INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Câmpus Gravataí

INSTITUTO FEDERAL SUL-RIOGRANDENSE
CAMPUS GRAVATAÍ
CURSO TÉCNCO EM INFORMÁTICA

APOSTILA

LÓGICA DE PROGRAMAÇÃO

Professor Me. Hunder Everto Correa Junior

Gravataí

2016

SUMÁRIO

1 Conceitos Básicos[i]

1.1 O que é lógica ?

Lógica é arte de pensar corretamente e, visto que a forma mais complexa do pensamento é o raciocínio, a lógica estuda ou tem em vista a "correção do raciocínio". A lógica ensina a colocar ordem no pensamento.

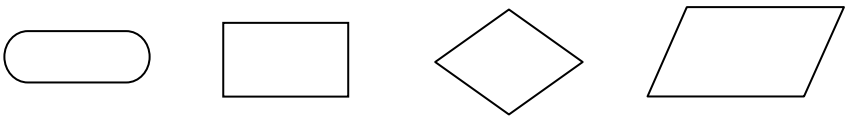
Exemplo: a) Todo mamífero é um animal
Todo cavalo é mamífero
Portanto, cavalo é animal

Exemplo: Para fazer um bolo devemos seguir uma receita de forma ordenada.
a) obter ingredientes
b) misturar ingredientes
c) colocar em uma forma
d) ligar o forno
e) colocar a forma dentro do forno
f) verificar se pronto
g) retirar do forno
h) servir o bolo

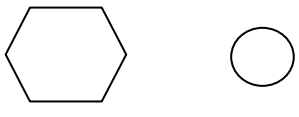
1.2 Fluxograma Tradicional e Diagrama de Chapin

Formas alternativas para representação de descrição de estrutura de soluções para problemas.

Fluxograma Tradicional - Principais Objetos

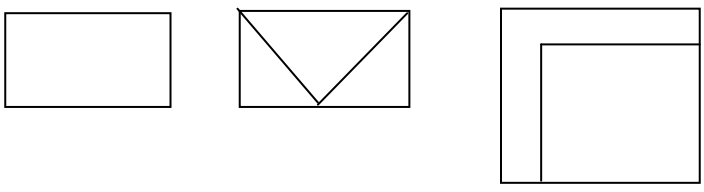


Terminador Processo Decisão Dados



Condição Concector

Diagrama de CHAPIN - Principais Objetos



Processo Decisão Condição

Exercício: Demonstrar a execução de uma receita de bolo através das ferramentas acima.

1.3 Algoritmos

Conceito

"É um conjunto finito de ações que obedecidas atingem um objetivo determinado, em um tempo finito. É um caminho para a solução de um problema, sendo que para resolução deste existem vários caminhos. Exemplos: Troca de uma lâmpada, troca de pneu, receita de bolo."

"Um algoritmo é a descrição de um padrão de comportamento, expressado em termos de um repertório bem definido e finito de ações primitivas, das quais damos por certo que possam ser executadas."

"Um algoritmo é um texto onde cada linha contém uma ação primitiva (ação elementar, passível de execução por ser humano ou máquina). A função do algoritmo quando executado, é a de agir sobre os dados, transformando-os em informações."

Todo algoritmo é composto por um grupo de ações primitivas. As ações primitivas que utilizaremos na construção de algoritmos serão construções lingüísticas da nossa linguagem natural (português),

Ítens a serem observados

Confiabilidade vs Complexidade

A confiabilidade de um produto é uma medida da sua capacidade de suportar o manuseio dentro de suas especificações de uso, mantendo-se capaz de atender às suas especificações de manuseio.
A complexidade de um produto pode ser determinada pela complexidade das interações que ocorrem com o mesmo. Uma vez que sistemas computacionais de médio porte oferecem uma grande variedade de regras de negócio que devem ser implementadas através de algoritmos e, normalmente, representam visões diferentes de vários setores. A complexidade do produto torna-se um desafio para a confiabilidade do sistema.

Manutenibilidade

Um algoritmo possui este atributo a medida em que facilita a incorporação de alterações. As alterações são responsáveis por 40% do custo total de um sistema, ao menos. Isto acontece porque a solução é pensada como um todo, diferentemente da construção de algoritmos onde cada bloco é construído separadamente para posterior integração, fazendo com que possa ser alterado isoladamente.

Flexibilidade

Os algoritmos devem ser flexíveis de forma que alterações no mundo externo possam ser facilmente assimiladas.

Importância da Documentação

É fundamental que documentemos o máximo possível a respeito de uma solução computacional adotada. No futuro esta documentação servirá como base para resolução de problemas, histórico de decisões tomadas, facilidade de compreensão por parte de outras pessoas e histórico de alterações.

Legibilidade

É a compreensão do algoritmo por qualquer pessoa, e a clareza com que a lógica é exposta.

2 Método para construção de algoritmos

1. Ler atentamente o enunciado
2. Retirar do enunciado a relação das entradas de dados
3. Retirar do enunciado a relação das saídas de dados
4. Determinar o que deve ser feito para transformar as entradas especificadas nas saídas determinadas
5. Construir o algoritmo
6. Executar o algoritmo (teste de mesa)

2.1 Tipos de dados

Inteiros: São os dados numéricos positivos ou negativos, excluindo-se destes qualquer número fracionário. De -32.768 até 32.768

Reais: São os dados numéricos positivos, negativos e fracionários. De 2,9 E-39 até 1,7 E+38.

