Delphi 7

<< Iniciante >>

Diagramação e Arte Final: Albert Eije Barreto Mouta

Apresentação

Meu nome é Albert Eije. Já trabalho com informática há mais de 10 anos. Primeiramente trabalhei com computação gráfica. Com o tempo “migrei” para a programação. De início utilizei o Clipper. Velhos tempos. Depois que descobri o Delphi não larguei mais. Já desenvolvi vários sistemas comerciais em Delphi e também já usei alguns bancos de dados profissionalmente, tais como Paradox, Access, Interbase, Firebird e MySQL.

O objetivo desse curso é fazer o alicerce para quem quer aprender a programar. Muitas pessoas acham que basta abrir o Delphi e saber usar seus componentes. Mas isso não basta! Um bom programador precisa ter uma boa base de Lógica de Programação e vamos dedicar dois módulos a este assunto. Outra coisa importante é entender a estrutura do Object Pascal, a linguagem utilizada pelo Delphi. Conheço vários programadores, até bons, que não sabem declarar uma constante. Não sabem trabalhar com vetores e matrizes e quando pergunto o porquê, dizem que nunca precisaram usar esses recursos em seus sistemas. Imagino quantas linhas a mais de programação gastaram por não os terem usado.

Uma outra coisa importante para um programador que deseja fazer sistemas comerciais é a questão dos bancos de dados. Muitos programadores não tem noção de como elaborar um banco de dados, sendo que este é um dos primeiros passos ao se iniciar o desenvolvimento de um sistema. Por isso, vamos aprender quais os tipos de bancos de dados e como selecionar o melhor para o nosso caso. Depois vamos aprender a criar nossas tabelas de forma que quase não exista redundância dos dados.

Finalmente criaremos um programa no Delphi utilizando o banco de dados Paradox. A escolha por esse banco se deu pelo fato da facilidade de trabalhar com o mesmo e porque o mesmo também vem com o Delphi. Vamos também aprender a fazer qualquer tipo de relatório usando o QuickReport. A versão do Delphi utilizada no curso é a 7.

Parte do material aqui visto foi adquirido na própria internet. Meu trabalho foi compilá-lo em ordem para um melhor aprendizado e acrescentar pontos úteis.

A idéia inicial é lançar um módulo a cada semana, mas se alguns alunos já dominarem algum módulo ou concluirem rapidamente poderei disponibilizar logo outro para download.

Junto com os módulos irão arquivos e programas necessários para o aprendizado do mesmo.

Bem vindos e mãos a obra!

ÍNDICE

Módulo 1

Lógica de Programação I

INTRODUÇÃO

A automatização de tarefas é um aspecto marcante da sociedade moderna. O aperfeiçoamento tecnológico alcançado, com respeito a isto, teve como elementos fundamentais a análise e a obtenção de descrições da execução de tarefas em termos de ações simples o suficiente, tal que pudessem ser automatizadas por uma máquina especialmente desenvolvida para este fim, O COMPUTADOR.

Em ciência da computação houve um processo de desenvolvimento simultâneo e interativo de máquinas (hardware) e dos elementos que gerenciam a execução automática (software) de uma dada tarefa. E essa descrição da execução de uma tarefa, como considerada acima, é chamada algoritmo.

O objetivo dessa lição é a Lógica de Programação dando uma base teórica e prática, suficientemente boa, para que o aluno domine os algoritmos e esteja habilitado a aprender uma linguagem de programação. Será mostrado também um grupo de algoritmos clássicos para tarefas cotidianas, tais como: ordenação e pesquisa.

DEFINIÇÃO DE ALGORITMO

O conceito central da programação é o conceito de algoritmos, isto é, programar é basicamente construir algoritmos.

