A orientação da folha para paisagem.';
  Opcao_C[14]:='Nenhuma das opções.';
  Resposta_Certa[14]:='B';

  Perguntas[15]:='O tipo de texto de parágrafo e mais utilizado para?';
  Opcao_A[15]:='Digitar grandes quantidades de texto.';
  Opcao_B[15]:='Digitar pequenas quantidades de texto.';
  Opcao_C[15]:='Nenhuma das opções.';
  Resposta_Certa[15]:='A';
  //Fim da matriz

  FazerProva;
  if Nota>=7 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);   //Mostra as horas, de um em um
segundo
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);    //Fecha o programa
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);  //Troca a imagem
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);  //Imagem normal
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject); //Imagem esticada
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject); //Chama a calculadora do
Windows
```

```
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;

procedure TForm1.Paint1Click(Sender: TObject);    //Chama o Paint
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;

procedure TForm1.WordPad1Click(Sender: TObject); //Chama o WordPad
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;

procedure TForm1.Sobre1Click(Sender: TObject);   //Mostra o formulário SOBRE
begin
  Form3.ShowModal;
end;

end.
```

5ª Fase

```
unit ProvaU;


//5ª Fase
//Acrescentada a opção para escolher o número de perguntas, variável
Quant_Perguntas recebe o valor de SpinEdit1
//A nota para passar agora não é mais 7 e sim a fórmula 7*Quant_Perguntas/100,
que corresponde a 70% de acertos
//Melhoras no Form2

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs, Spin;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    SpinEdit1: TSpinEdit;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Minha_Resposta: String;
  Quant_Perguntas, Nota: Integer;
  Perguntas, Resposta_Certa,Opcao_A,Opcao_B,Opcao_C: array [1..15] of String;
```

```
implementation

uses PerguntaU, SobreU;

{$R *.dfm}

procedure FazerPergunta(Questao,Primeira_Opcao,Segunda_Opcao,Terceira_Opcao:
String);
begin
  Form2.Label1.Caption:=Questao;
  Form2.RadioButton1.Caption:=Primeira_Opcao;
  Form2.RadioButton2.Caption:=Segunda_Opcao;
  Form2.RadioButton3.Caption:=Terceira_Opcao;
  Form2.ShowModal;
end;

procedure FazerProva;
var
  X: Integer;
begin
  Form1.Hide;
  Nota:=0;
  for X:=1 to Quant_Perguntas do
    begin
      FazerPergunta(Perguntas[X],Opcao_A[X],Opcao_B[X],Opcao_C[X]);
      if Minha_Resposta=Resposta_Certa[X] then Inc(Nota);
    end;
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7*Quant_Perguntas/100 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  Quant_Perguntas:=SpinEdit1.Value;
  Perguntas[1]:='O CorelDraw trabalha principalmente com?';
  Opcao_A[1]:='Desenhos vetoriais e gráficos.';
  Opcao_B[1]:='Imagens.';
  Opcao_C[1]:='Fotos.';
  Resposta_Certa[1]:='A';

  Perguntas[2]:='A ferrramenta Mão-Livre, cria rapidamente?';
  Opcao_A[2]:='Elipses perfeitas.';
  Opcao_B[2]:='Retângulos perfeitos.';
  Opcao_C[2]:='Linhas e curvas.';
  Resposta_Certa[2]:='C';
```

```
  Perguntas[3]:='Qual ferramenta é usada para criar, apagar ou mover os nós dos
objeto?';
  Opcao_A[3]:='Elipse.';
  Opcao_B[3]:='Retângulo.';
  Opcao_C[3]:='Forma.';
  Resposta_Certa[3]:='C';

  Perguntas[4]:='Como mudamos a cor do contorno de um objeto selecionado?';
  Opcao_A[4]:='Clicando com o botão esquerdo do mouse na cor desejada.';
  Opcao_B[4]:='Clicando com o botão direito do mouse na cor desejada.';
  Opcao_C[4]:='Efetuando um duplo clíque no objeto.';
  Resposta_Certa[4]:='B';

  Perguntas[5]:='Para que servem os atalhos CTRL+D, CTRL+G e F4,
respectivamente?';
  Opcao_A[5]:='Duplicar, agrupar e zoom para todos.';
  Opcao_B[5]:='Agrupar, duplicar e zoom para todos.';
  Opcao_C[5]:='Zoom para todos, agrupar e duplicar.';
  Resposta_Certa[5]:='A';

  Perguntas[6]:='Qual a maneira mais fácil de colocar um texto a um caminho?';
  Opcao_A[6]:='Utilizando a ferramenta Forma.';
  Opcao_B[6]:='Digitando o texto diretamente ao caminho.';
  Opcao_C[6]:='Agrupando o texto ao caminho.';
  Resposta_Certa[6]:='B';

  Perguntas[7]:='Qual o atalho da ferramenta Forma?';
  Opcao_A[7]:='F10.';
  Opcao_B[7]:='F11.';
  Opcao_C[7]:='F12.';
  Resposta_Certa[7]:='A';

  Perguntas[8]:='Para apagar parte de um objeto devemos utilizar a ferramenta?';
  Opcao_A[8]:='Faca.';
  Opcao_B[8]:='Borracha.';
  Opcao_C[8]:='Selecão.';
  Resposta_Certa[8]:='B';

  Perguntas[9]:='Para trabalhar corretamente com linhas-guia, faz-se necessário
acionar o botão "Alinhar pelas linhas-guia"?';
  Opcao_A[9]:='Sim.';
  Opcao_B[9]:='Não.';
  Opcao_C[9]:='Qualquer uma das alternativa é correta.';
  Resposta_Certa[9]:='A';

  Perguntas[10]:='Qual a tecla que auxilia a criacão de objetos perfeitos?';
  Opcao_A[10]:='SHIFT.';
  Opcao_B[10]:='CTRL.';
  Opcao_C[10]:='ENTER.';
  Resposta_Certa[10]:='B';

  Perguntas[11]:='Os efeitos de cor como preenchimento Gradiente e a Textura são
feitos com as ferramentas?';
  Opcao_A[11]:='Mistura interativa.';
  Opcao_B[11]:='Preenchimento interativo.';
  Opcao_C[11]:='Extrusão interativa.';
  Resposta_Certa[11]:='B';