Caracteres: São os tipos caracteres, as sequências contendo letras, nros e símbolos especiais, sendo representadas entre aspas " ". Também conhecido como alfanumérico, string ou literal.

Lógicos: Tipos lógicos ou boleanos, os dados com valores verdadeiro e falso. Somente representam um valor por vez.

2.2 Operadores Aritméticos

Chamamos de operadores aritméticos o conjunto de símbolos que representa as operações básicas da matemática.

Operador	Operação
//	Radiciação
**	Potenciação
*	Multiplicação
/	Divisão
+	Adição
-	Subtração
MOD	Resto da divisão
DIV	Quociente inteiro

2.3 Funções Matemáticas

Função	Exemplo
ABS( ) - Valor absoluto	ABS(-23) = 23
FRAC( ) - Parte fracionária	FRAC(56.8) = 8
INT( ) Parte inteira	INT(78,6) = 78
ROUND( )Arredondamento	ROUND(2.5) = 3

2.4 Prioridades entre operadores aritméticos

1. Parênteses mais internos
2. Funções matemáticas (ABS, FRAC, INT, ROUND)
3. **, //
4. *, /, DIV, MOD
5. +, -

2.5 Expressões Lógicas

Em algoritmos existem situações nas quais a execução de um comando dependem de uma determinada condição. Esta condição é representada por uma expressão lógica.

2.6 Operadores Relacionais

Operadores Relacionais	Símbolo
Igual	=
Diferente	<>
Maior que	>
Menor que	<
Maior ou igual	>=
Menor ou igual	<=

- Resultado destas expressões sempre resulta em verdadeiro ou falso

2.7 Operadores Lógicos

AND: Uma ação somente será executada se todas as condições mencionadas forem verdadeiras.

Cond1	Op	Cond2	Resultado
V	AND	V	V
V	AND	F	F
F	AND	V	F
F	AND	F	F

OR: Uma ação será executada se pelo menos uma das ações for verdadeira

Cond1	Op	Cond2	Resultado
V	OR	V	V
V	OR	F	V
F	OR	V	V
F	OR	F	F

XOR: Uma ação será executada se uma e somente uma das condições for verdadeira.

Cond1	Op	Cond2	Resultado
V	XOR	V	F
V	XOR	F	V
F	XOR	V	V

F	XOR	F	F

NOT: Uma ação não será executada se determinada condição não for verdadeira.

Op	Cond	Resultado
NOT	V	F
NOT	F	V

2.8 Prioridades entre operadores lógicos

1. NOT
2. AND , OR
3. XOR

2.9 Prioridade entre todos os operadores

1. Expressões internos dentro de parênteses
2. (-) sinal de menos utilizado como troca de sinal
3. operador NOT
4. Operadores multiplicativos (**, //, *, /, DIV, MOD, AND)
5. Operadores aditivos (+, -,OR, XOR)
6. Operadores Relacionais (<, <=, =, >=, >, <>)

2.10 Memória e Conceito de Variáveis

Nos computadores, dispositivos chamados de memória permitem que as instruções a serem executadas e os dados a serem manipulados sejam armazenados temporariamente. Esta memória pode ser vista como um armário composto por muitas gavetas que são chamadas de variáveis. A designação variáveis é devido ao fato de podermos substituir seu conteúdo quando necessário. Todas as variáveis devem receber um nome para a sua identificação, estes nomes são denominados de identificadores.

Tipos de Dados Primitivos

Toda variável pertence a um tipo de dado que define o conjunto de valores que ela pode receber (armazenar), assim como o conjunto de operadores que podem trabalhar sobre a mesma.

Tipos primitivos são os tipos de dados predefinidos pela linguagem de programação que se está utilizando. Iremos assumir que nossa linguagem trabalha com os seguintes tipos de dados: Inteiro, Real, Caracter e lógico.

Identificadores

Os nomes escolhidos para rotular as variáveis devem obedecer as seguintes regras:

1- primeiro caracter deve ser uma letra
2- nomes devem ser formados por caracteres pertencentes ao seguinte conjunto:{A,B,....,Z, 0,1,....9, _)
3- Os nomes escolhidos devem explicitar seu conteúdo

2.11 Construção de um Algoritmo

2.11.1 Declaração de Variáveis

Declarar uma variável significa reservar uma "gaveta" na memória, dar-lhe um rótulo. O tipo de dados com o qual a variável é declarada serve para especificar qual é o conjunto de valores que ela pode armazenar.

Exemplo:

DECLARAÇÃO DE VARIÁVEIS
INTEIRO: total;

2.11.2 Declaração de Constantes

Uma constante é determinado valor fixo que não se modifica ao longo do tempo, durante a execução do algoritmo. Por exemplo, se tivermos quatro avaliações durante o curso, a média aritmética das notas será dada pela expressão: (n1+n2+n3+n4)/4
As notas são variáveis, dependem do desempenho do aluno, mas o número de avaliações é constante.

Exemplo:

DECLARAÇÃO DE CONSTANTES
Num_avaliacoes = 4;

2.11.3 Comandos de atribuição

Comando a descrição de uma ação a ser executada em um dado momento. Para atribuirmos um valor a uma determinada variável usaremos o seguinte comando:

TOTAL <- 78;

2.11.4 Comandos de Entrada e Saída

Comandos de entrada e saída têm por finalidade receber dados e devolver informações para o meio externo. Para isso utilizaremos os seguintes comandos:

LEIA(Dado1, Dado2, ...); {leitura dos dados informados}
ESCREVA("Título", Informação1, Informação2);{envio de informações para o usuário}

{} - É a notação utilizada para inserir comentários em um algoritmo.