Algoritmo é a descrição, de forma lógica, dos passos a serem executados no cumprimento de determinada tarefa. O algoritmo pode ser usado como uma ferramenta genérica para representar a solução de tarefas independente do desejo de automatizá-las, mas em geral está associado ao processamento eletrônico de dados, onde representa o rascunho para programas (Software). Serve como modelo para programas, pois sua linguagem é intermediária à linguagem humana e às linguagens de programação, sendo então, uma boa ferramenta na validação da lógica de tarefas a serem automatizadas.

Um algoritmo é uma receita para um processo computacional e consiste de uma série de operações primitivas, interconectadas devidamente, sobre um conjunto de objetos. Os objetos manipulados por essas receitas são as variáveis.

É a forma pela qual descrevemos soluções de problemas do nosso mundo, a fim de serem implementadas utilizando os recursos do mundo computacional. Como este possui severas limitações em relação ao nosso mundo, exige que sejam impostas algumas regras básicas na forma de solucionar os problemas, para que possamos utilizar os recursos de hardware e software disponíveis. Pois, os algoritmos, apesar de servirem para representar a solução de qualquer problema, no caso do Processamento de Dados, eles devem seguir as regras básicas de programação para que sejam compatíveis com as linguagens de programação.

LINGUAGEM DE DESCRIÇÃO DE ALGORITMO (LDA)

Para escrevermos algoritmos é preciso uma linguagem clara e que não deixe margem a ambigüidades, para isto devemos definir uma sintaxe e uma semântica, de forma a permitir uma única interpretação das instruções num algoritmo.

Estrutura de um algoritmo

```
Algoritmo Nome_Do_Algoritmo
variáveis
Declaração das variáveis
Procedimentos
Declaração dos procedimentos
Funções
Declaração das funções
Início
Corpo do Algoritmo
Fim
```

Exemplo de um algoritmo cujo objetivo é usar um telefone público.

```
Início
Tirar o fone do gancho;
Ouvir o sinal de linha;
Introduzir o cartão;
Teclar o número desejado;
Se der o sinal de chamar
   Conversar;
   Desligar;
   Retirar o cartão;
Senão
   Repetir;
Fim.
```

Exercícios de Fixação

Um homem precisa atravessar um rio com um barco que possui capacidade de carregar apenas ele mesmo e mais uma de suas três cargas, que são: um leão, uma ovelha e um maço de capim. O que o homem deve fazer para atravessar o rio sem perder suas cargas?
Observação – Se a Ovelha ficar só, devora o capim.
Se o Leão ficar só, devora a ovelha.

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.

Variáveis

Unidades básicas de armazenamento das informações em nível de linguagens de programação. Os tipos de dados e variáveis utilizados dependem da finalidade dos algoritmos, mas, podemos definir alguns, pelo fato de serem largamente utilizados e implementados na maioria das linguagens, sendo estes:

* INTEIRO (INTEGER): qualquer número inteiro, negativo, nulo ou positivo.
* REAL (DOUBLE): qualquer número real, negativo, nulo ou positivo.
* CARACTER (STRING): qualquer conjunto de caracteres alfanuméricos.
* LÓGICO (BOOLEAN): tipo especial de variável que armazena apenas os valores V e F, onde V representa VERDADEIRO e F FALSO.

Declaração de variáveis

Para que os programas manipulem valores, estes devem ser armazenados em variáveis e para isso, devemos declará-las de acordo com a sintaxe:

```
NomeVariável : tipo;
Ex: ValorTotal : Double;
```

Constantes

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

Declaração de constantes

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

Exemplos:
```
Pi = 3.1416;
Empresa = 'Nivek Informatica'
V = Verdadeiro
```

Operações Básicas

Na solução da grande maioria dos problemas é necessário que as variáveis tenham seus valores consultados ou alterados e, para isto, devemos definir um conjunto de OPERADORES, sendo eles:

OPERADOR DE ATRIBUIÇÃO

NomeDaVariavel := Valor ou Expressão Atribuída. (":=" é o operador de atribuição utilizado pelo Delphi).

