

Conceitos Básicos de Algoritmos

Durante a execução de determinadas tarefas ou durante a execução de um processo para obtenção da solução de um determinado problema são necessários alguns dados e ao final deve-se retornar uma saída.

Da mesma forma, em alguns algoritmos tornam-se necessários mecanismos que possibilitem uma interface com o ambiente externo, ou seja, são necessários comandos que possibilitem a entrada e saída de dados.

Conceitos Básicos de Algoritmos

Entrada de dados

Palavra-reservada: leia - utilizada para receber dados externos ao algoritmo e armazená-los na memória, ou melhor, em variáveis.

Exemplos:

```
...  
Var    a: inteiro  
        b: real  
  
...  
        leia(a)  
  
...  
        leia(b, a)  
  
...
```

Conceitos Básicos de Algoritmos

Entrada de Dados

Tecnicamente podemos utilizar a seguinte definição: O comando de entrada de dados *leia* possui a sintaxe

`leia (<lista-de-variáveis>)`

Onde este recebe valores digitados pelo usuário, atribuindo-os às variáveis cujos indentificadores (nomes) estão em <lista-de-variáveis> (é respeitada a ordem especificada nesta lista).

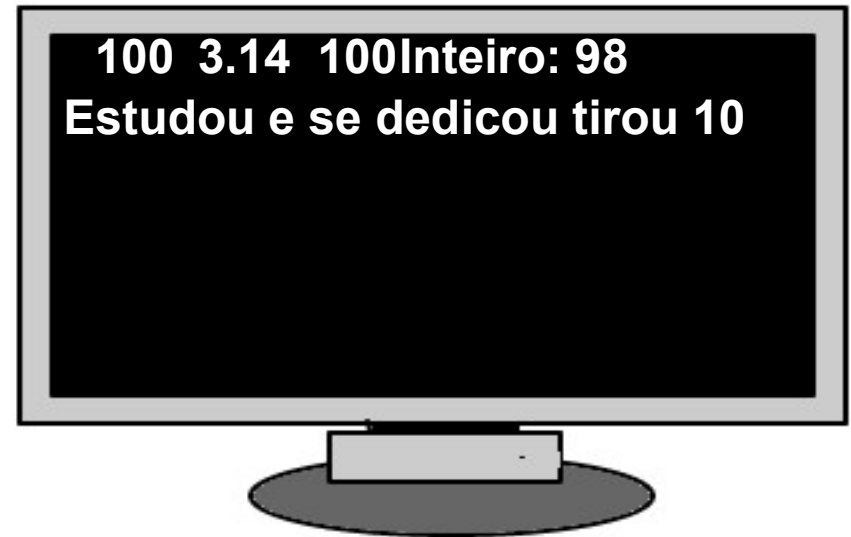
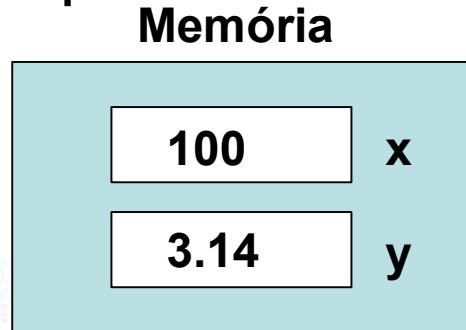
Conceitos Básicos de Algoritmos

Saída de dados

Palavra-reservada: escreva - utilizada para externar (enviar para o monitor) dados gerados pelo algoritmo.

Exemplos:

```
...  
Var  
  x: inteiro  
  y: real  
...  
leia (y,x)  
escreva (x)  
escreva (y,x)  
escreva ("Inteiro: ", x-2)  
escreva ("Estudou e se dedicou tirou ", 10)  
...
```



Conceitos Básicos de Algoritmos

Saída de Dados

Tecnicamente podemos utilizar a seguinte definição: O comando de saída de dados **escreva** possui a sintaxe

escreva (*<lista-de-expressões>*)

Onde este escreve no dispositivo de saída padrão (monitor) o conteúdo de cada uma das expressões que compõem *<lista-de-expressões>*. As expressões dentro desta lista devem estar separadas por vírgulas; depois de serem avaliadas, seus resultados são impressos na ordem indicada.

Conceitos Básicos de Algoritmos

Saída de dados

É possível especificar o número de colunas da tela que se deseja reservar para escrever um determinado valor. Por exemplo, considerando uma variável inteira x , o comando `escreva (x:5)` escreve o valor da variável x em 5 colunas, alinhado-o à direita.

Para variáveis reais, pode-se também especificar o número de casas fracionárias que serão exibidas.

Por exemplo, considerando y como uma variável real, o comando `escreva(y:6:2)`, escreve seu valor em 6 colunas colocando 2 casas decimais.

Conceitos Básicos de Algoritmos

Operador de Atribuição <-

Exemplo:

```
...  
> var  
  > a: inteiro  
  > valor1, valor2: real  
  > ...  
  > A <- 5  
  > valor1 <- 3.14  
  > valor2 <- valor1  
  > escreva(valor2)  
  > ...
```

Conceitos Básicos de Algoritmos

Operadores Aritméticos

Unários: +, -

Exemplos:

+1

-5.9

...

var a: inteiro

...

a <- -a

Conceitos Básicos de Algoritmos

Operadores Aritméticos

Binários: +, -, *, /, \, %, ^

Associação

<i>Símbolo</i>	<i>Operação</i>
+	Soma
-	Subtração
*	Multiplicação
/	Divisão
\	Quociente da divisão inteira
%	Resto da divisão inteira
^	Potenciação