  Perguntas[12]:='Para girar um objeto, usamos qual ferramenta para selecioná-
lo?';
  Opcao_A[12]:='Forma.';
  Opcao_B[12]:='Selecão.';
  Opcao_C[12]:='Mistura.';
  Resposta_Certa[12]:='B';
```

```
  Perguntas[13]:='O comando desfazer serve para?';
  Opcao_A[13]:='Criar novos objetos.';
  Opcao_B[13]:='Voltar as ações.';
  Opcao_C[13]:='Nehuma das opções.';
  Resposta_Certa[13]:='B';

  Perguntas[14]:='Quando quermos usar uma folha de papel modelo A4 deitada, por
exemplo, devemos mudar?';
  Opcao_A[14]:='A orientação da folha para retrato.';
  Opcao_B[14]:='A orientação da folha para paisagem.';
  Opcao_C[14]:='Nenhuma das opções.';
  Resposta_Certa[14]:='B';

  Perguntas[15]:='O tipo de texto de parágrafo e mais utilizado para?';
  Opcao_A[15]:='Digitar grandes quantidades de texto.';
  Opcao_B[15]:='Digitar pequenas quantidades de texto.';
  Opcao_C[15]:='Nenhuma das opções.';
  Resposta_Certa[15]:='A';

  FazerProva;
  if Nota>=70*Quant_Perguntas/100 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject);
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject);
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;

procedure TForm1.Paint1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;

procedure TForm1.WordPad1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;
```

```
procedure TForm1.Sobre1Click(Sender: TObject);
begin
  Form3.ShowModal;
end;

end.
```

6ª Fase

```
unit ProvaU;

//6ª Fase – Fase não necessária
//Acrescentado 2 ProgressBar no Form2, uma para as perguntas e outro para o
tempo de prova

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs, Spin;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    SpinEdit1: TSpinEdit;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Minha_Resposta: String;
  Tempo_Total, Quant_Perguntas, Nota: Integer;
  Perguntas, Resposta_Certa,Opcao_A,Opcao_B,Opcao_C: array [1..15] of String;

implementation

uses PerguntaU, SobreU;
```

```
{$R *.dfm}

procedure FazerPergunta(Questao,Primeira_Opcao,Segunda_Opcao,Terceira_Opcao:
String);
begin
  Form2.Label1.Caption:=Questao;
  Form2.RadioButton1.Caption:=Primeira_Opcao;
  Form2.RadioButton2.Caption:=Segunda_Opcao;
  Form2.RadioButton3.Caption:=Terceira_Opcao;
  Form2.ShowModal;
end;

procedure FazerProva;
var
  X: Integer;
begin
  Form1.Hide;
  Nota:=0;
  Form2.Timer1.Enabled:=True;                    //Ativa o relógio para contar o
tempo de prova
  Form2.ProgressBar1.Position:=0;                //A contagem começa em zero
  Form2.ProgressBar1.Max:=Tempo_Total;           //E termina no Tempo_Total
  Form2.ProgressBar2.Position:=0;                //As perguntas começam em zero
  Form2.ProgressBar2.Max:=Quant_Perguntas;       //E terminam na Quant_Perguntas
  for X:=1 to Quant_Perguntas do
    begin
      Form2.ProgressBar2.Position:=X;            //A cada pergunta, aumenta uma
posição
      if Form2.ProgressBar1.Position>=Tempo_Total then Break; //Sai do Loop se o
tempo acabou
      FazerPergunta(Perguntas[X],Opcao_A[X],Opcao_B[X],Opcao_C[X]);
      if Minha_Resposta=Resposta_Certa[X] then Inc(Nota);
    end;
  Form2.Timer1.Enabled:=False;                   //Desliga o relógio para contar
o tempo de prova, pois a prova acabou
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7*Quant_Perguntas/100 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  Quant_Perguntas:=SpinEdit1.Value;
  Tempo_Total:=Quant_Perguntas*10;               //Para cada pergunta, a aluno
possui 10 segundos: 5 perguntas = 50 segundos, 10 perguntas = 100 segundos, etc.
  Perguntas[1]:='O CorelDraw trabalha principalmente com?';
  Opcao_A[1]:='Desenhos vetoriais e gráficos.';
```

```
  Opcao_B[1]:='Imagens.';
  Opcao_C[1]:='Fotos.';
  Resposta_Certa[1]:='A';

  Perguntas[2]:='A ferrramenta Mão-Livre, cria rapidamente?';
  Opcao_A[2]:='Elipses perfeitas.';
  Opcao_B[2]:='Retângulos perfeitos.';
  Opcao_C[2]:='Linhas e curvas.';
  Resposta_Certa[2]:='C';

  Perguntas[3]:='Qual ferramenta é usada para criar, apagar ou mover os nós dos
objeto?';
  Opcao_A[3]:='Elipse.';
  Opcao_B[3]:='Retângulo.';
  Opcao_C[3]:='Forma.';
  Resposta_Certa[3]:='C';

  Perguntas[4]:='Como mudamos a cor do contorno de um objeto selecionado?';
  Opcao_A[4]:='Clicando com o botão esquerdo do mouse na cor desejada.';
  Opcao_B[4]:='Clicando com o botão direito do mouse na cor desejada.';
  Opcao_C[4]:='Efetuando um duplo clíque no objeto.';
  Resposta_Certa[4]:='B';

  Perguntas[5]:='Para que servem os atalhos CTRL+D, CTRL+G e F4,
respectivamente?';
  Opcao_A[5]:='Duplicar, agrupar e zoom para todos.';
  Opcao_B[5]:='Agrupar, duplicar e zoom para todos.';
  Opcao_C[5]:='Zoom para todos, agrupar e duplicar.';
  Resposta_Certa[5]:='A';

  Perguntas[6]:='Qual a maneira mais fácil de colocar um texto a um caminho?';
  Opcao_A[6]:='Utilizando a ferramenta Forma.';
  Opcao_B[6]:='Digitando o texto diretamente ao caminho.';
  Opcao_C[6]:='Agrupando o texto ao caminho.';
  Resposta_Certa[6]:='B';

  Perguntas[7]:='Qual o atalho da ferramenta Forma?';
  Opcao_A[7]:='F10.';
  Opcao_B[7]:='F11.';
  Opcao_C[7]:='F12.';
  Resposta_Certa[7]:='A';

  Perguntas[8]:='Para apagar parte de um objeto devemos utilizar a ferramenta?';
  Opcao_A[8]:='Faca.';
  Opcao_B[8]:='Borracha.';
  Opcao_C[8]:='Selecão.';
  Resposta_Certa[8]:='B';