2.11.5 Conceito de Bloco

Um bloco pode ser definido como uma sequência de ações primitivas, que como um todo, possui uma função bem definida, neste caso um algoritmo pode ser visto como um bloco.

Algoritmo <nome> { Nome do algoritmo em desenvolvimento}

INICIO {Início do algoritmo}

Declaração de variáveis e constantes

Sequência de ações primitivas

Fim {Fim do bloco}

2.11.6 Sequência Simples

Sequência simples é um conjunto de ações primitivas que serão executadas numa sequência linear de cima para baixo, ou seja, na ordem em que foram escritas. As ações primitivas sempre devem ser encerradas por um ponto e vírgula (;).

Ex:
ALGORITMO

DECLARAÇÃO DE CONSTANTES

DECLARAÇÃO DE VARIÁVEIS

INÍCIO

Ação primitiva 1;

Ação primitiva 2;

FIM;

FIM.

2.11.7 Estrutura Alternativa Simples

Podemos fazer com que um bloco seja ou não executado, dependendo do resultado obtido na inspeção de uma condição, pode ser falso ou verdadeiro.

```
SE ( Condiçao_A )
   ENTÃO
       Início {Início do bloco verdade}
           A_P_1;
           A_P_2;
       Fim {fim do bloco verdade};
```

2.11.8 Estrutura Alternativa Composta

```
SE Condição_A
    ENTÃO
        A_P_1;
SENÃO
    INÍCIO {Início do bloco falso}
        A_P_2;
        A_P_3;
    FIM;
```

2.11.9 Seleção de múltipla escolha

SINTAXE:

```
ESCOLHA <variável> DE
        Valor1 : comando1;
        Valor2: Comando2;
        Valor3,valor4,valor5:INICIO
                                Comando3;
                                Comando4;
                            FIM;
        SENÃO
                Comando6;
FIM ESCOLHA;
```

3 Máximas da Programação

a) Algoritmos devem ser feitos para serem lidos por seres humanos. Tenha em mente que seus algoritmos deverão ser lidos e entendidos por outras pessoas de tal forma que possam ser corrigidos, receber manutenção e ser modificados.
b) Escreva os comentários no momento que estiver escrevendo o algoritmo.
c) Os comentários deverão acrescentar alguma coisa, o conjunto de comandos nos diz o que o algoritmo faz os comentários deverão nos contar o porquê.
d) Use comentários no prólogo. Alguns destes comentários seriam:
   - Descrição do que faz o algoritmo
   - Como utilizá-lo
   - Explicações do significado das variáveis mais importantes
   - Estruturas de dados utilizadas
   - Autor
   - Data de escrita
e) Utilize espaços em branco para facilitar a visibilidade
f) Escolher nomes representativos para suas variáveis
g) Utilizar apenas um comando por linha
h) Utilizar parênteses para melhorar a legibilidade e prevenir-se de erros
i) Utilizar a "identação" para mostrar a estrutura lógica do algoritmo

4 Estruturas de Repetição

Muitas vezes temos necessidade de repetir uma sequência de ações primitivas (bloco). Por exemplo, em uma escola, o cálculo da média final é exatamente o mesmo para todos os alunos, neste caso, basta executar o bloco que calcula a média várias vezes (uma para cada aluno). O número de repetições pode ser indeterminado, porém finito, caso contrário o algoritmo não teria sentido, pois sua execução consumiria um tempo infinito de tempo.

No caso dos alunos, poderíamos estabelecer uma condição de parada da seguinte forma:

Se o número de alunos não for conhecido, utilizaremos um valor predefinido com sendo um finalizador. Por exemplo, podemos definir que o número de matrícula igual a 0 representa o fim dos alunos e não deve ser considerado, neste caso, o número de repetições poderá ser controlado por uma das sentenças abaixo:

- Calcular a média enquanto o número de matrícula for diferente de zero.
- Calcular a média até que o número da matrícula seja igual a zero.

Se o número de alunos for conhecido, podemos controlar o número de repetições, contanto o número de médias calculadas.
Podemos criar uma variável, digamos "contador", acumular mais um ao seu conteúdo após a execução de cada cálculo:

Contador <- contador + 1;

Neste caso o número de repetições poderá ser controlado por uma das sentenças abaixo:

- Calcular a média, enquanto o contador for menor ou igual ao número de alunos.
- Calcular a média, até que o contador seja maior que o número de alunos.
- Calcular a média, fazendo o contador variar de 1 até o número de alunos.

4.1 Estrutura ENQUANTO

Esta estrutura permite que um bloco ou uma ação primitiva seja repetida, enquanto uma determinada condição for satisfeita.

```
ENQUANTO (cond_laco) FAÇA

    INICIO

        A_p_1;
        A_p_2;

    FIM;
```

Exemplo

Construir algoritmo para cálculo da média de alunos segundo definição anterior.

4.2 Estrutura REPITA...ATÉ

Esta estrutura tem o seu funcionamento também controlado por decisão, porém irá efetuar a execução de um conjunto de instruções pelo menos uma vez antes de verificar a validade da condição estabelecida.

