

# Algoritmos

## Aula 03 – Algoritmos: elementos básicos

**Professora**

**Laysa Mabel de Oliveira Fontes**

mabel.fontes@ufersa.edu.br

Pau dos Ferros/RN

2025

# Palavras Reservadas

## *Palavras Reservadas*

*São palavras que tem significados pré-determinados e fazem parte da estrutura da linguagem utilizada.*

# Palavras Reservadas

## Palavras Reservadas

<b>aleatorio</b>	<b>character</b>	<b>e</b>	<b>fimalgoritmo</b>	<b>grauprad</b>	<b>maiusc</b>	<b>passo</b>	<b>randi</b>
<b>abs</b>	<b>caso</b>	<b>eco</b>	<b>fimenquanto</b>	<b>inicio</b>	<b>mensagem</b>	<b>pausa</b>	<b>repita</b>
<b>algoritmo</b>	<b>compr</b>	<b>enquanto</b>	<b>fimescolha</b>	<b>int</b>	<b>minusc</b>	<b>pi</b>	<b>se</b>
<b>arccos</b>	<b>copia</b>	<b>entao</b>	<b>fimfuncao</b>	<b>interrompa</b>	<b>nao</b>	<b>pos</b>	<b>sen</b>
<b>arcsen</b>	<b>cos</b>	<b>escolha</b>	<b>fimpara</b>	<b>leia</b>	<b>numerico</b>	<b>procedimento</b>	<b>senao</b>
<b>arctan</b>	<b>cotan</b>	<b>escreva</b>	<b>fimprocedimento</b>	<b>literal</b>	<b>numpcarac</b>	<b>quad</b>	<b>timer</b>
<b>arquivo</b>	<b>cronometro</b>	<b>exp</b>	<b>fimrepita</b>	<b>log</b>	<b>ou</b>	<b>radpgrau</b>	<b>tan</b>
<b>asc</b>	<b>debug</b>	<b>faca</b>	<b>fimse</b>	<b>logico</b>	<b>outrocaso</b>	<b>raizq</b>	<b>verdadeiro</b>
<b>ate</b>	<b>declare</b>	<b>falso</b>	<b>função</b>	<b>logn</b>	<b>para</b>	<b>rand</b>	<b>xou</b>

# Função de Saída

Se uma calculadora realiza várias operações, mas não tem um *display* para mostrar os resultados, qual a utilidade dela?



# Função de Saída

**Todo algoritmo deve exibir mensagens com os valores de saída.**

- Todas as linguagens de programação permitem isso;
- Em pseudocódigo, utiliza-se a função **escreva**<sup>1</sup>.


<sup>1</sup> A função escreval exibe uma mensagem na tela e, em seguida, pula uma linha.

# Função de Saída

- **Sintaxe:**

**escreva**(<mensagem composta por texto<sup>2</sup> e/ou variáveis/expressões<sup>3</sup>>)

- **Exemplo:**



```
algoritmo "Olá Mundo"  
inicio  
    escreva ("Olá mundo!")  
fimalgoritmo
```

<sup>2</sup> Todo texto deve estar entre aspas.

<sup>3</sup> As variáveis/expressões devem estar fora das aspas e separadas do texto por meio de vírgulas.

Ao criar algoritmos ou programas é necessário armazenar informações para poder utilizá-las durante sua execução.