OPERADORES ARITMÉTICOS

+ = Adição	Quociente = Quociente da divisão de inteiros
* = Multiplicação	Resto = Resto da divisão de inteiros
- = Subtração ou inversor do sinal	EXP(a,b) = Exponenciação ab
/ = Divisão	

FUNÇÕES PRIMITIVAS: SEN(x); COS(x); TG(x); ABS(x); INT(x); Raiz(x); PI( );

OPERADORES RELACIONAIS

São utilizados para relacionar variáveis ou expressões, resultando num valor lógico (Verdadeiro ou Falso), sendo eles:

= - igual	<> - diferente
< - menor	> - maior
<= - menor ou igual	>= - maior ou igual

OPERADORES LÓGICOS

São utilizados para avaliar expressões lógicas, sendo eles:
* e (and) – e lógico ou conjunção
* ou (or) – ou lógico ou disjunção
* não (not) – negação.

PRIORIDADE DE OPERADORES

Durante a execução de uma expressão que envolve vários operadores, é necessária a existência de prioridades, caso contrário poderemos obter valores que não representam o resultado esperado.

A maioria das linguagens de programação utiliza as seguintes prioridades de operadores:

1º - Efetuar operações embutidas em parênteses "mais internos"
2º - Efetuar funções
3º - Efetuar multiplicação e/ou divisão
4º - Efetuar adição e/ou subtração
5º - Operadores relacionais
6º - Operadores lógicos

OBS: O programador tem plena liberdade para incluir novas variáveis, operadores ou funções para adaptar o algoritmo às suas necessidades, lembrando sempre de que estes devem ser compatíveis com a linguagem de programação a ser utilizada.

COMANDOS DE ENTRADA E SAÍDA

No algoritmo é preciso representar a troca de informações que ocorrerá entre o mundo da máquina e o nosso mundo, para isso devemos utilizar comandos de entrada e saída, sendo que, em nível de algoritmo, esses comandos representam apenas a entrada e a saída da informação, independe do dispositivo utilizado (teclado, discos, impressora,

monitor,...), mas, sabemos que nas linguagens de programação essa independência não existe, ou seja, nas linguagens de programação temos comandos específicos para cada tipo de unidade de Entrada/Saída.

Comando de Entrada de Dados
```
Leia(variável_1, variável_2,...)
```
Comando de Saída de Dados
```
Imprima(expressão_1, expressão_2,...)
```

COMANDOS DE CONTROLE DE FLUXO

Para representar a solução de um problema devemos escrever o conjunto de passos a serem seguidos, sendo que a maioria dos problemas exige uma dinâmica na sua solução, impondo assim que os algoritmos executem conjuntos de instruções de acordo com as possíveis situações encontradas no problema original. E de acordo com a Programação Estruturada os mecanismos utilizados para esse controle são: *Seqüência, Seleção e Repetição.*

SEQÜÊNCIA: usada para executar comandos passo a passo, sabendo que todos eles serão executados na ordem de escrita, sem nenhum desvio. Uma seqüência pode possuir um ou vários comandos, os quais devem ser delimitados pelos identificadores Início e Fim.

```
Inicio
Comando_1
...
Comando_n
Fim
```

CORPO GERAL DE UM PROGRAMA

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

ALGORITMO UM

Segue um Algoritmo que lê o nome e as 4 notas bimestrais de um aluno. Em seguida o Algoritmo calcula e escreve a média obtida.

```
PROGRAMA MEDIA_FINAL;
VAR
NOTA1, NOTA2, NOTA3, NOTA4, MEDIA: REAL;
NOME : CARACTERE [35]
INICIO
   LEIA (NOME);
   LEIA (NOTA1, NOTA2, NOTA3, NOTA4);
   MEDIA := (NOTA1 + NOTA2 + NOTA3 + NOTA4) / 4;
   ESCREVA (NOME, MEDIA);
FIM.
```

