



EJA IV – Ensino Fundamental  
**Qualificação Profissional – Informática Básica**  
**Análise e Lógica da Programação**

# **Algoritmo: Formas de representação e refinamentos sucessivos**

Rildo Oliveira



05/03/2024

## ROTEIRO DE AULA

### **OBJETO DO CONHECIMENTO:**

Algoritmo: Formas de representação e refinamentos sucessivos

**HABILIDADE:** (EMIFFTP02) Levantar e testar hipóteses para resolver problemas do cotidiano pessoal, da escola e do trabalho, utilizando procedimentos e linguagens adequados à investigação científica.

### **OBJETIVOS:**

Apresentar as diferentes formas que podem ser representados um algoritmo, fluxograma e pseudocódigo;

Realizar exercícios que exemplifica como construir código.

**DA TEORIA À PRÁTICA:** Uso de imagens, texto e conceitos para um melhor entendimento do tema abordado.

# Pasta Compartilhada EJA



<https://github.com/rildexter/eja2024/tree/main>

# Algoritmo

Um algoritmo é considerado completo se os seus comandos forem do entendimento do seu destinatário.

Num algoritmo, um comando que não for do entendimento do destinatário terá que ser desdobrado em novos comandos, que constituirão um refinamento do comando inicial, e assim sucessivamente, até que os comandos sejam entendidos pelo destinatário.

# Algoritmo

Por exemplo, o algoritmo para calcular a média aritmética de dois números pode ser escrito da seguinte forma:

Algoritmo CALCULA\_MÉDIA

Início

Receba os dois números;

Calcule a média dos dois números;

Exiba o resultado;

Fim

# Algoritmo

Podemos desdobrar o comando

“Calcule a média dos dois números” em:

**Soma os dois números;**

**Divida o resultado por 2;**

Após esse refinamento, o algoritmo pode ser considerado completo, a menos que o destinatário não saiba fazer as operações de adição e divisão, ou não seja capaz de entender diretamente algum comando.

# Algoritmo

O algoritmo estando completo, podemos reescrevê-lo, inserindo o refinamento na posição do comando que foi refinado.

## Algoritmo **CALCULA\_MÉDIA**

Início

Receba os dois números;

Soma os dois números;

Divida o resultado por 2;

Exiba o resultado;

Fim

# Algoritmo

Reescrever um algoritmo completo, com os refinamentos sucessivos inseridos nos seus devidos lugares, permite ter uma visão global de como o algoritmo deve ser executado.

À medida que o algoritmo passa a ser maior e mais complexo, esta visão global torna-se menos clara e, neste caso, um algoritmo apresentado com os refinamentos sucessivos separados oferece uma melhor abordagem para quem precisar entendê-lo.



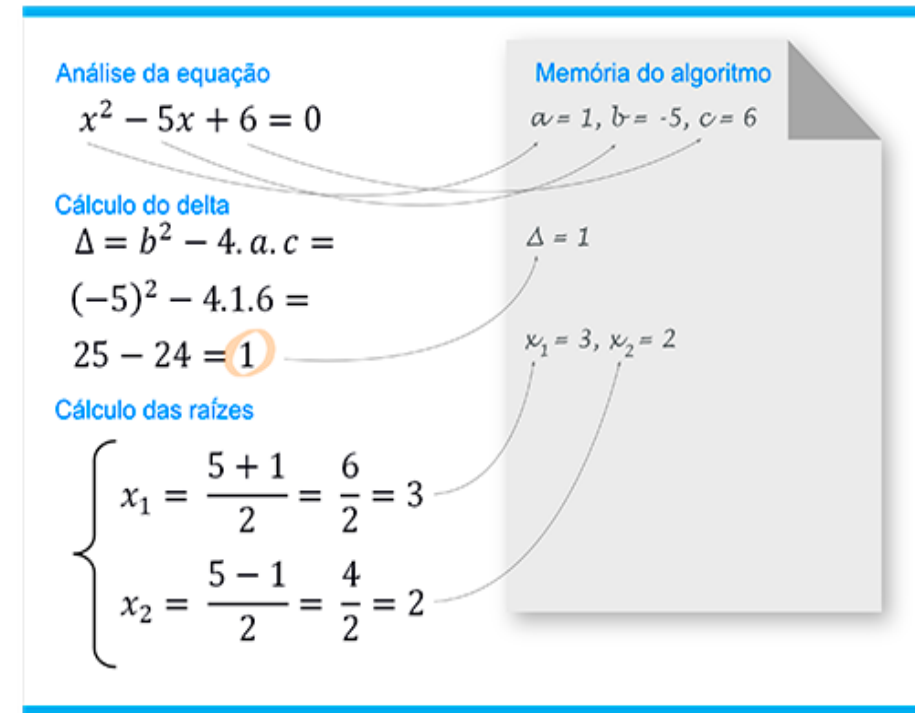
# LÓGICA?