  Perguntas[9]:='Para trabalhar corretamente com linhas-guia, faz-se necessário
acionar o botão "Alinhar pelas linhas-guia"?';
  Opcao_A[9]:='Sim.';
  Opcao_B[9]:='Não.';
  Opcao_C[9]:='Qualquer uma das alternativa é correta.';
  Resposta_Certa[9]:='A';

  Perguntas[10]:='Qual a tecla que auxilia a criacão de objetos perfeitos?';
  Opcao_A[10]:='SHIFT.';
  Opcao_B[10]:='CTRL.';
  Opcao_C[10]:='ENTER.';
  Resposta_Certa[10]:='B';

  Perguntas[11]:='Os efeitos de cor como preenchimento Gradiente e a Textura são
feitos com as ferramentas?';
  Opcao_A[11]:='Mistura interativa.';
```

```
  Opcao_B[11]:='Preenchimento interativo.';
  Opcao_C[11]:='Extrusão interativa.';
  Resposta_Certa[11]:='B';

  Perguntas[12]:='Para girar um objeto, usamos qual ferramenta para selecioná-
lo?';
  Opcao_A[12]:='Forma.';
  Opcao_B[12]:='Selecão.';
  Opcao_C[12]:='Mistura.';
  Resposta_Certa[12]:='B';

  Perguntas[13]:='O comando desfazer serve para?';
  Opcao_A[13]:='Criar novos objetos.';
  Opcao_B[13]:='Voltar as ações.';
  Opcao_C[13]:='Nehuma das opções.';
  Resposta_Certa[13]:='B';

  Perguntas[14]:='Quando quermos usar uma folha de papel modelo A4 deitada, por
exemplo, devemos mudar?';
  Opcao_A[14]:='A orientação da folha para retrato.';
  Opcao_B[14]:='A orientação da folha para paisagem.';
  Opcao_C[14]:='Nenhuma das opções.';
  Resposta_Certa[14]:='B';

  Perguntas[15]:='O tipo de texto de parágrafo e mais utilizado para?';
  Opcao_A[15]:='Digitar grandes quantidades de texto.';
  Opcao_B[15]:='Digitar pequenas quantidades de texto.';
  Opcao_C[15]:='Nenhuma das opções.';
  Resposta_Certa[15]:='A';

  FazerProva;
  if Nota>=70*Quant_Perguntas/100 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject);
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject);
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;
```

```
procedure TForm1.Paint1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;

procedure TForm1.WordPad1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;

procedure TForm1.Sobre1Click(Sender: TObject);
begin
  Form3.ShowModal;
end;

end.
```

7ª Fase

```
unit ProvaU;

//7ª Fase
//Acrescentada a rotina Efetuar_Sorteio

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs, Spin;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    SpinEdit1: TSpinEdit;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Minha_Resposta: String;
  Tempo_Total, Quant_Perguntas, Nota: Integer;
  Perguntas, Resposta_Certa,Opcao_A,Opcao_B,Opcao_C: array [1..15] of String;
  Sorteio: array [1..15] of Integer;


implementation

uses PerguntaU, SobreU;
```

```
{$R *.dfm}

procedure FazerPergunta(Questao,Primeira_Opcao,Segunda_Opcao,Terceira_Opcao:
String);
begin
  Form2.Label1.Caption:=Questao;
  Form2.RadioButton1.Caption:=Primeira_Opcao;
  Form2.RadioButton2.Caption:=Segunda_Opcao;
  Form2.RadioButton3.Caption:=Terceira_Opcao;
  Form2.ShowModal;
end;

//Esta rotina gera uma matriz de 15 posições, contendo os números de 1 a 15
//Cada número corresponde a uma pergunta, o número 7 é a 7ª pergunta, etc.
//Não é permitida a repetição deste números, para não repetir perguntas na mesma
prova
procedure Efetuar_Sorteio;
var
  Sair: String;
  X,Y,Z: Integer;
begin
  Randomize;                                    //Libera o Random
  for X:= 1 to 15 do Sorteio[X]:=0;             //Zera a matriz para a
recuperação, pois existe novo sorteio
  for X:= 1 to 15 do                            //São 15 perguntas
    begin
      //Este loop permite que a matriz tenha os números de 1 a 15 sem repetição
      repeat                                    //Loop de Repeat a Until
        Sair:='Sim';                            //Variável auxiliar, começa com
SIM
        Y:=Random(15)+1;                        //Gera um número aleatório
entre 0 e 14 (15 primeiros números inteiros) e soma um a este (1 a 15, neste
caso)
        for Z:= 1 to 15 do if Y= Sorteio[Z] then Sair:= 'Não'; //Se existe Y na
matriz Sorteio, então não sai e sorteia de novo, isto é, se um número já saiu,
sorteia de novo. Informa que NÃO pode SAIR do loop
      until Sair= 'Sim';                        //Só sai do loop, se SAIR for
SIM
      Sorteio[X]:=Y;                            //A matriz SORTEIO, recebe na
posição X (de 1 a 15), um número aleatório, que não saiu antes
    end;
  Sleep(1000);  //Pausa de um segundo. Esta linha não é necessária, mas foi
colocada aqui para permitir colocar um BreakPoint para estudar a matriz SORTEIO.
Coloque um BreakPoint nela (clicando na barra cinza a esquerda) e analise.
end;

procedure FazerProva;
var
  X,Y: Integer;
begin
  Form1.Hide;
  Nota:=0;
  Efetuar_Sorteio;                              //Rotina para gerar 15 números
aleatórios sem repetição, sempre que uma prova for feita. Dica: Presione CTRL e
clique em Efetuar_Sorteio, nesta linha
  Form2.Timer1.Enabled:=True;
  Form2.ProgressBar1.Position:=0;
  Form2.ProgressBar1.Max:=Tempo_Total;
  Form2.ProgressBar2.Position:=0;
  Form2.ProgressBar2.Max:=Quant_Perguntas;
  for X:=1 to Quant_Perguntas do
    begin
      Form2.ProgressBar2.Position:=X;
```

