



# Lógica de Programação

Lucas Mendes

0

## ▼ Algoritmos



**Algoritmos** são conjuntos de passos finitos e organizados que, quando executados, resolvem um determinado problema.

1

## ▼ Algoritmos Computacionais



São passos a serem seguidos por um módulo processador (tudo que pode efetuar processamento ou ser programável) e seus respectivos usuários (pessoas que vão utilizar o software) que, quando executados na ordem correta, conseguem realizar determinada tarefa (resolver algum problema ou necessidade).



São rotinas executadas por algum processador.



**Todo algoritmo computacional começa com:** lógica de programação → linguagem de programação → sistema completo.

## ▼ Comandos de saída



// **Escreva** ("Qualquer coisa") - Exibe na tela a expressão.



// **Escreval** ("Qualquer coisa") - Exibe na tela a expressão e pula uma linha.



// **Escreva** (msg) - Exibe na tela o conteúdo da variável se a mesma foi declarada.



// **Escreva** ("Qualquer coisa", msg) - Exibe na tela a expressão e o conteúdo da variável se a mesma foi declarada.

## ▼ Variáveis



São espaços na memória do computador destinado a um dado que é alterado durante a execução do algoritmo.



As variáveis precisam ser definidas por nomes e tipos.



var identificador: tipo - para declaração de variável. (nome do identificador e tipo primitivo).



4 tipos primitivos (Inteiro, real, caractere e lógico).



Atribuição usamos  $\leftarrow$  para atribuir conteúdo para a variável. msg  $\leftarrow$  "Ola, Mundo!" . (msg vai receber Olá, Mundo!).

## 2

### ▼ Comandos de entrada



// **Leia (var)** - Vai armazenar o valor para dentro do espaço da variável.



Solicita algum conteúdo para o usuário digitar.

### ▼ Operadores aritméticos

+ (Adição), - (Subtração), \* (Multiplicação), / (Divisão), \ (Divisão Inteira), ^ (Exponenciação), % (Módulo(Resto da divisão)).

### ▼ Ordem de precedência

() (Parênteses), ^ (Exponenciação), \*/ (Multiplicação e Divisão), +- (Adição e Subtração).

### ▼ Funções Aritméticas

Abs (Valor absoluto) - Ex: Abs(-10) =10

Exp (Exponenciação) - Ex: Exp(3,2) =9

Int (Valor Inteiro) - Ex: Int(3.9) =3

RaizQ (Raiz Quadrada) - Ex: RaizQ(25) =5

Pi (Retorna Pi) - Ex: Pi =3.14

Sen (Seno (rad)) - Ex: Sen(0.523) =0.5

Cos (Cosseno (rad)) - Ex: Cos(0.523) =0.86

Tan (Tangente (rad)) - Ex: Tan(0.523) =0.57

GraupRad (Graus para Rad) - Ex: GraupRad(30) =0.52)

### 3

#### ▼ Operadores relacionais



Criam a relação entre variáveis ou expressões, comparar variáveis, expressões, gerando resultados lógicos, como: V ou F. (Gerando sempre um valor lógico)

> (Maior que), < (Menor que), >= (Maior ou igual a), <= (Menor ou igual a), = (Igual a), <> (Diferente de).

#### ▼ Operadores lógicos



Servem apenas para comparar outros resultados lógicos (diferente dos relacionais). Retornam resultados lógicos: V ou F.



E = V e V = V, V e F = F, F e V = F, F e F = F. OU = V ou V = V, V ou F = V, F ou V = V, F ou F = F. NÃO = não V = F, não F = V.



## Macete (Operador E, OU).

p	q	p E q	p	q	p OU q

Operador E. (Se Paula e Quésia estão felizes, logo, eu estou também. As 2 precisam estar felizes para que eu esteja também). Operador OU. (Se Paulo ou Quésia estão felizes, logo, eu estou também. Um das 2 precisam estar feliz para que eu esteja feliz).

### ▼ Ordem de precedência geral



1º Aritméticos. () (Parênteses), ^ (Exponenciação), \*/ (Multiplicação e divisão), +- (Adição e subtração). 2º Relacionais. Todos. 3º Lógicos. E, OU, NÃO.

## 4

### ▼ Comandos da formatação de casas do número real



// Escreva("Qualquer coisa", Variável:4:1) - Serão exibidas 4 casas ao todo, e 1 após a vírgula.

## 5

### ▼ Estruturas condicionais (Simples e Composta)



Executa um bloco, caso a expressão seja verdadeira (Condicional simples).