Vamos definir o que é lógica: “A lógica é a arte de pensar corretamente ou a lógica é um estudo dos modos corretos do pensamento” (SOARES, 2014, p. 1).

## Você sabia?

Chamamos de memória o local onde guardamos os dados de entrada e que foram gerados durante o uso de um algoritmo.

A memória pode ser uma folha de papel, por exemplo, se estivermos executando um algoritmo manualmente, como no cálculo das raízes de uma equação do segundo grau da **figura** , a seguir:



# Vamos observar?

**Análise da equação**  
 $x^2 - 5x + 6 = 0$

**Cálculo do delta**  
 $\Delta = b^2 - 4.a.c =$   
 $(-5)^2 - 4.1.6 =$   
 $25 - 24 = 1$

**Cálculo das raízes**  
$$\begin{cases} x_1 = \frac{5 + 1}{2} = \frac{6}{2} = 3 \\ x_2 = \frac{5 - 1}{2} = \frac{4}{2} = 2 \end{cases}$$

**Memória do algoritmo**  
 $a = 1, b = -5, c = 6$   
 $\Delta = 1$   
 $x_1 = 3, x_2 = 2$

The diagram illustrates the process of solving a quadratic equation. It shows the initial equation, the calculation of the discriminant (delta), and the subsequent calculation of the roots. Arrows indicate the flow of information from the calculations to a 'Memory of the algorithm' box, which stores the coefficients (a, b, c), the discriminant (delta), and the roots (x1, x2). The value 1 in the discriminant calculation is highlighted with an orange circle.

# Memória

Já no computador, existem diversos componentes que fazem a função de memória: discos rígidos, pen drives, DVDs etc.

Porém, o componente que é considerado como memória principal é a memória RAM (Random-Access Memory, ou memória de acesso aleatório).

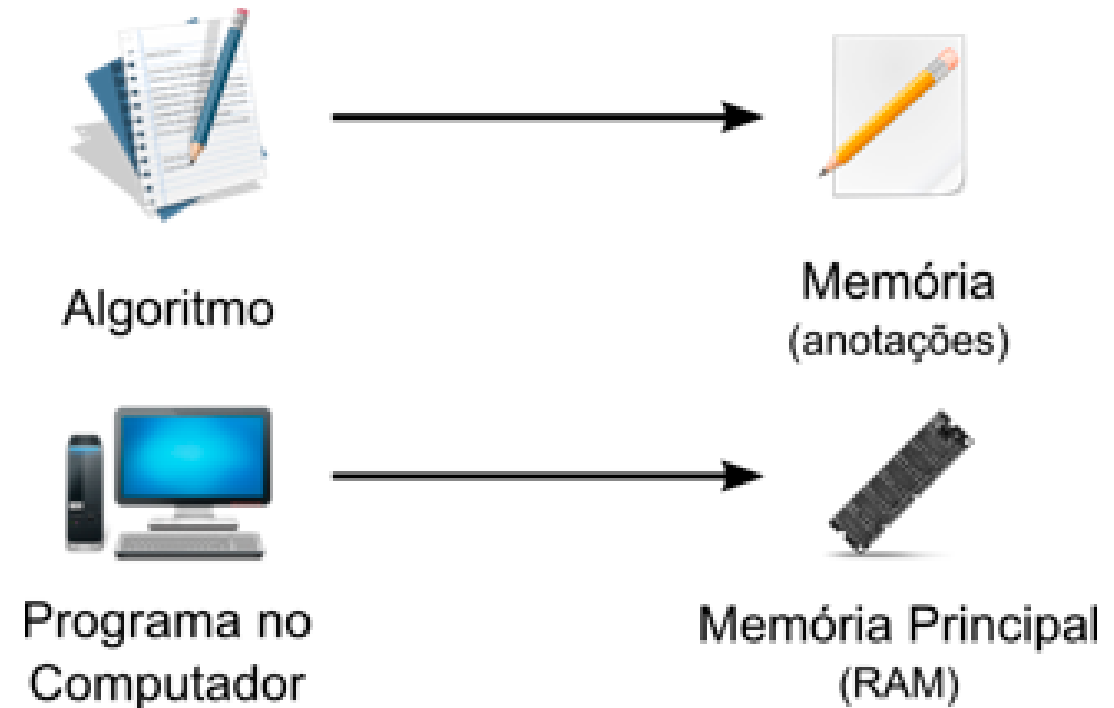
Esse componente é bastante rápido e trabalha diretamente ligado ao processador para prover os valores que os programas precisam no menor tempo possível.