**Para tornar isso possível, utiliza-se variáveis!**

# Variáveis

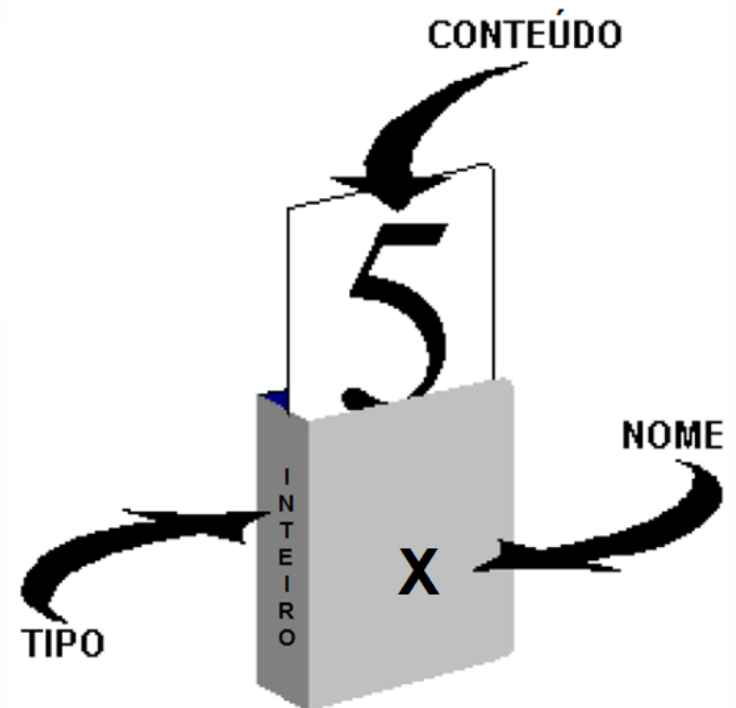
## *Variáveis*

*São espaços reservados para armazenar algum dado, por analogia, como uma caixa que serve para armazenar algo.*



# Variáveis

Quando se diz que uma **variável** **x** assume um **valor 5**, se quer dizer, na realidade, que existe uma **posição de memória**, representada simbolicamente por **x**, que contém o **valor 5**.



# Declaração de Variáveis

**Para que o computador possa executar comandos que envolvem variáveis da maneira correta, ele deve conhecer os detalhes das variáveis que pretendemos utilizar.**

- Esses detalhes são:
  - O identificador dessa variável;
  - O tipo de valores que essa variável irá conter.

# Declaração de Variáveis

- **Sintaxe:**

**var**

<variável 1>, <variável 2>, ..., <variável  $n$ >: <tipo das variáveis>

- **Exemplo:**

algoritmo "Idade"

var

i: inteiro

inicio

escreval("Quantos anos você tem? ")

leia(i)

escreva("Você tem", i, " anos de idade.")

fimalgoritmo

# Nomeação de Variáveis

- Deve seguir as seguintes regras:
  - Só pode conter letras, números e *underline*;
  - Deve começar com uma letra ou *underline*;
  - Letras não podem ter acentos;
  - Não pode começar com número;
  - Não pode ter espaços;
  - Não pode ser uma palavra reservada;
  - Deve ter no máximo 127 caracteres.
  - Não podem ser repetidas dentro do algoritmo.

# Nomeação de Variáveis

- **Exemplos válidos:**

- media
- \_media
- nota2
- media\_final

- **Exemplos inválidos:**

- média
- 2nota
- media final
- nome-completo

# Nomeação de Variáveis

**Na sintaxe do pseudocódigo, não há diferença entre letras maiúsculas e minúsculas.**

- **Exemplo:**
  - NOME é o mesmo que noMe

# Função de Entrada

**Nem todos os dados que um algoritmo manipula são gerados por ele. Deve haver um meio para que sejam atribuídos os dados (entradas) para o algoritmo.**


- Todas as linguagens de programação permitem isso;
- Em pseudocódigo, utiliza-se a função **leia**.

# Função de Entrada

- **Sintaxe:**

**leia**(<variável 1>, <variável 2>, ..., <variável  $n$ >)

- **Exemplo:**



```
algoritmo "Olá"  
var  
    n: caractere  
inicio  
    escreva("Digite seu nome: ")  
    leia(n)  
    escreva("Olá ", n)  
finalgoritmo
```



# Tipos de Dados

- Os dados podem ser divididos em quatro tipos:
  - Inteiro
  - Real
  - Caractere
  - Lógico

# Inteiro

**Toda e qualquer informação numérica que pertença ao conjunto dos números inteiros.**

- **Exemplos:**

- 5
- -15
- 0
- 1340

# Real

**Toda e qualquer informação numérica que pertença ao conjunto dos números reais.**

- **Exemplos:**

- 3
- -52.453
- 0
- 3.74

# Caractere

**Representa textos, ou seja, cadeia de caracteres entre aspas. Esses textos podem ser constituídos por números, letras e símbolos.**