Desta forma, REPITA tem seu controle de funcionamento em sentido contrário a ENQUANTO, pois sempre irá processar um conjunto de comandos, no mínimo uma vez, até que condição se torne verdadeira.

O Comando REPITA dispensa o uso do bloco INICIO / FIM.

```
REPITA
      {Conjunto de comandos};
ATÉ <Condição>;
```

4.3 Estrutura PARA...DE...ATÉ...PASSO...FAÇA...FIMPARA

Esta estrutura têm seu funcionamento controlado por uma variável denominada contador. Sendo assim, poderá executar um conjunto de comandos (Bloco) um determinado número de vezes.

Caso haja mais de um comando a ser executado é necessário incluir INÍCIO / FIM.

```
PARA <variável> de <Início> até <Fim> PASSO <Incremento> FAÇA
       {Comandos};
FIMPARA;
```

4.4 Comparação entre estruturas de repetição

Podemos estabelecer dois postulados que relacionam as estruturas de repetição:

a) Toda estrutura ENQUANTO pode ser convertida para REPITA e vice-versa
b) Toda estrutura PARA pode ser convertida em ENQUANTO, mas nem toda estrutura ENQUANTO pode ser convertida em PARA.

Características de cada estrutura de repetição

Estrutura	Condição	Qtd. de Execuções	Condição de existência
ENQUANTO	Início	?	Condição verdadeira
REPITA	Final	Minímo 1	Condição falsa
PARA	Não tem	Número definido	(Ni < Nf) passo positivo (Nf < Ni) passo negativo

Exemplo :

Escreva os números pares entre 1 e 30, efetuando a soma entre eles, utilizando as 3 estruturas de repetição.

5 Estruturas de Dados

Técnica de programação que permitirá trabalhar com o agrupamento de várias informações dentro de uma mesma variável. Vale salientar que este agrupamento ocorrerá obedecendo sempre ao mesmo tipo de dado, e por esta razão é chamado de estruturas de dados homogêneas ou tipo matriz.
A utilização deste tipo de estrutura de dados recebe diversos nomes, como: variáveis indexadas, variáveis compostas, vetores, matrizes, tabelas em memória ou arrays. Assim ,são vários os nomes utilizados. Nesta disciplina utilizaremos a denominação matriz.
As matrizes são tipos de dados que podem ser "construídos" à medida que se fazem necessários, pois não é sempre que os tipos básicos (real, inteiro, caracter e lógico) e/ou variáveis simples são suficientes para representar a estrutura de dados utilizada em um programa.

5.1 Matrizes de uma dimensão ou vetores:

Este tipo de estrutura é também chamado de matrizes unidimensionais. Caracteriza-se por ser definida uma única variável dimensionada com um determinado tamanho. A dimensão de uma matriz é constituída por constantes inteiras e positivas.
Os nomes dados às matrizes seguem as mesmas regras de identificadores para variáveis simples.

Associação: Imaginemos um edifício com um determinado número de andares, com um único apartamento por andar, representando uma estrutura de dados. Cada andar representa partições desta estrutura. Visto que os andares são uma segmentação direta do prédio, estes compõe então o que é chamado estrutura unidimensional .

SINTAXE:

DECLARAÇÃO DE TIPOS

    Var_Tipo_Vetor : VETOR[dimensão] de TIPO;

DECLARAÇÃO DE VARIÁVEIS

    Var_Tipo_Vetor: Variável;

Onde:

Var_Tipo_Vetor: Identificador para o novo tipo de estrutura de dados criado;
Vetor: Tipo de estrutura de dados definido;
Dimensão: Valor inicial e final do tamanho do vetor
TIPO: Qualquer um dos tipos básicos já definidos
Variável: Variável para a qual foi atribuído o tipo vetor.

Manipulação: Ao imaginar o elevador de um prédio sabemos que este é capaz de acessar qualquer um de seus andares. Para acessarmos o conteúdo de um vetor precisamos saber o identificador da variável criada e a posição que queremos acessar.

Exemplo 1:

Algoritmo Predio;

Declaração de Tipos

Vet_Predio : Vetor[1..10] de Inteiros;

Declaração de Variáveis

Vet_Predio: viAndares;

Inicio

viAndares[1] <- 5; {Atribuição}
LEIA(viAndares[2]);
ESCREVA(viAndares[2]);

Fim;

Exemplo:

Calcular e obter a média geral de uma turma de oito alunos. A média a ser obtida deve ser a média geral das médias de cada aluno obtida durante o ano letivo.

6 Matrizes multidimensionais

São estruturas de dados homogêneas que podem ser construídas a partir do tipo vetor. São estruturas bastante utilizadas em matemática, mas do ponto de vista da estrutura de dados, estamos mais interessados na forma de acesso aos dados armazenados.

A mais comum é a matriz de duas dimensões, por se relacionar diretamente com a utilização de tabelas.

Um importante aspecto a ser considerado é que na manipulação de uma matriz do tipo vetor, é utilizada uma única instrução de looping. No caso de matrizes com mais dimensões, deverá ser utilizado o número de loopings relativos ao seu número de dimensões. Desta forma, uma matriz de duas dimensões deverá ser controlada por dois loopings e assim sucessivamente.