# Memória

Cada dado na memória deve ter um identificador (um nome) e um tipo.

A memória funciona como uma grande tabela de consulta, na qual colocamos valores para uma consulta posterior.

Valores armazenados na memória também podem ser modificados durante o tempo.



# Tipos de dados básicos

O computador, para poder trabalhar com alguma destas informações, precisa saber onde, na memória, o dado está localizado.

Fisicamente, cada posição de memória, possui um endereço, ou seja, um número, que indica onde cada informação está localizada. este número é representado através da notação hexadecimal, tendo o tamanho de quatro, ou mais bytes. Abaixo segue alguns exemplos:

Endereço Físico	Informação
3000: B712	'João'
2000: 12EC	12345
3000: 0004	'H'

# Tipos de dados básicos

Sabendo que podemos armazenar diversos tipos de dados durante o uso de um algoritmo, tornou-se evidente a necessidade de padronizar os tipos de dados que poderiam ser usados na memória. Daí surgiram os tipos básicos de dados, também chamados de tipos primitivos.

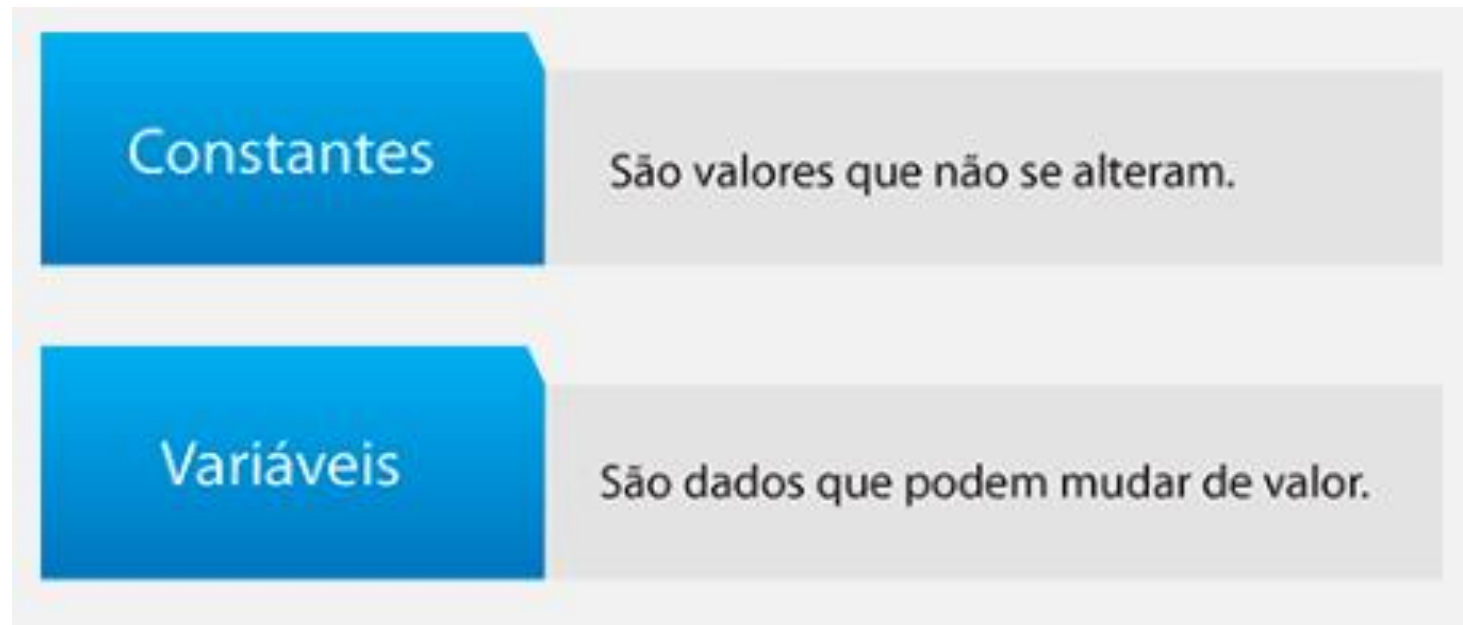
# Variáveis x Constantes

**Valores guardados** na memória são chamados de **variáveis**. Variáveis, como o próprio nome diz, podem variar, ou seja, ter seu valor modificado. A função das variáveis é manter o valor atual de algum dado relevante dentro do algoritmo e, por isso, podem ser modificadas com o tempo.



# Variáveis x Constantes

Para esses casos, chamamos os valores guardados de constantes, pois eles não precisam ser modificados durante toda execução do algoritmo. Em resumo:



# Diferenças entre tipos de variáveis

caractere ocupa 1 sizeof

Inteiro ocupa de 2 a 4 sizeof



# Palavras-reservadas (palavras-chave).

São identificadores predefinidos que possuem significados especiais para o interpretador do algoritmo.

inicio	senao	para	enquanto
var	logico	se	ate
faca	inteiro	real	

# Dados do tipo numérico

São variáveis que armazenam dados numéricos, como: **a idade** de uma pessoa, **o preço** de um produto, **o salário** de um funcionário, entre outros dados caracterizados pelos números.

# Ainda falando do tipo numérico

Podemos dividi-lo em duas classes:

- **Inteiro:** são caracterizados pelos números inteiros, positivos ou negativos. Exemplo: (110), (90), (−80), (−2).