- **Exemplos:**
  - “Rua Getúlio Vargas, nº 8”
  - “Luiz Felipe da Silva”
  - “F”

**Representa valores lógicos, ou seja, verdadeiro ou falso.**

- **Exemplos:**
  - verdadeiro
  - falso

# Atribuição

## *Atribuição*

*Uma atribuição, representada pelo operador “:=”, define a ação de atribuir um determinado valor a uma variável.*

# Atribuição

```
algoritmo "Declarações_e_Atribuições"
```

```
var
```

```
    idade: inteiro
```

```
    altura: real
```

```
    tiposangue, endereco: caractere
```

```
    doador: logico
```

```
inicio
```

```
    idade := 26
```

```
    altura := 1.70
```

```
    altura := 1.67
```

```
    tiposangue := "A"
```

```
    endereco := "Av. Norte, 34, Recife"
```

```
    doador := verdadeiro
```

```
fimalgoritmo
```

idade	altura	tipoSangue	endereço	doador
(inteiro)	(real)	(caractere)	(caractere)	(lógico)

idade	altura	tipoSangue	endereço	doador
(inteiro)	(real)	(caractere)	(caractere)	(lógico)
26	1.67	A	Av. Norte, 34, Recife	verdadeiro

# Atribuição

Nas atribuições, **NÃO** é permitido inserir valores em uma variável de um tipo diferente do seu.

**Exceção:** o único caso permitido é a atribuição de um valor inteiro a uma variável real. O contrário não é permitido!



# Atribuição

```
algoritmo "Atribuições_Simples"  
var  
    peso: real  
    nome: caractere  
    achei: logico  
inicio  
    peso := 78.7  
    nome := "João da Silva"  
    achei := falso  
fimalgoritmo
```

Correto!

# Atribuição

```
algoritmo "Atribuições_Simples"  
var  
    x, y: inteiro  
inicio  
    y := 2  
    x := y + 2  
    x := x + 2  
fimalgoritmo
```

Correto!

# Atribuição

```
algoritmo "Atribuições_Simples"  
var  
    salario: real  
inicio  
    salario := "Insuficiente"  
fimalgoritmo
```