ALGORITMO DOIS

Segue um Algoritmo que lê o raio de uma circunferência e calcula sua área.

```
PROGRAMA AREA_CIRCUNFERENCIA;
CONST PI = 3.1416;
VAR RAIO, AREA : REAL;
INICIO
   LER (RAIO); {PROCESSAMENTO}
   AREA := PI * SQR(RAIO); {ENTRADA}
   ESCREVA (‘AREA = ’, AREA); {SAÍDA}
FIM.
```

{LINHAS DE COMENTÁRIO}

Podemos inserir comentários em um Algoritmo 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}
```

SELEÇÃO: usada para tomar decisões, ou seja, desviar a execução do algoritmo de acordo com uma condição, podendo ser simples ou composta.

Simples	Composta
Se (Expressão Lógica) Então Seqüência_1	Se (Expressão Lógica) Então Seqüência_1 Senão Seqüência_2

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:	Composta 1:
SE <<CONDIÇÃO>> ENTÃO <<COMANDO1>>	SE <<CONDIÇÃO>> ENTÃO <<COMANDO1>> SENÃO <<COMANDO1>>

Composta 2:

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

ALGORITMO TRÊS

Segue um Algoritmo que lê 2 números e escreve o maior.

```
PROGRAMA ACHA_MAIOR;
VAR A, B : INTEIRO;
INICIO
   LEIA (A, B);
   SE A>B ENTÃO
      ESCREVA (A)
   SENÃO
      ESCREVA (B)
FIM.
```

ALGORITMO QUATRO

Segue um Algoritmo que lê o nome e as 4 notas bimestrais de um aluno. Em seguida o Algoritmo calcula e escreve a média obtida pelo aluno escrevendo também se o aluno foi aprovado ou reprovado.

Média para aprovação = 6.

```
PROGRAMA MEDIA_FINAL;
VAR
NOTA1, NOTA2, NOTA3, NOTA4, MEDIA: REAL;
NOME : CARACTERE [35]
INICIO
   LEIA (NOME);
   LEIA (NOTA1, NOTA2, NOTA3, NOTA4);
   MEDIA := (NOTA1 + NOTA2 + NOTA3 + NOTA4) / 4;
   SE MEDIA>=6 ENTÃO
      ESCREVA (‘APROVADO’)
   SENÃO
      ESCREVA (‘REPROVADO’)
   ESCREVA (NOME, MEDIA)
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>>
```

ALGORITMO CINCO

Segue um Algoritmo que lê 3 números e escreve o maior.

```
PROGRAMA ACHA_MAIOR;
VAR A, B, C : INTEIRO;
INICIO
   LEIA (A, B, C);
   SE (A>B) E (A>C) ENTÃO
      ESCREVA (A)
   SENÃO SE (B>A) E (B>C) ENTÃO
      ESCREVA (B)
   SENÃO
      ESCREVA (C)
FIM.
```

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
```

ALGORITMO SEIS

Segue um Algoritmo que lê 3 números e escreve o maior.

```
PROGRAMA ACHA_MAIOR;
VAR A, B, C : INTEIRO;
INICIO
   LEIA (A, B, C);
   FACA CASO
      CASO (A>B) E (A>C)
         ESCREVA (A);
      CASO (B>A) E (B>C)
         ESCREVA (B);
      OUTROS CASOS
         ESCREVA (C);
   FIM DE CASO
FIM.
```

EXERCÍCIOS (Resolva todos os exercícios e concorra a um brinde no final do curso)