- O tipo de dados int serve para guardar números inteiros, positivos e negativos.
- **Real:** são caracterizados por números inteiros e decimais (fracionais), sendo positivos ou negativos. Exemplo: (10,50), (−30,22), (20).

# Exemplo de Algoritmo - Dado do tipo inteiros

```
algoritmo "soma2inteiros"  
var  
    // Reservar 3 espaços de memória do tipo inteiro,  
    // chamados x, y e z.  
    x, y, z : inteiro  
inicio  
    //Lê os valores fornecidos pelo usuário e  
    //armazena em x e y  
    leia(x)  
    leia(y)  
    //O resultado da soma de x + y será armazenado no  
    //espaço de memória z  
    z <- x + y  
    //Apresenta a resposta (tela, impressora, arquivo, etc)  
    escreva(z)  
fimalgoritmo
```

# Dados tipo : real

Os dados de tipo real são aqueles que podem possuir componentes decimais ou fracionários, e podem também ser positivos ou negativos.

- Como dito anteriormente, os elementos dos conjuntos de números fracionários e reais são necessariamente representados nos computadores por dados do tipo real.
- **Exemplos de dados do tipo real:**
  - 24.01 - número real positivo com duas casas decimais;
  - 13.3 - número real negativo com uma casa decimal

# Exemplo de Algoritmo - Dado do tipo real

```
1 algoritmo "SomaNumero"
2 // Função : Somar dois valores e retornar um resultado
3 // Autor : Nathanael Bonfim
4 // Data : 7/3/2020
5
6 var
7   n1, n2, r: real // Declaração das variáveis
8 inicio
9   // Armazena a primeira variável
10  EscrevaL("Entre com o valor de n1:")
11  leia(n1)
12  // Armazena a segunda variável
13  EscrevaL("Entre com o valor de n2:")
14  leia(n2)
15  // Processa a soma e armazena o resultado em r
16  r <- n1 + n2
17  Escreva("A soma de n1 + n1 é igual a:", r)
18 fimalgoritmo
```



# Dados do tipo literal

- O tipo de dados literal é constituído por uma sequência de caracteres contendo letras, dígitos e/ou símbolos especiais. Este tipo de dados é também muitas vezes chamado de alfanumérico, cadeia (ou cordão) de caracteres, ou ainda, do inglês STRING.
- Usualmente, os dados literais são representados nos algoritmos pela coleção de caracteres, delimitada em seu início e término com o caractere aspas (").