**Incorreto!**

# Atribuição

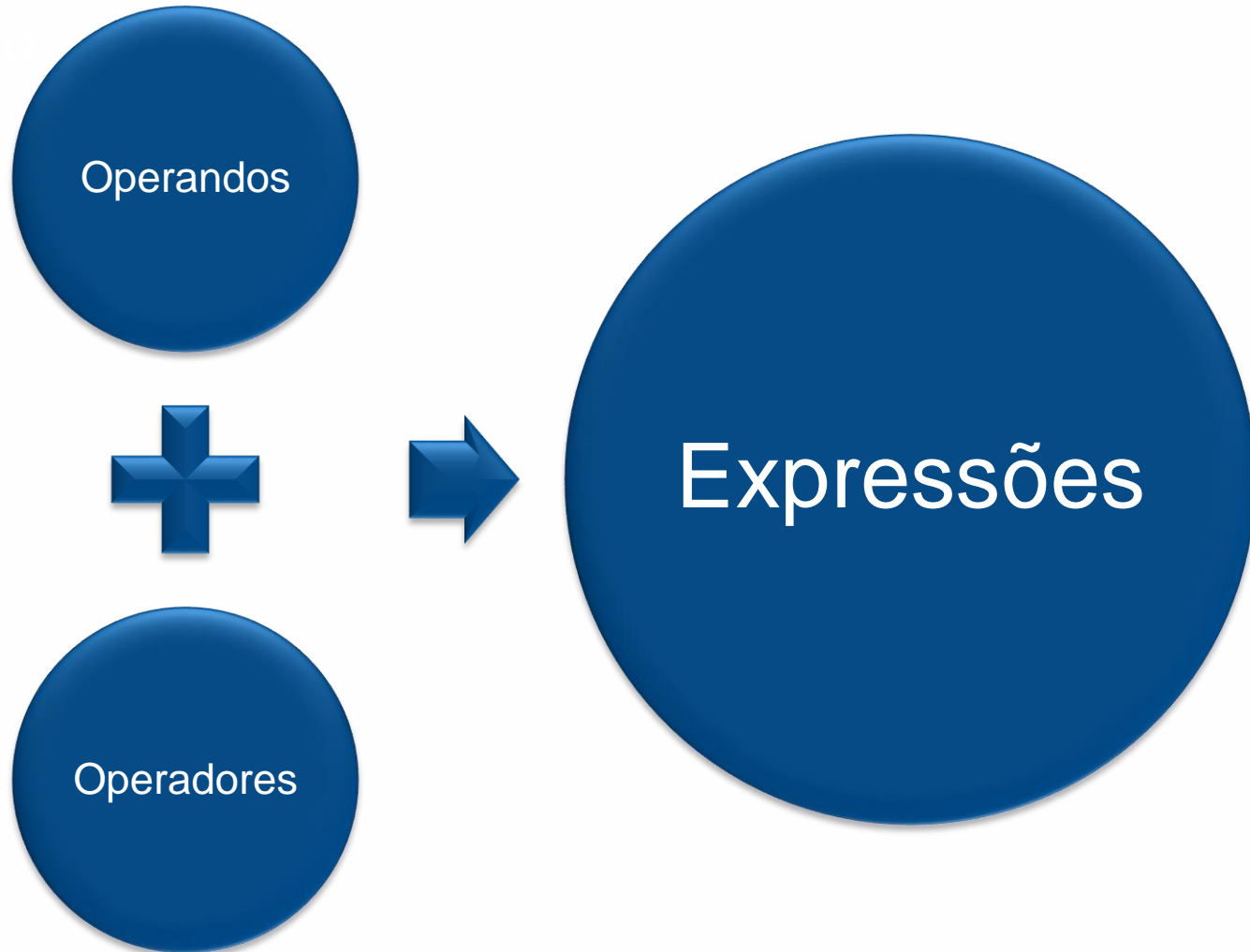
Deve estar claro também que sempre à esquerda do operador de atribuição deve haver uma, e somente uma, variável.

# Atribuição

```
algoritmo "Atribuições_Simples"  
var  
    numeroConta, numeroAgencia, digitoControle: inteiro  
    nome, sobrenome: caractere  
inicio  
    3063 := numeroConta  
    numeroAgencia + digitoControle := 1021 + 011  
    nome + sobrenome := "João" + "Silva"  
fimalgoritmo
```

Incorreto!

# Expressões



# Operandos

**São os elementos de uma expressão que sofrem uma ação.**

- **Exemplos:**
  - Variáveis;
  - Valores;
  - Outras expressões.

# Operadores

**São os elementos de uma expressão que realizam a ação.**

- **Exemplos:**
  - Operadores aritméticos;
  - Operadores relacionais;
  - Operadores lógicos.



# Expressões

```
algoritmo "Exemplo_Expressão"  
var  
    x: inteiro  
inicio  
    x := 3 + 2  
fimalgoritmo
```

- Na expressão  $x := 3 + 2$ , temos:

x  
3  
2

} Operandos

+  
:=

} Operadores

# Operadores Aritméticos

Operador Aritmético	Pseudocódigo
Adição	+
Subtração	-
Multiplicação	*
Divisão	/
Divisão inteira	\
Exponenciação	^
Módulo (resto da divisão)	%

# Operadores Aritméticos

Expressão	Resultado
$1 + 2$	3
$5.1 - 1$	4.1
$2 * 1.5$	3
$5 / 2$	2.5
$5 \setminus 2$	2
$8 ^ 2$	64
$10 \% 5$	0

# Operadores Aritméticos

Operador Aritmético	Prioridade
Exponenciação	3
Multiplicação	2
Divisão	2
Divisão inteira	2
Módulo (resto da divisão)	2
Adição	1
Subtração	1