Ex. Matriz de 3 dimensões (3 linhas e 3 colunas)

1	100	12
2	243	12
3	32	17

Sintaxe

DECLARAÇÃO DE TIPOS

Var_Tipo_Matriz : MATRIZ[dimensão1, dimensão2] de <TIPO>;

DECLARAÇÃO DE VARIÁVEIS

Var_Tipo_Matriz: Variável;

Onde:

Var_Tipo_Matriz: Identificador para o novo tipo de estrutura de dados criado;
Matriz: Tipo de estrutura de dados definido;
Dimensão: Valor final do tamanho de cada dimensão da matriz;
TIPO: Qualquer um dos tipos básicos já definidos
Variável: Variável para a qual foi atribuído o tipo matriz.

Exemplo:

```
Algoritmo Predio;

Declaração de Tipos

        Mat_Predio : Matriz[10,2] de Inteiros;

Declaração de Variáveis

        Mat_Predio: viAndares;

Inicio

        viAndares[1,1] <- 5; {Atribuição}
        LEIA(viAndares[1,2]);
       ESCREVA(viAndares[1,2]);

Fim;
```

Exemplos:

1. Escrever um algoritmo que leia uma matriz de duas dimensões com 3 linhas e 3 colunas do tipo inteiro. A primeira coluna se refere ao código de uma produto, a segunda coluna se refere ao valor do produto e a terceira coluna se refere a quantidade existente em estoque. Ao final o algoritmo deve escrever os valores informados.
2. Escrever um algoritmo que leia uma matriz de três dimensões (2x2x2) e ao final escreva a soma de cada linha.
3. Ler duas matrizes A e B, cada uma de duas dimensões com 5 linhas e 3 colunas. Construir uma matriz C de mesma dimensão; onde C é formada pela soma dos elementos da matriz A com os elementos da matriz B. No final escreva os valores das 3 matrizes.

7 Tipo Escalar (Tipos Definidos Pelo Usuário)

Este tipo pode ser definido como uma sequência de elementos que constituem um conjunto.

Como o objetivo do português estruturado é permitir a descrição de algoritmos da forma mais simples e próxima do problema, existe uma grande flexibilidade para a criação de novos tipos.

Exemplo:

Declaração de Tipos

Dia = (Dom, Seg, Ter, Qua, Qui, Sex, Sab);

Inteirosde1a5 = 1:5;

Declaração de Variáveis

Dia   : Dias_da_semana;
Inteirosde1a5 : SubConjunto;

Nos exemplos acima a variável Dias_da_semana somente aceitará atribuições de valores entre Dom e Sab, já a variável SubConjunto somente aceitará atribuições de valores entre 1 e 5.

8 Estruturas de Dados Heterogêneas

Segundo Wilson Silva Pinto em seu livro Introdução ao Desenvolvimento de Algoritmos e Estrutura de Dados, o tipo Registro é: "Uma estrutura composta por um conjunto de variáveis de tipos diferentes, primitivos e/ou estruturados, logicamente relacionadas, que podem ser referenciadas por um mesmo nome (identificador do tipo de registro ou individualmente). Este tipo de estrutura é utilizado para relacionar dados pertencentes a um mesmo objeto.
Por exemplo, se estivermos interessados em montar um fichário, contendo os dados dos funcionários, será necessário montar um modelo de ficha padrão e a partir daí cadastrar todos os funcionários. Note que todas as fichas terão o mesmo tamanho, portanto deverão conter o mesmo número de informações. Estas informações estão contidas em subdivisões chamadas de campo, logo, um registro é composto por campos, que por sua vez é composto por caracteres."

Sintaxe:

Declaração de Tipos

```
Reg = REGISTRO
        Campo1 :    Tipo_campo1;
        Campo2:     Tipo_campo2;
        Campo3:     Tipo_campo3;
                .
                .
                .
        Campon:     Tipo_Campon;
```

Declaração de Variáveis

```
        REGISTRO: Variável_de_Registro;
```

Exemplo:

```
Tipos
Alunos = REGISTRO
        Matricula:  Caracter;
        Nome:       Caracter;
        Idade:      Inteiro;
        Nota:       Real;

Variáveis
Alunos: Turma;

Inicio
Turma.Matricula < - "123456789";
Turma.Nome < - "José";
Turma.Idade < - 24;
Turma.Nota < - 8.5;

LEIA(Turma.Matricula);
```

Escreva(Turma.Idade);

Fim.

Exemplos:

1) Escreva um algoritmo que leia a seguinte ficha cadastral:

Funcionário:

Matrícula:		Nome:	
Função:		Salário:	

Para cada funcionário lido faça o seguinte:

a) se número da matrícula superior a 15840 então o valor de seu salário deve ser aumentado em 5%.
b) Se número da matrícula inferior a 15840 então o valor de seu salário deve ser aumentado em 10%
c) Após o cálculo escrever o resultado
d) A matrícula 99999 encerra a leitura

2) Escreva um algoritmo que leia a seguinte ficha cadastral:

Produtos

Código:		Descrição:	
Quantidade:		Tipo:	
Valor unitário:			

Para cada produto lido:

a) O código do produto é numérico superior a zero
b) A descrição não pode ser em branco ""
c) A quantidade não pode ser negativa
d) O tipo do produto deve ser entre 1 e 3
e) O valor deve ser maior que zero
f) Após a leitura, escrever o nome do produto e sua quantidade multiplicada pelo seu valor unitário
g) A leitura do produto de código 0 encerra a leitura