# Dados do tipo lógico

Estes tipos de dados são chamados de booleanos, devido a significativa contribuição de BOOLE à área da lógica matemática.

- Representam de certa forma a maneira como os computadores funcionam..
- O tipo de dados lógico é usado para representar dois únicos valores lógicos possíveis: verdadeiro e falso.

# Exemplo de Algoritmo

```
1  algoritmo aprendendo_variaveis;  
2  variaveis  
3  nome, sobrenome: literal;  
4  idade: inteiro;  
5  salario: real;  
6  tem_filho: logico;  
7  inicio  
8  //aqui vem todos os comandos do meu algoritmo  
9  fim.
```

```
1 Algoritmo "ParOuImpar"  
2  
3 Var  
4  
5     numero : inteiro  
6  
7 Inicio  
8  
9     escreva("Escreva um número: ")  
10    leia(numero)  
11  
12 Fimalgoritmo
```

[illegible]

```

Início da execução
Escreva um número: 7654

Fim da execução.

```

## Quais Ferramentas devo ter em meu PC?

Para iniciar vamos usar o Visual G, pois estaremos aprendendo a lógica de programação através do português.

<https://visualg3.com.br/>

VisuALG



## Vamos visualizar a tela inicial do visualG

The screenshot displays the VISUALG 3.0.7.0 application window. The title bar indicates it's an "Interpretador e Editor de Algoritmos" (Interpreter and Algorithm Editor) with the last update on October 3, 2015, for the entity AESPI.

**Menu Bar:** Arquivo, Editar, Run (executar), Exportar para, Manutenção, Help (Ajuda).

**Toolbar:** Includes icons for file operations (new, open, save, print), editing (undo, redo, cut, copy, paste), execution (run, stop), and other utilities like a calculator and help.

**Main Area:** Titled "Área dos algoritmos [ Edição do código fonte ] -> Nome do arquivo: [semnome]". It contains the following algorithm code:

```

1 Algoritmo "semnome"
2 // Disciplina : [Lógica de Programação e Algoritmos]
3 // Professor : Rildo da Silva Oliveira
4 // Descrição : Aqui você descreve o que o programa faz! (função)
5 // Autor(a) : Nome do(a) aluno(a)
6 // Data atual : 16/08/2021
7 Var
8 // Seção de Declarações das variáveis
9
10
11 Inicio
12 // Seção de Comandos, procedimento, funções, operadores, etc...
13
14
15 Fimalgoritmo

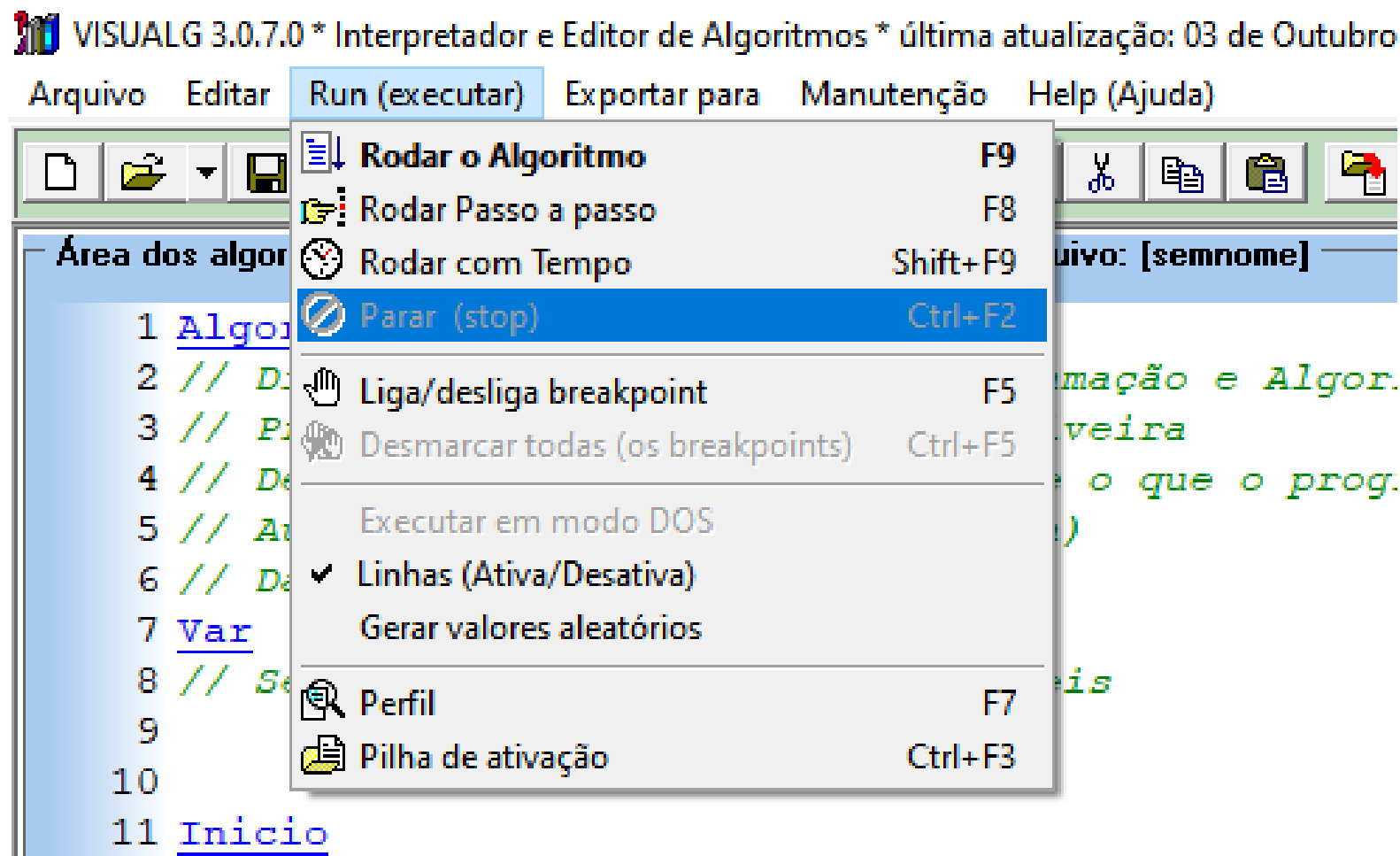
```