**Obs.: para alterar a ordem de prioridade, utiliza-se parênteses.**

# Operadores Aritméticos

- Exemplo:

$$\begin{aligned} 2 + 12 / 2 * 3 \\ 2 + 6 * 3 \\ 2 + 18 \\ 20 \end{aligned}$$



$$\begin{aligned} ((2 + 12) / 2) * 3 \\ (14 / 2) * 3 \\ 7 * 3 \\ 21 \end{aligned}$$

# Expressões Lógicas

**Podem ser consideradas afirmações que serão testadas pelo computador.**

- Tendo como resultado:
  - verdadeiro
  - falso
- São utilizadas com os operadores relacionais e lógicos.

# Operadores Relacionais

Operador Relacional	Pseudocódigo
Maior	>
Menor	<
Maior ou igual	>=
Menor ou igual	<=
Igual	=
Diferente	<>

# Operadores Relacionais

Expressão	Resultado
$1 = 2$	falso
"A" = "a"	verdadeiro
$5 > 2$	verdadeiro
$3 \leq 3$	verdadeiro
$2 + 3 \neq 5$	falso



# Operadores Lógicos

Operador Lógico	Pseudocódigo
Conjunção	e
Disjunção	ou
Negação	nao

# Operadores Lógicos

- **e:**
  - Resulta verdadeiro se ambas as expressões forem verdadeiras.
- **ou:**
  - Resulta verdadeiro se ao menos uma das expressões for verdadeira.
- **nao:**
  - Nega a expressão, ou seja, inverte o valor.

# Operadores Lógicos

- Exemplo do operador e:

**$(n > 0) \text{ e } (n \% 2 = 0)$**

# Operadores Lógicos

- Exemplo do operador ou:

**$(i \geq 65)$  ou  $(t \geq 30)$**

# Operadores Lógicos

- Exemplo do operador nao:

**nao (n % 2 = 0)**

# Operadores Lógicos

- Tabela verdade do operador e:

A	B	A e B
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	falso
falso	verdadeiro	falso
falso	falso	falso

# Operadores Lógicos

- Tabela verdade do operador ou:

A	B	A ou B
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	verdadeiro
falso	verdadeiro	verdadeiro
falso	falso	falso

# Operadores Lógicos

- Tabela verdade do operador nao:

A	nao A
verdadeiro	falso
falso	verdadeiro

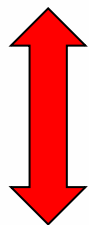


# Operadores

O software Visualg não possui relacionamento de categorias.

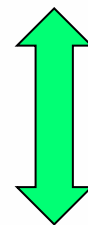
- Exemplos:

$2 * 5 > 3$  ou  $5 + 1 < 2$  e  $2 < 7 - 2$



Incorreto!

$(2 * 5 > 3)$  ou  $((5 + 1 < 2) \text{ e } (2 < 7 - 2))$



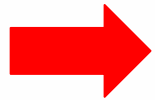
Correto!

# Comentários

**Os comentários são declarações não compiladas que podem conter qualquer informação textual para referência e documentação de seu programa.**

- São representados por duas barras normais “//”;
- Todo o texto inserido após as duas barras será comentário.

# Comentários



```
algoritmo "Olá"  
//Exemplo de comentário  
var  
    n: caractere  
inicio  
    escreva("Digite seu nome: ")  
    leia(n)  
    escreva("Olá ", n)  
fimalgoritmo
```

# Linearização de Expressões

**Todas as expressões aritméticas devem ser linearizadas, ou seja, colocadas em linhas, devendo também ser feito o mapeamento dos operadores da aritmética tradicional para os do pseudocódigo.**

- **Exemplo:**

Tradicional	Computacional
$\left\{ \left[ \frac{2}{3} - (5 - 3) \right] + 1 \right\} \cdot 5$	$((2 / 3 - (5 - 3)) + 1) * 5$

# Referência

- MANZANO, J. A. N. G; OLIVEIRA, J. F. **Algoritmos: lógica para desenvolvimento de programação de computadores**. 27<sup>a</sup> ed. São Paulo: Érica, 2014. (Capítulo 3).