```
      if Form2.ProgressBar1.Position>=Tempo_Total then Break;
      Y:=Sorteio[X];
      FazerPergunta(Perguntas[Y],Opcao_A[Y],Opcao_B[Y],Opcao_C[Y]);
      if Minha_Resposta=Resposta_Certa[Y] then Inc(Nota);
    end;
  Form2.Timer1.Enabled:=False;
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7*Quant_Perguntas/100 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  Quant_Perguntas:=SpinEdit1.Value;
  Tempo_Total:=Quant_Perguntas*10;
  Perguntas[1]:='O CorelDraw trabalha principalmente com?';
  Opcao_A[1]:='Desenhos vetoriais e gráficos.';
  Opcao_B[1]:='Imagens.';
  Opcao_C[1]:='Fotos.';
  Resposta_Certa[1]:='A';

  Perguntas[2]:='A ferrramenta Mão-Livre, cria rapidamente?';
  Opcao_A[2]:='Elipses perfeitas.';
  Opcao_B[2]:='Retângulos perfeitos.';
  Opcao_C[2]:='Linhas e curvas.';
  Resposta_Certa[2]:='C';

  Perguntas[3]:='Qual ferramenta é usada para criar, apagar ou mover os nós dos
objeto?';
  Opcao_A[3]:='Elipse.';
  Opcao_B[3]:='Retângulo.';
  Opcao_C[3]:='Forma.';
  Resposta_Certa[3]:='C';

  Perguntas[4]:='Como mudamos a cor do contorno de um objeto selecionado?';
  Opcao_A[4]:='Clicando com o botão esquerdo do mouse na cor desejada.';
  Opcao_B[4]:='Clicando com o botão direito do mouse na cor desejada.';
  Opcao_C[4]:='Efetuando um duplo clíque no objeto.';
  Resposta_Certa[4]:='B';

  Perguntas[5]:='Para que servem os atalhos CTRL+D, CTRL+G e F4,
respectivamente?';
  Opcao_A[5]:='Duplicar, agrupar e zoom para todos.';
  Opcao_B[5]:='Agrupar, duplicar e zoom para todos.';
  Opcao_C[5]:='Zoom para todos, agrupar e duplicar.';
  Resposta_Certa[5]:='A';
```

```
  Perguntas[6]:='Qual a maneira mais fácil de colocar um texto a um caminho?';
  Opcao_A[6]:='Utilizando a ferramenta Forma.';
  Opcao_B[6]:='Digitando o texto diretamente ao caminho.';
  Opcao_C[6]:='Agrupando o texto ao caminho.';
  Resposta_Certa[6]:='B';

  Perguntas[7]:='Qual o atalho da ferramenta Forma?';
  Opcao_A[7]:='F10.';
  Opcao_B[7]:='F11.';
  Opcao_C[7]:='F12.';
  Resposta_Certa[7]:='A';

  Perguntas[8]:='Para apagar parte de um objeto devemos utilizar a ferramenta?';
  Opcao_A[8]:='Faca.';
  Opcao_B[8]:='Borracha.';
  Opcao_C[8]:='Selecão.';
  Resposta_Certa[8]:='B';

  Perguntas[9]:='Para trabalhar corretamente com linhas-guia, faz-se necessário
acionar o botão "Alinhar pelas linhas-guia"?';
  Opcao_A[9]:='Sim.';
  Opcao_B[9]:='Não.';
  Opcao_C[9]:='Qualquer uma das alternativa é correta.';
  Resposta_Certa[9]:='A';

  Perguntas[10]:='Qual a tecla que auxilia a criacão de objetos perfeitos?';
  Opcao_A[10]:='SHIFT.';
  Opcao_B[10]:='CTRL.';
  Opcao_C[10]:='ENTER.';
  Resposta_Certa[10]:='B';

  Perguntas[11]:='Os efeitos de cor como preenchimento Gradiente e a Textura são
feitos com as ferramentas?';
  Opcao_A[11]:='Mistura interativa.';
  Opcao_B[11]:='Preenchimento interativo.';
  Opcao_C[11]:='Extrusão interativa.';
  Resposta_Certa[11]:='B';

  Perguntas[12]:='Para girar um objeto, usamos qual ferramenta para selecioná-
lo?';
  Opcao_A[12]:='Forma.';
  Opcao_B[12]:='Selecão.';
  Opcao_C[12]:='Mistura.';
  Resposta_Certa[12]:='B';

  Perguntas[13]:='O comando desfazer serve para?';
  Opcao_A[13]:='Criar novos objetos.';
  Opcao_B[13]:='Voltar as ações.';
  Opcao_C[13]:='Nehuma das opções.';
  Resposta_Certa[13]:='B';

  Perguntas[14]:='Quando quermos usar uma folha de papel modelo A4 deitada, por
exemplo, devemos mudar?';
  Opcao_A[14]:='A orientação da folha para retrato.';
  Opcao_B[14]:='A orientação da folha para paisagem.';
  Opcao_C[14]:='Nenhuma das opções.';
  Resposta_Certa[14]:='B';

  Perguntas[15]:='O tipo de texto de parágrafo e mais utilizado para?';
  Opcao_A[15]:='Digitar grandes quantidades de texto.';
  Opcao_B[15]:='Digitar pequenas quantidades de texto.';
  Opcao_C[15]:='Nenhuma das opções.';
  Resposta_Certa[15]:='A';
```

```
  FazerProva;
  if Nota>=70*Quant_Perguntas/100 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject);
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject);
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;

procedure TForm1.Paint1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;

procedure TForm1.WordPad1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;

procedure TForm1.Sobre1Click(Sender: TObject);
begin
  Form3.ShowModal;
end;

end.
```

8ª Fase

```
unit ProvaU;

//8ª Fase
//Uso de banco de dados, eliminando as matrizes de perguntas, opções e respostas
//O banco de dados é vinculado ao componente Table, no Form4
//Procedimentos:
//Após criar o banco de dados no Database Desktop, use um componente Table,
alterando no sua propriedade DatabaseName para o diretório da aplicação (se esta
estiver no mesmo diretório do banco de dados, não é necessário). Em TableName,
selecione o banco de dados;
//Efetue um duplo clique no Table e com o botão direito Add All Fields;
//Acrescente um componente DataSource e vincule este ao Table em DataSet;
//Acrescente um DBGrid e vincule este ao DataSourse.
//Pode-se também após efetuar um duplo clique no Table, arrastar todos os campos
para o formulário, criando os DBEdit e DataSource automaticamente.