9 Lista de Exercícios

1) Faça um algoritmo que leia 3 números inteiros e escreva o produto desses números
2) Ler uma temperatura em graus Centígrados e apresentá-la em graus Fahrenheidt. A fórmula de conversão é: F <- (9 * C + 160) /5
3) Faça um algoritmo que leia 2 números e diga se eles são múltiplos ou não. Um número inteiro dividido por outro dando resto zero é múltiplo.
4) Faça um algoritmo que leia a primeira letra do nome de um estado da região sul e escreva o nome deste estado por extenso.
5) O custo ao consumidor de um carro novo é a soma do custo de fábrica com a porcentagem do distribuidor e dos impostos (aplicados ao custo de fábrica). Supondo que a porcentagem do distribuidor seja de 28% e dos impostos de 45%, escrever um algoritmo para ler o custo de fábrica de um carro e escrever o valor final ao consumidor
6) Faça um algoritmo que tendo como dados de entrada o preço de um produto e um código de origem, emita o preço junto com a sua procedência. Caso o código não seja nenhum dos especificados, o produto deve ser encarado como importado.
   1 - sul
   2 - norte
   3 - leste
   4 - oeste
   5 ou 6 - nordeste
   7,8 ou 9 - sudeste
   10 até 20 - centro-oeste
   25 até 50 - nordeste
7) Faça um algoritmo que leia um número e determine se ele é múltiplo de 8 (NÃO usar MOD)
8) Construir um algoritmo que converta um valor lido em reais para um valor em moeda estrangeira escrevendo o resultado utilizando a seguinte codificação:
   1 - libra esterlina (R$ 3,80)
   2 - franco suiço (R$ 1,85)
   3 - dólar americano (R$ 2,43)
   4 - marco alemão (R$ 2,31)
9) dado um conjunto de valores inteiros e positivos, escrever ao final qual o menor valor do conjunto. A leitura de um valor zero ou negativo indicará o fim da leitura dos dados.
10) Altere o algoritmo anterior para que o mesmo escreva ao seu final o menor e o maior valor informado.
11) Construir um algoritmo para calcular a média de um conjunto de valores inteiros e positivos fornecidos. A Leitura de um número zero ou negativo indicará o final da leitura dos dados.
12) Calcular o imposto de renda de um grupo de contribuintes considerando:

a) Os dados fornecidos serão: número do CPF, número de dependentes e renda anual.
b) Para cada contribuinte será feito um desconto de R$ 120,00
c) Os valores da alíquota para cálculo do imposto são:

Renda líquida	Alíquota
Até R$ 900,00	isento
de R$ 900,01 até R$ 2000,00	5%
de R$ 2000,01 até R$ 3000,00	10%
de R$ 3000,01 até R$ 5000,00	15%
acima de R$5000,00	20%

d) O último contribuinte, que não deverá ser considerado, terá seu CPF igual a 0.

13) Escrever um algoritmo para calcular o fatorial de um número (N), após sua leitura.
14) Escrever um algoritmo que:
    Leia um número real
    determine e imprima o seguinte somatório: s = x - (x / 1!) + (x / 2!) - (x / 3!) + ... usando os primeiros 20 números da série.
15) Escreva um algoritmo para um programa que leia um conjunto de 50 informações contendo, para cada um, a altura e o código do sexo de uma pessoa (código 1 se for masculino e código 2 se for feminino) , que calcule e escreva:
    - a maior e a menor altura da turma
    - a média das alturas das mulheres
    - a média da altura da turma
16) Escreva um algoritmo para um programa que leia um conjunto de informações sobre 100 alunos de um curso contendo matrícula e nota, determine as duas maiores notas, juntamente com o número da matrícula do aluno que obteve cada uma delas.
17) A série de Fibonacci é formada pela sequência:
    1,1,2,3,4,5,13,21,34,55.... etc.
    Escreva um algoritmo que calcule a série de Fibonacci até o vigésimo termo.
18) Faça um algoritmo que utilize as três estruturas de repetição, p/imprimir a tabuada do número 5.
19) Faça um algoritmo que leia um valor N. Este valor indica quantos mais devem ser lidos (inteiros e positivos). Após a leitura desses valores, escreva o somatório desses números lidos.
20) Faça um algoritmo que leia um número e escreva a tabuada de todos os números de 1 até o valor lido.
21) Faça um algoritmo que fique lendo valores até encontrar o valor zero, com cada valor lido faça a soma de 10 valores subsequentes e mostre a soma e a média desses valores.
22) Dados dois números inteiros A e B tal que A > 0 e B > A construir um algoritmo que encontre todos os números primos existentes entre A e B. (nros.primos divisíveis exatamente por 1 e por ele mesmo).
23) Dada uma sequência de letras, construa um algoritmo para localizar a letra que ocorreu mais vezes e determine o número de ocorrências. Considere que o caracter "*" seja o finalizador.
24) Faça um algoritmo que liste todos os números pares de 1 a 1000
25) Faça um algoritmo que liste todos os números ímpares de 1 a 1000
26) Faça um algoritmo para ler um vetor de 5 elementos e imprimi-los ao final da leitura. Os elementos são do tipo caracter.
27) Faça um leia um número qualquer. Após esta leitura o número lido deve ser procurado em um vetor de 20 elementos. Ao final o algoritmo deve mostrar
28) Faça um algoritmo que leia um vetor de 50 elementos, todos inteiros. O algoritmo deve gerar um outro vetor de 50 elementos do tipo lógico, que assume verdadeiro (V) na posição correspondente se o número do vetor original for Par e falso (F) se o número for ímpar.
29) Construir um algoritmo para calcular o número de alunos que tiraram nota acima da média da turma. As notas estão dispostas da seguinte forma: N1, N2,N3, ...NK. onde Nk é a nota do aluno número K. Considere uma turma de 40 alunos.
30) Construir um algoritmo capaz de verificar se um determinado caracter existe em um vetor de caracteres de 100 posições. Ao final o algoritmo deve imprimir a posição onde se encontra o caracter.