**Right Panel:**

- Áreas das variáveis de memória (Globais e Locais):** A table with columns: Escopo, Nome, Tipo, Valor. It is currently empty.
- Área de visualização dos resultados:** An empty area at the bottom right for displaying program output.

In the bottom right corner of the overall image, there is a page number "30".

## Compilar Projeto



## Nosso primeiro Projeto

```
1 Algoritmo "Ola mundo"  
2 // Disciplina   : [Lógica de Programação e Algoritmos]  
3 // Professor    : Rildo da Silva Oliveira  
4 // Descrição    : Aqui você descreve o que o programa faz! (função)  
5 // Autor(a)     : Nome do(a) aluno(a)  
6 // Data atual   : 16/08/2021  
7 Var  
8 // Seção de Declarações das variáveis  
9  
10  
11 Inicio  
12 // Seção de Comandos, procedimento, funções, operadores, etc...  
13 escreva("Olá mundo!")  
14  
15 Fimalgoritmo
```



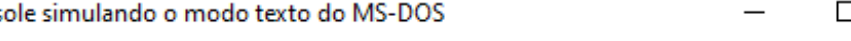


## Projeto: escreva

```
1 algoritmo "Escreva"  
2 // Disciplina : [Lógica de Programação e Algoritmos]  
3 // Professor : Rildo da Silva Oliveira  
4 // Autor :  
5 // Data :  
6 // Programa Base  
7  
8 var  
9     varNum1: inteiro  
10  
11 inicio  
12     varNum1 <- 2  
13  
14     escreva("A Variavel 'varNum1' contem: ", varNum1)  
15 fimalgoritmo  
16
```

## Projeto: escreva

```
1 algoritmo "Escreva"
2 // Disciplina : [Lógica de Programação e Algoritmos]
3 // Professor : Rildo da Silva Oliveira
4 // Autor :
5 // Data :
6 // Programa Base
7
8 var
9     varNum1: inteiro
10
11 inicio
12     varNum1 <- 2
13
14     escreva("A Variavel 'varNum1' contem: ", varNum1)
15 finalgoritmo
16
```



C:\> Console simulando o modo texto do MS-DOS

A Variavel 'varNum1' contem: 2

>>> Fim da execução do programa !