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs, Spin;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    SpinEdit1: TSpinEdit;
    Editor1: TMenuItem;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
    procedure Editor1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
```

```
  end;

var
  Form1: TForm1;
  Minha_Resposta: String;
  Tempo_Total, Quant_Perguntas, Nota: Integer;
  Sorteio: array [1..15] of Integer;

implementation

uses PerguntaU, SobreU, EditorU;

{$R *.dfm}

procedure FazerPergunta(Questao,Primeira_Opcao,Segunda_Opcao,Terceira_Opcao:
String);
begin
  Form2.Label1.Caption:=Questao;
  Form2.RadioButton1.Caption:=Primeira_Opcao;
  Form2.RadioButton2.Caption:=Segunda_Opcao;
  Form2.RadioButton3.Caption:=Terceira_Opcao;
  Form2.ShowModal;
end;

//Esta rotina gera uma matriz de 15 posições, contendo os números de 1 a 15
//Cada número corresponde a uma pergunta, o número 7 é a 7ª pergunta, etc.
//Não é permitida a repetição deste números, para não repetir perguntas na mesma
prova
procedure Efetuar_Sorteio;
var
  Sair: String;
  X,Y,Z: Integer;
begin
  Randomize;                                    //Libera o Random
  for X:= 1 to 15 do Sorteio[X]:=0;            //Zera a matriz para a
recuperação, pois existe novo sorteio
  for X:= 1 to 15 do                            //São 15 perguntas
    begin
      //Este loop permite que a matriz tenha os números de 1 a 15 sem repetição
      repeat                                    //Loop de Repeat a Until
        Sair:='Sim';                            //Variável auxiliar, começa com
SIM
        Y:=Random(15)+1;                        //Gera um número aleatório
entre 0 e 14 (15 primeiros números inteiros) e soma um a este (1 a 15, neste
caso)
        for Z:=1 to 15 do if Y= Sorteio[Z] then Sair:= 'Não'; //Se existe Y na
matriz Sorteio, então não sai e sorteia de novo, isto é, se um número já saiu,
sorteia de novo. Informa que NÃO pode SAIR do loop
      until Sair= 'Sim';                        //Só sai do loop, se SAIR for
SIM
      Sorteio[X]:=Y;                            //A matriz SORTEIO, recebe na
posição X (de 1 a 15), um número aleatório, que não saiu antes
    end;
  Sleep(1000);  //Pausa de um segundo. Esta linha não é necessária, mas foi
colocada aqui para permitir colocar um BreakPoint para estudar a matriz SORTEIO.
Coloque um BreakPoint nela (clicando na barra cinza a esquerda) e analise.
end;

procedure FazerProva;
var
  X,Y,Z: Integer;
begin
  Form1.Hide;
  Nota:=0;
```

```
  Efetuar_Sorteio;                                //Rotina para gerar 15 números
aleatórios sem repetição, sempre que uma prova for feita. Dica: Presione CTRL e
clique em Efetuar_Sorteio, nesta linha
  Form2.Timer1.Enabled:=True;
  Form2.ProgressBar1.Position:=0;
  Form2.ProgressBar1.Max:=Tempo_Total;
  Form2.ProgressBar2.Position:=0;
  Form2.ProgressBar2.Max:=Quant_Perguntas;
  for X:=1 to Quant_Perguntas do
    begin
      Form2.ProgressBar2.Position:=X;
      if Form2.ProgressBar1.Position>=Tempo_Total then Break;
      Y:=Sorteio[X];
      Form4.Table1.First;                          //Pula para a 1ª pergunta
      for Z:=1 to Y-1 do Form4.Table1.Next;        //Pula para a pergunta Y-1
(não contar a 1ª pergunta, pois já estamos nela), que é a matriz SORTEIO

FazerPergunta(Form4.Table1Pergunta.AsString,Form4.Table1OpcaoA.AsString,Form4.Ta
ble1OpcaoB.AsString,Form4.Table1OpcaoC.AsString);
      if Minha_Resposta=Form4.Table1Resposta.AsString  then Inc(Nota);
    end;
  Form2.Timer1.Enabled:=False;
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7*Quant_Perguntas/100 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  Quant_Perguntas:=SpinEdit1.Value;
  Tempo_Total:=Quant_Perguntas*10;
  FazerProva;
  if Nota>=70*Quant_Perguntas/100 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);
begin
```

```
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject);
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject);
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;

procedure TForm1.Paint1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;

procedure TForm1.WordPad1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;

procedure TForm1.Sobre1Click(Sender: TObject);
begin
  Form3.ShowModal;
end;

procedure TForm1.Editor1Click(Sender: TObject);
begin
  Form4.Show;
end;

end.
```

9ª Fase

```
unit ProvaU;

//9ª Fase
//Troca da matriz SORTEIO estática por dinâmica, isto é, sem tamanho fixo
//Esta matriz ocasiona um problema, ela não começa em 1 e sim em 0, necessitando
alguns ajustes.
//Exemplo: a matriz de 15 posições seria de 0 a 14
//Liberado o total de perguntas

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs, Spin;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    SpinEdit1: TSpinEdit;
    Editor1: TMenuItem;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
    procedure Editor1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  Minha_Resposta: String;
  Tempo_Total, Quant_Perguntas, Nota, Total_Perguntas: Integer;
  Sorteio: array of Integer;
```

```
implementation

uses PerguntaU, SobreU, EditorU;

{$R *.dfm}

procedure FazerPergunta(Questao,Primeira_Opcao,Segunda_Opcao,Terceira_Opcao:
String);
begin
  Form2.Label1.Caption:=Questao;
  Form2.RadioButton1.Caption:=Primeira_Opcao;
  Form2.RadioButton2.Caption:=Segunda_Opcao;
  Form2.RadioButton3.Caption:=Terceira_Opcao;
  Form2.ShowModal;
end;