Executa um bloco, caso a expressão seja verdadeira, se caso contrário executa o segundo/outro bloco (Condicional composta).

#### ▼ Comandos da estrutura condicional simples



```
// Se (expressão) entao  
    Bloco
```

FimSe

- Se a expressão for verdadeira, o bloco será executado.

#### ▼ Comandos da estrutura condicional composta



```
// Se (expressão) entao  
    Bloco A
```

```
senao  
    Bloco B
```

FimSe

- Se a expressão for verdadeira, o bloco A será executado, caso contrário o bloco B será executado.

#### ▼ Indentação

É uma forma de organizar o código, para mostrar quando um bloco de código pertence a um comando pai.



Sempre que abrir uma estrutura feche em seguida para não se perder (Se FimSe) (Escolha FimEscolha).

## 6

### ▼ Estruturas condicionais (Aninhada e Escolha Caso)



Se a primeira situação ocorre, executa o primeiro bloco, se a segunda situação ocorrer, executa o segundo bloco, caso contrário o terceiro bloco será executado (Condicional aninhada).



Quando existem muitos teste com valores numéricos simples (Inteiro), é útil utilizar a estrutura condicional escolha caso. É colocada a variável como escolha, e um caso para cada valor, de acordo com o valor de cada caso um bloco será executado, opcionalmente existe o OutroCaso, se nenhum valor acima for escolhido, o bloco do OutroCaso será executado (Condicional escolha caso).



Não serve para testar faixa de valores (<, >, <=, =, ...), funciona para valores inteiro (VisuAlg) (Condicional escolha caso).



É possível separar o número do valor por vírgulas. (Condicional escolha caso). Ex:

Escolha D

Caso 1, 2, 3

Bloco A

FimEscolha

#### ▼ Comandos da estrutura condicional aninhada



```
// Se (situação 1) entao
    Bloco A
senao
    Se (situação 2) entao
        Bloco B
    senao
        Bloco C
    FimSe
FimSe
```

- Se a situação 1 ocorrer, o bloco A será executado,  
Se a situação 2 ocorrer, o bloco B será executado,  
Caso contrário, o bloco C será executado.

#### ▼ Comandos da estrutura condicional escolha caso





// Escolha (variável)

Caso valor

Bloco A

Caso valor

Bloco B

Caso valor

Bloco C

OutroCaso (Opcional)

Bloco D

FimEscolha

- Se caso o valor 1 for escolhido o bloco A será executado, se nenhum valor acima for escolhido, o Bloco D será executado.

## 7

### ▼ Estrutura de repetição (Enquanto)

Enquanto uma expressão for verdadeira, o bloco será executado, quando chegar no FimEnquanto voltará para o Enquanto fazendo um looping, toda vez que "entrar" no Enquanto a expressão será testada.



A repetição (looping) será desfeita quando a expressão for falsa.



Caso ocorra a repetição (looping) infinito, basta pressionar CTRL + F2, ou parar o algoritmo (VisuAlg).

## ▼ Comandos da estrutura de repetição (Enquanto)



// Enquanto (expressão) faça

Bloco

FimEnquanto

- Enquanto uma expressão for verdadeira faça um bloco e FimEnquanto. (looping entre Enquanto e FimEnquanto)

## 8

## ▼ Estrutura de repetição (Repita)



O teste lógico é feito no final, repita o bloco até que a expressão lógica for verdadeira, caso não seja verdadeiro, ele repete até a expressão lógica ser verdadeira.



A repetição (looping) será desfeita quando a expressão for verdadeira.

## ▼ Comandos da estrutura de repetição (Repita)



// Repita

Bloco

Ate (expressão)

- Repita o bloco até a expressão lógica ser verdadeira.


## 9

### ▼ Estrutura de repetição (Para)

Variável de controle, O bloco será executado de acordo com a quantidade determinada de vezes (controle).

A estrutura Para é usada especificamente quando já se sabe quantas vezes as coisas vão acontecer, sabendo o início e fim da contagem.

### ▼ Comandos da estrutura de repetição (Para)

```
 // Para variavel ← inicio ate fim [passo salto] faca  
    Bloco
```

FimPara

- A variável recebe a quantidade de vezes que se deseja fazer a repetição (o salto é opcional), o bloco será executado de acordo com a quantidade de vezes determinada.

## 10

### ▼ Rotinas (Procedimento)

Todos os passos são colocados dentro do procedimento. Sempre que tiver um procedimento repetitivo (rotina), basta escrever apenas uma vez dentro de um procedimento e toda vez que for preciso utilizar os passos, basta fazer uma chamada do procedimento.