31) Dado o algoritmo abaixo qual o valor de L após a execução ?

```
ALGORITMO X;

DECLARAÇÃO DE VARIÁVEIS
        LÓGICO: A, B, C;
        REAL:X, Y
        INTEIRO: V,L;
INICIO
        A<- Falso;
        B<- Verdadeiro;
        C<-Falso;
        X-< 1.5;
        Y<- 3,2;
        SE C OU (( X + Y > 5) OU (NOT A E B)) ENTÃO
                L <- 0;
        SENÃO
                L<-1;
        FIM SE;
FIM.
```

32) O que será impresso na execução do algoritmo seguinte se :

a) NUM = 10 b) NUM = 0 c) NUM = -4 ?

```
ALGORITMO y;

DECLARAÇÃO DE VARIÁVEIS
        CARACTERE: QUAL;
        INTEIRO: NUM;
INICIO
        LEIA(NUM); {um dos valores acima}

        SE NUM > 0 ENTÃO
                QUAL <- "Número Positivo";
        SENÃO
                 SE NUM < 0 ENTÃO
                      QUAL <- "Número Negativo" ;

                 SENAO
                       QUAL <- "ZERO";
                FIMSE;
        FIM SE;
FIM.
```

33) O que está errado no algoritmo abaixo ?

```
ALGORITMO xy;

DECLARAÇÃO DE VARIÁVEIS
        INTEIRO: n, par, x;
INICIO
        LEIA(n);
        x <- n MOD 2

        SE X = 0 ENTÃO
                PAR < - "Verdadeiro";
        SENÃO
                PAR < - "Falso";
        FIM SE;
FIM.
```

34) Qual é a primeira operação a ser executada em cada um dos comandos abaixo ?

a) X + Y - Z;
b) A + B / C ** 2;
c) JOAO + JOSE / JOEL;
d) MARIA + JOAO + BETE + JUNIA;
e) X + Y + B ** 2 + R * 3;
f) A * B / C * D;

35) Os comandos abaixo são equivalentes ? Explique porquê.

```
a <- b = c;
e
SE B = C  ENTÃO
    A <- verdadeiro;
SENÃO
   A <- falso;
FIMSE;
```

36) Sabendo que anos bissextos são múltiplos de 4 (ex: 1984). Construa um algoritmo que verifique se um determinado ano é bissexto.
37) Construir um algoritmo que dado um triângulo faça o seguinte: classificar em equilátero, isósceles ou escaleno.
38) Efetuar o cálculo de uma prestação em atraso utilizando a seguinte fórmula: prestação = valor + (valor * (taxa / 100) * tempo)
39) Faça um algoritmo que leia três valores e determine o maior e o menor dos 3
40) Construa um algoritmo que verifique uma senha fornecida pelo usuário. A senha é um conjunto de 5 caracteres "ABCDE". De acordo com a senha fornecida escreva: "ACESSO NEGADO" ou "ACESSO PERMITIDO"

41) Dado o seguinte algoritmo:

```
Algoritmo Sequencia;
Variáveis
        Inteiro:    Atual, Final, Ultimo, Penultimo, Contador;
Inicio
        Atual := 1;
        Ultimo := 0;
        Penultimo := 1;
        Contador := 1;
        Final := 8;
        Para Contador de 1 até Final faça
        Inicio
          Atual := Ultimo + Penultimo;
          Penultimo := Ultimo;
          Ultimo := Atual;
          Imprima (Atual);
        Fim;
Fim.
```

- Escreva a seqüência de valores escritos:

42) Escreva um algoritmo que dados dois valores faça o seguinte:
- Efetue as operações de soma, subtração, multiplicação e divisão entre os dois valores escrevendo o resultado obtido em cada uma das operações;
- Não deve ser efetuada a operação de divisão se algum dos valores for igual a zero;
- Não deve ser efetuada a operação de multiplicação caso dos dois valores informados sejam iguais a zero.

43) Num frigorífico existem 90 bois. Cada boi traz preso no seu pescoço um cartão contendo um número de identificação e seu peso. Implementar um algoritmo que escreva o número e o peso do boi mais gordo e do boi mais magro (não é necessário armazenar os dados de todos os bois). O número de identificação igual a zero encerra a leitura.

44) Leia os seguintes dados dos funcionários de uma empresa: matrícula, nome e salário. Para cada matrícula lida calcular o valor de desconto do INSS utilizando a seguinte fórmula:

Salário	Desconto
< 500	isento
500-700	8%
700-900	9%
> 1000,00	10%

Para cada funcionário imprimir sua matrícula, nome, salário, desconto INSS e salário líquido.

Ao final apresentar o total pago a todos os funcionários e total de descontos calculados. O número de matrícula 999 encerra a leitura de dados.