//Esta rotina gera uma matriz de 15 posições, contendo os números de 1 a 15
//Cada número corresponde a uma pergunta, o número 7 é a 7ª pergunta, etc.
//Não é permitida a repetição deste números, para não repetir perguntas na mesma
prova
procedure Efetuar_Sorteio;
var
  Sair: String;
  X,Y,Z: Integer;
begin
  Randomize;                                    //Libera o Random
  for X:=0 to Total_Perguntas-1 do Sorteio[X]:=0; //Zera a matriz para a
recuperação, pois existe novo sorteio
  for X:=0 to Total_Perguntas-1 do               //São 15 perguntas
    begin
      repeat                                    //Loop de Repeat a Until
        Sair:='Sim';                            //Variável auxiliar, começa com
SIM
        Y:=Random(Total_Perguntas)+1;            //Gera um número aleatório
entre 1 e Total_Pergutas
        for Z:=0 to Total_Perguntas-1 do if Y= Sorteio[Z] then Sair:= 'Não';
//Se existe Y na matriz Sorteio, então não sai e sorteia de novo, isto é, se um
número já saiu, sorteia de novo. Informa que NÃO pode SAIR do loop
      until Sair= 'Sim';                        //Só sai do loop, se SAIR for
SIM
      Sorteio[X]:=Y;                            //A matriz SORTEIO, recebe na
posição X (de 1 a 15), um número aleatório, que não saiu antes
    end;
  Sleep(1000);  //Pausa de um segundo. Esta linha não é necessária, mas foi
colocada aqui para permitir colocar um BreakPoint para estudar a matriz SORTEIO.
Coloque um BreakPoint nela (clicando na barra cinza a esquerda) e analise.
end;

procedure FazerProva;
var
  X,Y,Z: Integer;
begin
  Form1.Hide;
  Nota:=0;
  Efetuar_Sorteio;                              //Rotina para gerar números
aleatórios sem repetição, sempre que uma prova for feita. Dica: Presione CTRL e
clique em Efetuar_Sorteio, nesta linha
  Form2.Timer1.Enabled:=True;
  Form2.ProgressBar1.Position:=0;
  Form2.ProgressBar1.Max:=Tempo_Total;
  Form2.ProgressBar2.Position:=0;
  Form2.ProgressBar2.Max:=Quant_Perguntas;
  for X:=0 to Quant_Perguntas-1 do
```

```
    begin
      Form2.ProgressBar2.Position:=X;
      if Form2.ProgressBar1.Position>=Tempo_Total then Break;
      Y:=Sorteio[X];
      Form4.Table1.First;                          //Pula para a 1ª pergunta
      for Z:=1 to Y-1 do Form4.Table1.Next;        //Pula para a pergunta Y-1
(não contar a 1ª pergunta, pois já estamos nela), que é a matriz SORTEIO

FazerPergunta(Form4.Table1Pergunta.AsString,Form4.Table1OpcaoA.AsString,Form4.Ta
ble1OpcaoB.AsString,Form4.Table1OpcaoC.AsString);
      if Minha_Resposta=Form4.Table1Resposta.AsString  then Inc(Nota);
    end;
  Form2.Timer1.Enabled:=False;
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7*Quant_Perguntas/100 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  Total_Perguntas:=Form4.Table1.RecordCount;
  SetLength(Sorteio,Total_Perguntas);
  Quant_Perguntas:=SpinEdit1.Value;
  Tempo_Total:=Quant_Perguntas*10;
  FazerProva;
  if Nota>=70*Quant_Perguntas/100 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);
begin
  Application.Terminate;
end;

procedure TForm1.Trocar1Click(Sender: TObject);
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);
begin
  Image1.Stretch:=False;
```

```
end;

procedure TForm1.Esticar1Click(Sender: TObject);
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject);
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;

procedure TForm1.Paint1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;

procedure TForm1.WordPad1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;

procedure TForm1.Sobre1Click(Sender: TObject);
begin
  Form3.ShowModal;
end;

procedure TForm1.Editor1Click(Sender: TObject);
begin
  Form4.Show;
end;

end.
```

10ª e 11ª Fase

```
unit ProvaU;

//10ª Fase
//Inclusão de mais 2 opções nas respostas (D e E) e matéria, no Banco de Dados e
ajustes necessários: mais 2 RaddioButtons no Form2 e FazerPergunta.
//Inclusão de um ComboBox no Form1 para selecionar e filtrar a matéria.
//Inclusão de botões para inclusão de matérias...

//11ª Fase
//DBGrid retirado do Form4 e colocado no novo Form5.
//No Form4 efetuado duplo clique no Table, selecionado todos os campos (fields)
e arratando-os para o formulário. Inclusão de um DBNavigator vinculado ao
DataSource.

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ComCtrls, ExtCtrls, Buttons, ExtDlgs, Spin;

type
  TForm1 = class(TForm)
    Button1: TButton;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Sair1: TMenuItem;
    Timer1: TTimer;
    Image1: TImage;
    Configurao1: TMenuItem;
    OpenPictureDialog1: TOpenPictureDialog;
    Imagem: TMenuItem;
    Trocar1: TMenuItem;
    Normal1: TMenuItem;
    Esticar1: TMenuItem;
    Salvar1: TMenuItem;
    Ferramentas1: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    Calculadora1: TMenuItem;
    Paint1: TMenuItem;
    WordPad1: TMenuItem;
    SpinEdit1: TSpinEdit;
    Editor1: TMenuItem;
    ComboBox1: TComboBox;
    procedure Button1Click(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Trocar1Click(Sender: TObject);
    procedure Normal1Click(Sender: TObject);
    procedure Esticar1Click(Sender: TObject);
    procedure Calculadora1Click(Sender: TObject);
    procedure Paint1Click(Sender: TObject);
    procedure WordPad1Click(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
    procedure Editor1Click(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
```

```
  end;

var
  Form1: TForm1;
  Minha_Resposta: String;
  Tempo_Total, Quant_Perguntas, Nota, Total_Perguntas: Integer;
  Sorteio: array of Integer;

implementation

uses PerguntaU, SobreU, EditorU;

{$R *.dfm}

procedure
FazerPergunta(Questao,Primeira_Opcao,Segunda_Opcao,Terceira_Opcao,Quarta_Opcao,Q
uinta_Opcao: String);
begin
  Form2.Label1.Caption:=Questao;
  Form2.RadioButton1.Caption:=Primeira_Opcao;
  Form2.RadioButton2.Caption:=Segunda_Opcao;
  Form2.RadioButton3.Caption:=Terceira_Opcao;
  Form2.RadioButton4.Caption:=Quarta_Opcao;
  Form2.RadioButton5.Caption:=Quinta_Opcao;
  Form2.ShowModal;
end;