01) Crie um Algoritmo não computacional, que troque um pneu de carro.
02) Implemente um algoritmo capaz de encontrar o maior dentre 3 números inteiros quaisquer. Suponha todos serem distintos.
03) Implemente um algoritmo que leia 3 números quaisquer e os imprima em ordem crescente.
04) Escreva um subprograma capaz de calcular a média aritmética de três parâmetros passados.
05) Crie um algoritmo que pegue o nome de três pessoas, pegue a data de nascimento delas e identifique o sexo das mesmas. Depois mostre os nomes dessas pessoas seguido pelo sexo e idade e diga no final do algoritmo qual dos três é o mais velho.
06) NÃO 2 ** 3 < 4 ** 2 OU ABS(INT(-15 / 2)) < 10
07) 3 * (C / 4 + 5) < – 8 * 3 + (15 MOD 8 – 3) OU 5 ** 2 > INT(C * 0.7) onde C = 20
08) A ** 3 / B + 5 – C * D > C * D + A - B OU A // 2 / D < 18 – A onde A = 9 , B = 3 , C = 4 e D = 2
09) 5 + A * B ³ 16 // 4 – D E 6 / A * C / (A – B) = 234 OU A / 4 – (7 + 5 * C) < A ** 2 – 3 * B
10) Faça um algoritmo para calcular o peso ideal de uma pessoa sabendo sua altura.
DADO: para homens (72,7 * altura) – 58;
para mulheres (62,1 * altura) – 44,7.
11) Faça um algoritmo para ler nove número inteiros quaisquer. Tirar a média aritmética dos três primeiros, depois a média dos outros três e por fim a média dos três últimos. Escreva as três médias e a média das três médias.
12) Faça um algoritmo para ler os catetos de um triângulo retângulo e calcular e imprimir a sua hipotenusa.
13) Faça um algoritmo para ler duas variáveis inteiras e trocar o conteúdo lido de uma pela outra.
14) Faça um algoritmo para ler dois números e imprimir o maior, o menor ou então dizer se são iguais.
15) Faça um algoritmo para ler três números e imprimir se estes podem ou não formar um triângulo.
Observação – Para formar os lados de um triângulo cada um dos valores tem que ser menor que a soma dos outros dois.
16) Faça um algoritmo que leia as três notas, as faltas e o nome de um aluno e imprima sua situação. (“APROVADO”, “REPROVADO POR FALTA” ou “REPROVADO POR MÉDIA”)
Observação – A média para aprovação é 5.0 e o limite de faltas é 17.
17) Uma pessoa precisa comprar 3 remédios. Porém tem somente R$ 100,00. Faça um algoritmo que leia o nome e o preço de cada medicamento e escreva quais os medicamentos que ela pode comprar, se é que pode. (Para facilitar o algoritmo faça a compra por ordem de leitura.)
18) Faça um algoritmo que leia 9 números e escreva quantos números pares foram lidos.
19) Leia o nome e a idade de três pessoas e escreva seus nomes em ordem crescente de idade.
20) Leia três palavras e escreva se das palavras lidas, as três são diferentes, as três são iguais ou pelo menos duas são iguais.

ADQUIRA JÁ A SÉRIE DE CDS COMPLETE

260 Programas Com Fontes Prontos para Usar e Comercializar!

Entre eles:
SIGE PLUS 7.0
LAN-MAXX
SIS CLÍNICA 3.5
SIS COMÉRCIO
SIS HOTEL

E mais:
Mais de 150 apostilas
6 Aulas em Vídeo
Mais de 740 Componentes
Mais de 30 Programas Utiliários
Mais de 11.000 Glyphs
Mais de 18.000 Ícones
Mais de 400 Cursores Animados

3.158 Documentos

Apostilas, Livros, E-Books, Tutoriais, Dicas, How-Tos, Cursos e muito mais...

2.052 Documentos de Informática
Apenas de programação são 969 documentos.

E mais:

513 Documentos Profissionais
423 E-Books (Excelente para Vestibular)
21 Documentos para Prosperidade
6 Cursos de Idiomas (Espanhol, Francês, Latim, Esperanto, Japonês, Italiano)

E outros 143 Documentos!

114 Aulas em Vídeo

* Curso Completo de FireWorks em 20 Aulas
* Curso Completo de PHP em 25 Aulas
* Curso de Delphi em 6 Aulas
* Curso Visual Basic em 8 Aulas
* Curso Completo Flash MX
* Curso PhotoShop 7
* Curso Completo Visual Studio.Net em 45 Aulas

Acesse o site www.alberteije.com e veja mais detalhes sobre cada CD e como Adquirí-los.