### ▼ Comandos de rotinas (Procedimento)



```
// Procedimento NomeProcedimento( )
```

```
    inicio
```

```
        Bloco
```

```
        Bloco
```

```
FimProcedimento
```

- Quando NomeProcedimento for chamado o bloco será executado, chegando no FimProcedimento retorna ao programa principal, até uma nova chamada do Procedimento.

#### ▼ Parâmetro (Valor)



Parâmetros são tratados como variáveis, mas por serem passados por um procedimento, são chamados de parâmetros.



Não usar o nome do parâmetro com o mesmo nome da variável (VisuAlg).



Apenas o valor é copiado para dentro do parâmetro.

#### ▼ Comandos parâmetros (Valor)



// Procedimento Soma(A, B: Inteiro)

início

EscrevaL("Recebi o valor",

A)

EscrevaL("Recebi o valor", B)

EscrevaL("A soma vale", A + B)

FimProcedimento

- A soma recebe 2 parâmetros, Escreva recebeu o valor A, o próximo recebe o valor B e por fim ele escreve a Soma A + B.

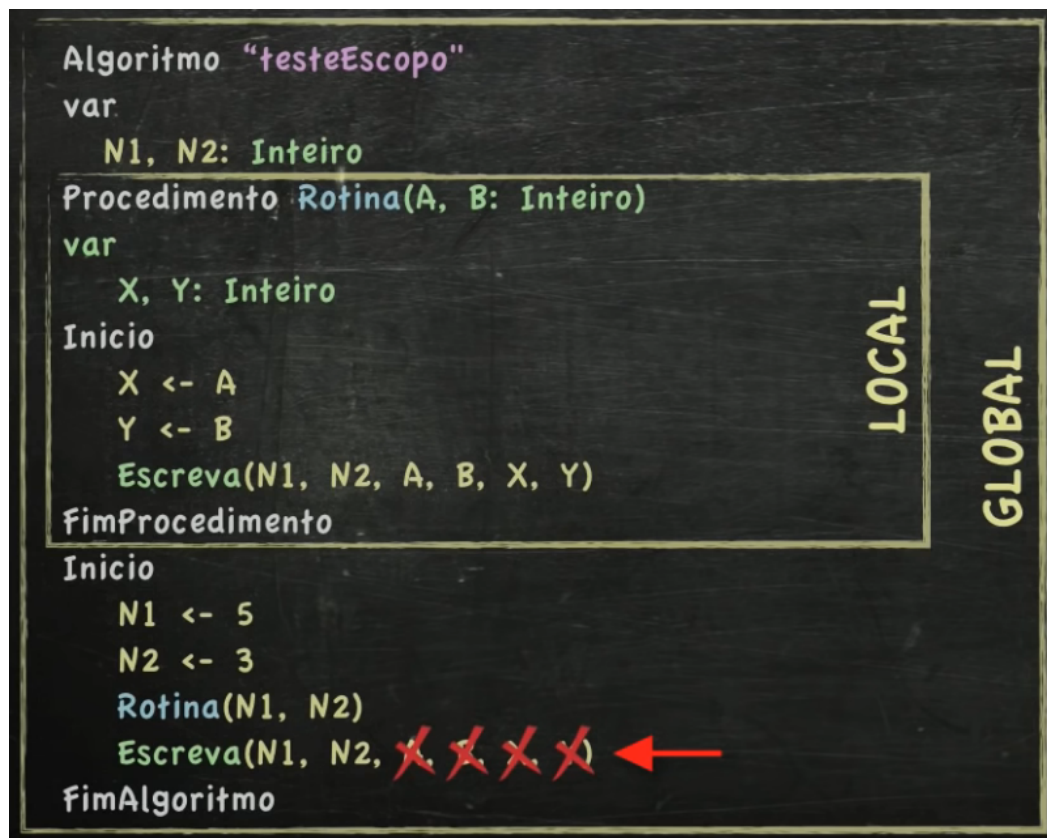
#### ▼ Escopo



Local onde determinada variável vai funcionar. Escopo global variáveis definidas fora do procedimento (Funcionarão em qualquer lugar do programa). Escopo local variáveis definidas dentro do procedimento (Funcionarão apenas dentro do procedimento).



Escopo (Global e Local).



Global N1, N2. As 2 variáveis vão funcionar em qualquer lugar do programa.  
Local A, B, X, Y. Os parâmetros e variáveis vão funcionar apenas no escopo local.