//Esta rotina gera uma matriz de 15 posições, contendo os números de 1 a 15
//Cada número corresponde a uma pergunta, o número 7 é a 7ª pergunta, etc.
//Não é permitida a repetição deste números, para não repetir perguntas na mesma
prova
procedure Efetuar_Sorteio;
var
  Sair: String;
  X,Y,Z: Integer;
begin
  Randomize;                                    //Libera o Random
  for X:=0 to Total_Perguntas-1 do Sorteio[X]:=0; //Zera a matriz para a
recuperação, pois existe novo sorteio
  for X:=0 to Total_Perguntas-1 do               //São 15 perguntas
    begin
      repeat                                    //Loop de Repeat a Until
        Sair:='Sim';                            //Variável auxiliar, começa com
SIM
        Y:=Random(Total_Perguntas)+1;            //Gera um número aleatório
entre 1 e Total_Pergutas
        for Z:=0 to Total_Perguntas-1 do if Y= Sorteio[Z] then Sair:= 'Não';
//Se existe Y na matriz Sorteio, então não sai e sorteia de novo, isto é, se um
número já saiu, sorteia de novo. Informa que NÃO pode SAIR do loop
      until Sair= 'Sim';                        //Só sai do loop, se SAIR for
SIM
      Sorteio[X]:=Y;                            //A matriz SORTEIO, recebe na
posição X (de 1 a 15), um número aleatório, que não saiu antes
    end;
  Sleep(1000);  //Pausa de um segundo. Esta linha não é necessária, mas foi
colocada aqui para permitir colocar um BreakPoint para estudar a matriz SORTEIO.
Coloque um BreakPoint nela (clicando na barra cinza a esquerda) e analise.
end;

procedure FazerProva;
var
  X,Y,Z: Integer;
begin
  Form1.Hide;
```

```
  Nota:=0;
  Efetuar_Sorteio;                              //Rotina para gerar números
aleatórios sem repetição, sempre que uma prova for feita. Dica: Presione CTRL e
clique em Efetuar_Sorteio, nesta linha
  Form2.Timer1.Enabled:=True;
  Form2.ProgressBar1.Position:=0;
  Form2.ProgressBar1.Max:=Tempo_Total;
  Form2.ProgressBar2.Position:=0;
  Form2.ProgressBar2.Max:=Quant_Perguntas;
  for X:=0 to Quant_Perguntas-1 do
    begin
      Form2.ProgressBar2.Position:=X;
      if Form2.ProgressBar1.Position>=Tempo_Total then Break;
      Y:=Sorteio[X];
      Form4.Table1.First;                        //Pula para a 1ª pergunta
      for Z:=1 to Y-1 do Form4.Table1.Next;      //Pula para a pergunta Y-1
(não contar a 1ª pergunta, pois já estamos nela), que é a matriz SORTEIO

FazerPergunta(Form4.Table1Pergunta.AsString,Form4.Table1OpcaoA.AsString,Form4.Ta
ble1OpcaoB.AsString,Form4.Table1OpcaoC.AsString,Form4.Table1OpcaoD.AsString,Form
4.Table1OpcaoE.AsString);
      if Minha_Resposta=Form4.Table1Resposta.AsString  then Inc(Nota);
    end;
  Form2.Timer1.Enabled:=False;
  Form1.Show;
end;

procedure Aprovado;
begin
  ShowMessage('Parabéns, você foi aprovado com nota '+IntToStr(Nota)+'.');
end;

procedure Reprovado;
begin
  ShowMessage('Você não conseguiu, estude mais! Sua nota foi
'+IntToStr(Nota)+'.');
end;

procedure Recuperacao;
begin
  ShowMessage('Atenção, você foi para a recuperação com nota
'+IntToStr(Nota)+'.');
  FazerProva;
  if Nota>=7*Quant_Perguntas/100 then Aprovado else Reprovado;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  Total_Perguntas:=Form4.Table1.RecordCount;
  SetLength(Sorteio,Total_Perguntas);
  Quant_Perguntas:=SpinEdit1.Value;
  Tempo_Total:=Quant_Perguntas*10;
  FazerProva;
  if Nota>=70*Quant_Perguntas/100 then Aprovado else Recuperacao;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  StatusBar1.Panels[0].Text:=FormatDateTIme('DD/MM/YYYY HH:MM:SS',Now);
end;

procedure TForm1.Sair1Click(Sender: TObject);
begin
  Application.Terminate;
```

```
end;

procedure TForm1.Trocar1Click(Sender: TObject);
begin
  if OpenPictureDialog1.Execute then
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
end;

procedure TForm1.Normal1Click(Sender: TObject);
begin
  Image1.Stretch:=False;
end;

procedure TForm1.Esticar1Click(Sender: TObject);
begin
  Image1.Stretch:=True;
end;

procedure TForm1.Calculadora1Click(Sender: TObject);
begin
  WinExec('C:\Windows\Calc.exe',SW_NORMAL);
end;

procedure TForm1.Paint1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\MSPaint.exe',SW_NORMAL);
end;

procedure TForm1.WordPad1Click(Sender: TObject);
begin
  WinExec('C:\Arquivos de Programas\Acessórios\WordPad.exe',SW_NORMAL);
end;

procedure TForm1.Sobre1Click(Sender: TObject);
begin
  Form3.ShowModal;
end;

procedure TForm1.Editor1Click(Sender: TObject);
begin
  Form4.Show;
end;

procedure TForm1.ComboBox1Change(Sender: TObject);
begin
  if ComboBox1.Text<>'' then
    begin
      Form4.Table1.Filter:='Materia = '+QuotedStr(ComboBox1.Text); //Filtra as
pergintas de acordo com a matéria selecionada no ComboBox. QuotedStr coloca
aspas na matéria, que é obrigado no Delphi
      Form4.Table1.Filtered:=True;                                  //Liga o
filtro
    end
  else Form4.Table1.Filtered:=False;                                //Desliga o
filtro se ComboBox em branco
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  ComboBox1.Items.Add(ComboBox1.Text);          //Adiciona o texto do ComboBox
nos itens do ComboBox
  ComboBox1.Items.SaveToFile('Matéria.txt');    //Salva os itens do ComboBox no
arquivo Matéria.txt
end;
```