[illegible]

Área de visualização dos resultados

```
Início da execução
A Variável 'varNum1' contém: 2
Fim da execução.
```

## Projeto: tipos de variáveis


```
7 Var
8 // Seção de Declarações das variáveis
9 x: real
10 y: inteiro
11 z: logico
12 w: caractere
13
14 Inicio
15 // Seção de Comandos, procedimento, funções, operadores, etc...
16
17 x<-1
18 y<-1
19 z<-verdadeiro
20 w<-"Letras"
21 escreval ("o valor da variavel x",x )
22 escreval ("o valor da variavel y",y )
23 escreval ("o valor da variavel z",z )
24 escreval ("o valor da variavel w ",w )
25 Fimalgoritmo
```

rea dos programas [ Edição do código fonte ] -> Nome do arquivo: [UI-tipos de variaveis.ALB]

```

1 Algoritmo "TIPOS DE VARIÁVEIS"
2 // Disciplina : [Linguagem e Lógica de Programação]
3 // Professor : Rildo da Silva Oliveira
4 // Descrição : Aqui você descreve o que o programa faz! (função)
5 // Autor(a) : Nome do(a) aluno(a)
6 // Data atual : 04/08/2021
7 Var
8 // Seção de Declarações das variáveis
9 x: real
10 y: inteiro
11 z: logico
12 w: caractere
13 Inicio
14 // Seção de Comandos, procedimento, funções, operadores, etc...
15 x<-1
16 y<-1
17 z<-verdadeiro
18 w<-"Letras"
19 escreval ("o valor da variavel x",x )
20 escreval ("o valor da variavel y",y )
21 escreval ("o valor da variavel z",z )
22 escreval ("o valor da variavel w ",w )
23 Fimalgoritmo

```

 Console simulando o modo texto do MS-DOS

```
o valor da variavel x 1
o valor da variavel y 1
o valor da variavel z VERDADEIRO
o valor da variavel w Letras
```

```
>>> Fim da execução do programa !
```

### Áreas das variáveis de memória (Globais e Locais)

[illegible]

Área de visualização dos resultados

```
Início da execução
o valor da variável x 1
o valor da variável y 1
o valor da variável z VERDADEIRO
o valor da variável w Letras
```

Fim da execução.

## Projeto: escreva-leia-soma

```
1 algoritmo "escreva-leia-soma"  
2 // Disciplina : [Lógica de Programação e Algoritmos]  
3 // Professor : Rildo da Silva Oliveira  
4 // Autor :  
5 // Data :  
6 // Programa Base  
7 var  
8     varNum1, varNum2, soma: inteiro  
9  
10 inicio  
11     escreva("Digite o PRIMEIRO numero: ")  
12     leia (varNum1)  
13  
14     escreva("Digite o SEGUNDO numero: ")  
15     leia (varNum2)  
16  
17     soma <- varNum1 + varNum2  
18     escreval("Total: ", soma)  
19  
20 fimalgoritmo  
21
```

# Projeto: escreva-leia-soma

```

1 algoritmo "escreva-leia-soma"
2 // Disciplina : [Lógica de Programação e Algoritmos]
3 // Professor : Rildo da Silva Oliveira
4 // Autor :
5 // Data :
6 // Programa Base
7 var
8     varNum1, varNum2, soma: inteiro
9
10 inicio
11     escreva("Digite o PRIMEIRO numero: ")
12     leia (varNum1)
13
14     escreva("Digite o SEGUNDO numero: ")
15     leia (varNum2)
16
17     soma <- varNum1 + varNum2
18     escreval("Total: ", soma)
19
20 fimalgoritmo

```

[illegible]

Área de visualização dos resultados

```
Início da execução
Digite o PRIMEIRO numero: 5
Digite o SEGUNDO numero: 4
Total: 9

Fim da execução.
```

```
C:\> Console simulando o modo texto do MS-DOS

Digite o PRIMEIRO numero: 5
Digite o SEGUNDO numero: 4
Total: 9

>>> Fim da execução do programa !
```