### ▼ Parâmetro (Referência)



O parâmetro tem uma referência automática ao valor da variável original, qualquer alteração no parâmetro vai afetar na variável original.

### ▼ Comandos parâmetros (Referência)



```
// Procedimento Soma(var A, B: Inteiro)
```

```
  inicio
```

```
    A ← A + 1
```

```
    B ← B + 2
```

```
    EscrevaL("A soma vale", A + B)
```

```
FimProcedimento
```

```
  inicio
```

```
    X ← 4
```

```
    Y ← 8
```

```
    Soma(X, Y)
```

```
    EscrevaL(X, Y)
```

```
FimAlgoritmo
```

- É passado a referência de X para A e de Y para B. Ao somar 1 no A dentro do procedimento, automaticamente é somado +1 no X valendo 5, o mesmo para B e Y valendo 10, mostrando os valores no final temos: 5, 10.

## 11

### ▼ Rotinas (Funções)



As funções podem retornar um resultado, podendo tratar no código principal o que será feito com esse resultado.



Funções permitem que você trate a formatação e a exibição do resultado diretamente no programa principal.



Os dois tipos de passagem de parâmetros é o mesmo tanto para os procedimentos quanto para as funções.

### ▼ Comandos de rotinas (Funções)



```
// Funcao Soma(A, B: Inteiro): Inteiro
```

```
var
```

```
    S: Inteiro
```

```
inicio
```

```
    S ← A + B
```

```
    Retorne S
```

```
FimFuncao
```

```
inicio
```

```
    N1 ← 5
```

```
    N2 ← 4
```

```
    RES ← Soma(N1, N2)
```

```
    EscrevaL("A soma é ", RES)
```

```
FimAlgoritmo
```

- Valor de N1 e N2 é passado para A e B (parâmetros), a soma é feita, na linha de retorne será retornado o resultado da soma (9) para a variável RES, no final será exibido o resultado.

### ▼ Funções do VisuAlg (Pré-definidas, valores caractere)





São funções já pré-definidas, funções de tratamento de string.

### ▼ Comandos de funções pré-definidas (Valores caractere)

Exemplo: Var Site ← "CursoEmVideo"



// Compr(Site) = comprimento da string = 12

Copia(Site, 6, 2) = var caractere, números onde começa, quantas casas = Em

Maiusc(Site) = letras maiúsculas = CURSOEMVIDEO

Minusc(Site) = letras minúsculas = cursoemvideo

Pos("Video", Site) = posição = 8

Asc("C") = código da letra = 67

Carac(67) = letra do código = C

- Funções pré-definidas.

## 12

### ▼ Variáveis compostas (Vetores)



As variáveis compostas são alocadas uma do lado da outra de acordo com índice (posição endereço dentro variável). (Variáveis compostas homogêneas unidimensionais - vetores)



Variáveis simples são alocadas de acordo com a necessidade do sistema operacional.



É possível guardar vários valores em uma mesma variável, como se fosse salvar dados, mas na memória do computador.

### ▼ Comandos de variáveis compostas (Vetores)



```
// var
    n: vetor[1..4] de inteiro
inicio

    n[1] ← 3
    n[2] ← 5
    n[3] ← 1
    n[4] ← 0
FimAlgoritmo
```

- É criada a variável composta n com 4 espaços alinhados, após a criação é atribuído 4 valores nos 4 espaços.

## 13

### ▼ Variáveis compostas (Matrizes)



A matriz possui mais de uma dimensão, sendo espaços para linhas e colunas (linhas verticais e colunas horizontais), são colocados verticalmente de acordo com o índice e horizontalmente de acordo com o outro índice. (Variáveis compostas homogêneas multidimensionais - matrizes).

12  
34

VisuAlg possui um ícone de geração de números aleatórios, muito útil para testar vários valores (inteiros ou reais).



1

Matriz identidade, matriz quadrada em que os elementos da diagonal principal são iguais a 1 e os demais elementos são iguais a 0.

#### ▼ Comandos de variáveis compostas (Vetores)



// var

m: vetor[1..3, 1..2] de inteiro

inicio

$m[1,2] \leftarrow 4$

$m[2,2] \leftarrow 5$

$m[3,1] \leftarrow 8$

FimAlgoritmo

- É criada a variável composta m com 3 linhas, 2 colunas, com a dimensão de 3 por 2. É atribuído 3 valores, 4 na linha 1, coluna 2. 5 na linha 2, coluna 2. 8 na linha 3, coluna 1.

1		4
2		5
3	8	
	1	2