```
procedure TForm1.FormCreate(Sender: TObject);
begin                                          //Ao criar o Form1
  if FileExists('Matéria.txt') then            //Se o arquivo Matéria.txt
existir
    ComboBox1.Items.LoadFromFile('Matéria.txt'); //Carregue este para o ComboBox
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
  DeleteFile('Matéria.txt');                   //Apaga o arquivo Matéria.txt
end;

end.
```

Anexo

E a lógica, como vai?

Existe a ilusão de que apenas conhecer uma boa linguagem credencia a bom desenvolvedor de sistema. Todavia, a Lógica de Programação tem papel fundamental na qualidade do programa.

Qual é a melhor linguagem?
"Qual é a linguagem do momento?" "Qual é o software mais usado no mercado hoje?" "Qual software você acha que devo aprender para estar atualizado em relação ao mercado?" "Qual é a Linguagem que mais tem futuro?"
É comum ouvirmos perguntas deste tipo de pessoas que desejam estar na vanguarda da Informática. Existe uma grande preocupação de atualizar-se em termos do melhor software de programação. Todavia, alguns, entusiasmados pelo software, esquecem daquilo que os move: a Lógica Computacional.

Balance Line?
Você já fez um "Balance Line"? Sabe o que é? Este termo (em desuso) significa fazer a consolidação de um ou mais arquivos seqüenciais, pré-ordenados, gerando um novo arquivo. Este era um problema típico a ser resolvido por um programador de Main Frame, quando o processamento Batch - em lotes - era largamente utilizado.
Programas como este exigiam do programador muito mais de sua lógica do que de seu conhecimento da linguagem de programação, ou seja, o maior esforço ficava por conta do raciocínio lógico para resolver o problema do que a codificação da solução em uma linguagem de programação.

O Retrabalho

Quando efetuamos um trabalho e este não está bem feito, naturalmente será preciso refazê-lo. Um programa mal construído, que não atinge o propósito a que foi concebido, está credenciado a ser refeito, mesmo que contenha requintes da linguagem. Programar "bonito", explorando bem os recursos que a linguagem proporciona não é garantia de que o programa desempenhará o papel esperado.

O que é um programa?
Uma das definições de programa é: "uma seqüência de instruções logicamente ordenadas a fim de solucionar um problema" .
Isto significa que, se desejamos resolver um problema, é preciso estabelecer quais passos precisamos efetuar e, depois disto, ordená-los de forma lógica a fim de que desempenhem o que deles se espera.

Elaborar Regras
Na verdade, o raciocínio Lógico utilizado para resolver um problema deve elaborar uma regra (ou várias regras) a ser seguida a fim de que o problema seja resolvido. A(s) regra(s) deve indicar por qual caminho a solução será alcançada.

E o Software?
Uma vez com a regra pronta, sabendo como chegar à solução, a próxima etapa é "traduzir" a regra em código, ou seja, programar a solução utilizando uma linguagem de programação.

E quem programa "direto"?
Você pode estar se perguntando: "mas e quem começa direto pela programação?" Ora, isto é comum nos programadores com maior experiência, mas, mentalmente, a solução foi arquitetada. Não há impeditivo a isto, mas é importante ter-se a solução antes da programação. Qualidade do Programa
Para que um programa tenha qualidade, são necessários alguns itens:

a) Requerimento:
definição clara (e preferencialmente documentada) do que se espera que o programa realize;
b) Estratégia de Solução:
A elaboração da Regra para solução do problema a resolver, ou seja, a Lógica do Programa;
c) Desenvolvimento Estruturado:
desenvolver o algoritmo que solucionará o problema / efetuará o trabalho;
d) Teste da Solução:
Validar a solução construída (elaborar e efetuar casos de teste, verificando se o programa produz os resultados esperados).

Estratégia de Solução (Regra)

A solução lógica do problema, seja ela elaborada via um diagrama de blocos ou mesmo arquitetada mentalmente, constitui fator primordial para a boa qualidade do programa.

Exemplo de Lógica
Para ilustrarmos, imagine a seguinte necessidade: temos a variável de memória mes, numérica, com valores variando de 1 a 12, e precisamos encontrar as 3 primeiras letras do mês correspondente ("JAN", "FEV", "MAR", ...), armazenando-as na variável string str.
Obviamente, há várias formas de resolver-se este problema, ou seja, não há uma forma lógica única de resolver-se. Porém, dependendo da solução lógica adotada, a programação será bem diferente.
Naturalmente, privilegiando o reuso, deve ser construída uma função (ou rotina) que possa ser utilizada em várias oportunidades. Comentaremos 3 alternativas para construção desta função:

Primeira Forma – Ninho de IF’s

```
IF mes = 1 then

str := "JAN"

else

if mes = 2 then
str := "FEV"
else

if mes = 3 then
str := "MAR" else ...
```

Obviamente este solução é longa e não é a mais adequada.

Segunda Forma - Vetor
Cria-se um vetor com 12 ocorrências, contendo as 3 primeiras letras dos 12 meses possíveis:

```
aMES [ 1] := "JAN"
aMES [ 2] := "FEV"
aMES [ 3] := "MAR"

.
.
.
aMES [12] := "DEZ"
str := aMES[mes]
```

Esta é uma solução bem mais curta e de fácil entendimento para quem lê o programa.

Terceira Forma - Substring de String

Cria-se um String contendo os 12 meses possíveis, enfileirando-se consecutivamente as 3 letras iniciais de cada mês e localiza-se o substring desejado por uma fórmula.

```
cMES = "JANFEVMARABRMAIJUN...DEZ";
str := copy(cMES,(mes-1)*3+1,3);
```

Nos dois primeiros exemplos, fica fácil verificar que funciona, embora seja preciso escrever bem mais. No terceiro exemplo, a escrita de programação é otimizada, sendo bem menor, porém o conteúdo de lógica é mais rebuscado.

Como saber se funciona? É aqui que entra o teste da solução...

Teste de Mesa
Este teste não precisa ser feito rodando-se o programa várias vezes. Basta a criação de uma pequena tabela (feita em Planilha Eletrônica, por exemplo):
Veja que o resultado da fórmula coincide com a primeira posição das 3 letras de cada mês!
Isto nos certifica de que o programa realizará exatamente o que dele se espera.

Conclusão
Conhecer bem uma boa linguagem de programação é importante mas não é tudo. Para termos programas com qualidade, que além de efetuar o trabalho desejado, facilitem a programação e a manutenção dos mesmos, é fundamental termos uma boa solução lógica do problema.

Meses Possíveis
Fórmula(mes- 1)*3+1

1	1
2	4
3	7
4	10
5	13
6	16
7	19
8	22
9	25
10	28
11	31
12	34

Paulo Sergio Borba, 33 anos, é Analista de Sistemas há 13 anos e Professor da Faculdade Radial São Paulo, onde leciona Linguagem e Técnicas de Programação no curso de Processamento de Dados.