# Dica da semana!

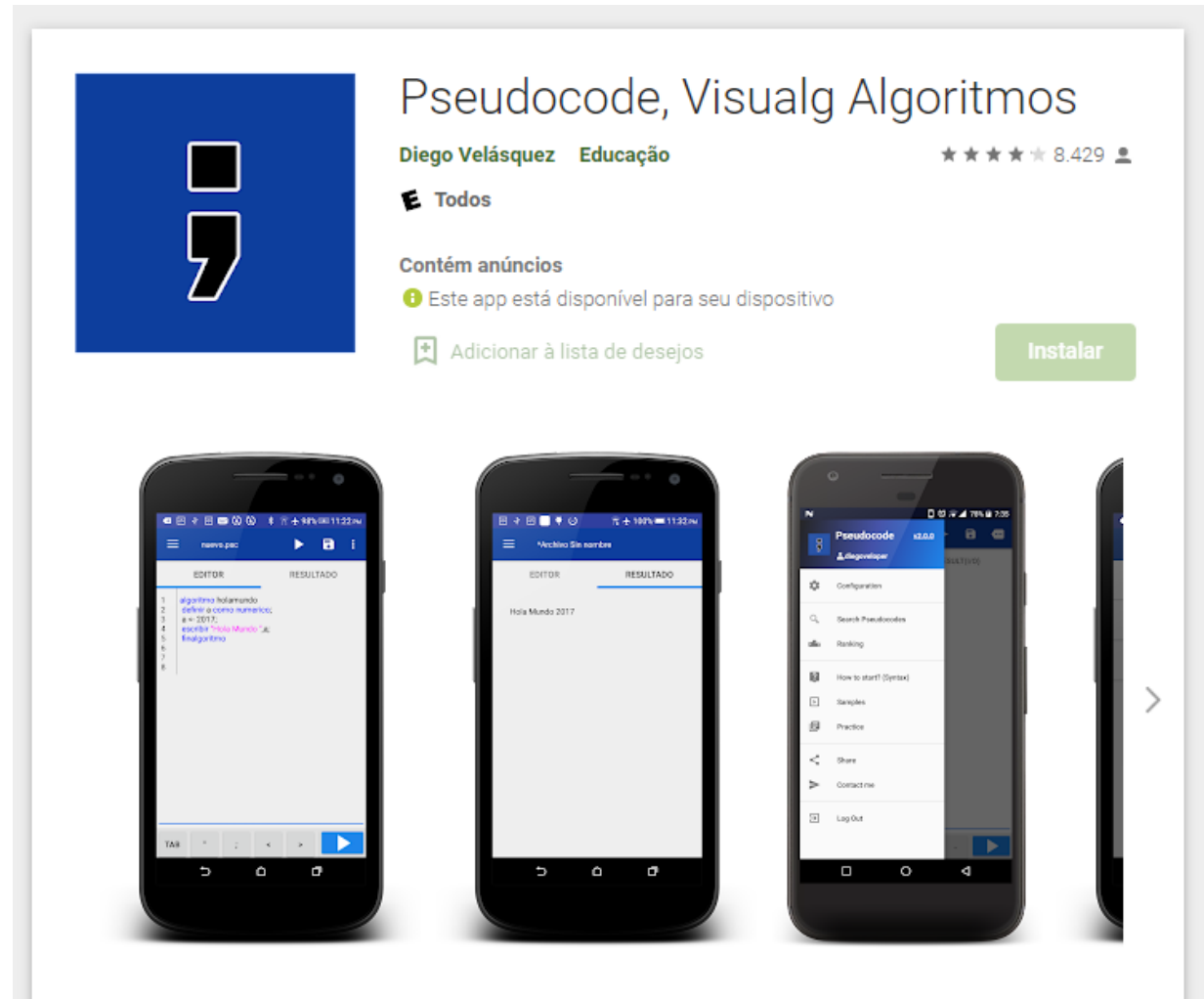
Site para criar algoritmos em portugol!

<https://portugol-webstudio.cubos.io/ide>





# Dica da semana!



# Referências

- CORMEN, Thomas H. **Desmistificando Algoritmos**. Rio de Janeiro: Elsevier, 2014.
- EDMONDS, Jeff. **Como pensar sobre algoritmos**. LTC, 2010.
- KLEINBERG, Jon e TARDOS, Éva. **Algorithm Design**. Pearson, 2005.



**ATÉ A PRÓXIMA AULA!**