45) Faça um algoritmo que escreva 4 notas e escreva a média final
46) Faça um algoritmo que calcule o salário líquido de um professor. Você possui os seguintes dados: Valor da hora aula, número de aulas dadas no mês e percentual de desconto do INSS. Em primeiro lugar estabeleça o salário bruto para efetuar o desconto e obter o salário líquido.
47) Faça um algoritmo que leia três números, se todos forem números pares efetue a soma dos mesmos, se todos forem ímpares efetue uma multiplicação, e em caso contrário efetue a subtração de um pelo outro. O algoritmo deve emitir mensagens do tipo:
"Todos pares, a soma é:"
"Todos ímpares, o resultado da multiplicação é:"
"Ambos tipos, o resultado da subtração é:"
48) Elaborar um algoritmo que efetue o cálculo do reajuste de salário de um funcionário. Considere que o funcionário deverá receber um reajuste de 15% caso seu salário seja menor que 500, se maior ou igual a 500 e inferior a 1000 o reajuste deverá ser de 12,5%, se maior ou superior a 1000 o reajuste deverá ser de 10%. Escreva o resultado final.
49) Tendo como dados de entrada a altura e o sexo de uma pessoa, construa um algoritmo que calcule seu peso ideal, utilizando as seguintes fórmulas:
a) sexo masculino: (72.7 * h) - 53 b) sexo feminino: (62.1 * h) - 48.7
50) Desenvolver um algoritmo que efetue a leitura dos elementos de uma matriz A (2 x 3). Construir uma matriz B (2 x 3), observando a seguinte lei de formação: Se o valor for par, ele deverá ser multiplicado por 5; sendo ímpar deverá ser somado 5. Ao final mostrar o conteúdo das duas matrizes.
51) Desenvolver um programa que efetue a leitura de uma matriz A (4 x 2). No final apresente o total da soma de todos os elementos que sejam ímpares.
52) Desenvolva um programa que leia uma matriz A (10 x 5), ou seja, 10 pessoas serão cadastradas com as informações de nome, endereço, cep, bairro e telefone. Após, faça a leitura de um número de telefone qualquer. Em seguida, efetue uma busca para identificar se o número de telefone foi, ou não, cadastrado. Se cadastrado imprima os dados da pessoa, se não solicite nova digitação. O número de telefone "99" encerra o programa.
53) Elabore um algoritmo que leia uma matriz 2 x 5 e escreva o total de números pares encontrados
54) Elabore um algoritmo que leia uma matriz 3 x 10 e escreva o total de números ímpares encontrados
55) Fazer um algoritmo que leia uma matriz de 5 x 5 elementos e após encontre o maior elemento, o menor elemento e o valor médio dos números contidos na matriz.

Ler um número não determinado de valores, até que o número lido seja 0, e para cada número lido verificar se ele não pertence a uma matriz 8 x 8. Em caso afirmativo, escreva o valor e a posição em que foi encontrado, caso contrário, escreva o valor e a mensagem "Não Encontrado".

56) Considere a ficha cadastral abaixo:

Funcionário:

Sexo:		Nome:	
Idade:		Salário:	

Para cada funcionário lido faça o seguinte:

a) O campo sexo deve aceitar somente os valores fem ou mas

b) O nome da pessoa não pode ser igual a branco ("")
c) A idade do funcionário deve ser maior que 18 e menor que 40
d) O salário deve ser maior que zero e calculado da seguinte forma:
se a idade do funcionário estiver entre 30 e 35 deve receber um abono de R$ 1.000,00, se for maior que 18 e menor que 25 deve receber um abono de R$ 10,00, nos outros casos recebe um abono de R$ 500,00.
e) Após o cálculo escrever o registro com o novo salário calculado.
f) A leitura de um salário igual a 0 encerra a leitura de cartões.

57) Considere a ficha cadastral abaixo:

Aluno:

Matrícula:		Nome:	
NotaG1:		NotaG2:	

Para cada aluno lido faça o seguinte:

a) A matrícula pode receber caracteres e números, porém não pode ser igual a branco ("")
b) O nome do aluno não pode ser igual a branco ""
c) A nota de G1 não pode ser inferior a 1.0 ou superior a 10.0
d) A nota de G2 não pode ser inferior a 7.0 ou superior a 10.0
e) A matrícula * encerra a leitura de alunos
f) Após a leitura de cada aluno imprimir a média do aluno e número de alunos lidos até o momento.

58) Dado o seguinte vetor de caracteres:

T	R	X	S		E	O	B	A	!
1	2	3	4	5	6	7	8	9	10

Qual será a sua configuração após serem executados os comandos a seguir:

```
AUX < - VET[6];
VET[6] < - VET[9];
VET[9] < - AUX;

Para I de 1 até 4 passo 1 faça
        AUX < - VET[I];
        VET[I] < - VET[9-I];
        VET[9-I] < - AUX;
FimPara;
VET[6] < - VET[2];
```

10 Bibliografia

Básica:

- BERG, A. C.; FIGUEIRÓ, J. P. Lógica de Programação . Canoas, Ed. ULBRA, 2ª edição, 2002.
- FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. Lógica de Programação - A construção de algoritmos e estruturas de dados. São Paulo: Makron Books, 2ª edição, 2000

Complementar:

- GUIMARÃES, Â. M.. Algoritmos e Estruturas de Dados. Livros Técnicos e Científicos Editora, 1985.
- SALIBA, W. L. C. Técnicas de Programação: uma abordagem estruturada. São Paulo: Makron Books, 1992.
- TERADA, R.; SETZER, V.. Introdução à computação e à construção de algoritmos. São Paulo: Makron Books, 1992.