Conceitos Básicos de Algoritmos

Operadores Aritméticos

Binários

Precedência (Hierarquia nas operações)

Hierarquia	Operação
1	Parênteses
2	Função
3	-, + (unários)
4	^
5	*, /, \, %
6	+, -

Conceitos Básicos de Algoritmos

Expressões aritméticas

Exemplos:

$$\begin{aligned}3/4+5 &= 5.75 \\3/(4+5) &= 0.33333333 \\3\sqrt{2*9} &= 9 \\11\%3^2 &= 2.0 \\11\%(3^2) &= 2.0 \\(11\%3)^2 &= 4.0 \\3\sqrt{2+(65-40)^{(1/2)}} &= 6.0\end{aligned}$$

**Observação: a potenciação gera um valor real!
Sendo assim, as expressões grifadas são inválidas.**

Conceitos Básicos de Algoritmos

Operadores Relacionais

Operador	Ação
>	maior que
>=	maior ou igual a
<	menor que
<=	menor ou igual a
=	igual a
<>	diferente de

Conceitos Básicos de Algoritmos

➤ Operadores Lógicos

Operador
e
ou
nao
xou

Conceitos Básicos de Algoritmos

Expressões lógicas

Exemplos:

$3 > 7 = \text{FALSO}$

“A” = “a” = VERDADEIRO

“a” > “B” = FALSO

“Ana” < “Aline” = FALSO

$(3 >= 13 \setminus 4) \text{ xou } (\text{nao } (5 \% 2 = 0)) = \text{FALSO}$

xou	V	F
V	F	V
F	V	F

Pseudocódigo

Pseudocódigo

Pseudocódigo é uma técnica textual de representação de um algoritmo. Também é conhecida como Português Estruturado ou Portugol.

Nele os verbos (ações) disponíveis para utilização são limitados e empregados no imperativo, deve-se evitar as expressões excessivamente longas, estas restrições visam eliminar a possibilidade de ambiguidade.

A técnica é baseada em uma PDL (Program Design Language), que é uma linguagem genérica na qual é possível representar um algoritmo de forma semelhante à das linguagens de programação.

Pseudocódigo – Estrutura

A estrutura de um algoritmo em pseudocódigo pode variar um pouco de acordo com o autor ou com base na linguagem de programação que será utilizada posteriormente, mas essas variações ocorrem apenas na sintaxe, pois a semântica deve ser exatamente a mesma.

A estrutura que empregaremos para a construção de nossos pseudocódigos será a seguinte:

```
-> algoritmo "nome" //Tem como objetivo identificar o algoritmo
-> var
    //Seção de Declarações
-> inicio
    // Seção de Comandos
-> fimalgoritmo //marca o final do algoritmo
```

VISUALG 

Pseudocódigo – Exemplo

Observaremos agora um pseudocódigo que recebe um valor inteiro, fornecido pelo usuário, e o retorna no monitor.

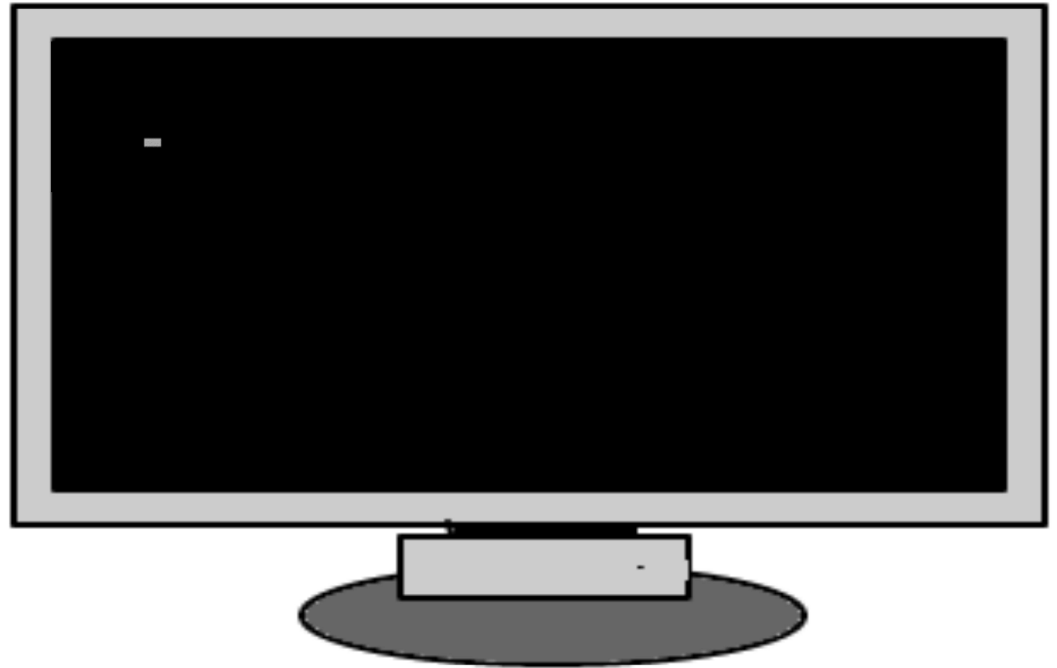
```
-> algoritmo "exemplo 1"  
-> var x: inteiro  
-> inicio  
->     leia (x)  
->     escreva (x)  
-> fimalgoritmo
```

Esta solução não é muito adequada.

Qual o motivo?

Pseudocódigo – Exemplo

```
➤ algoritmo "exemplo 1"  
➤ var x: inteiro  
➤ inicio  
➤     leia (x)  
     escreva (x)  
fimalgoritmo
```



Pseudocódigo – Exemplo

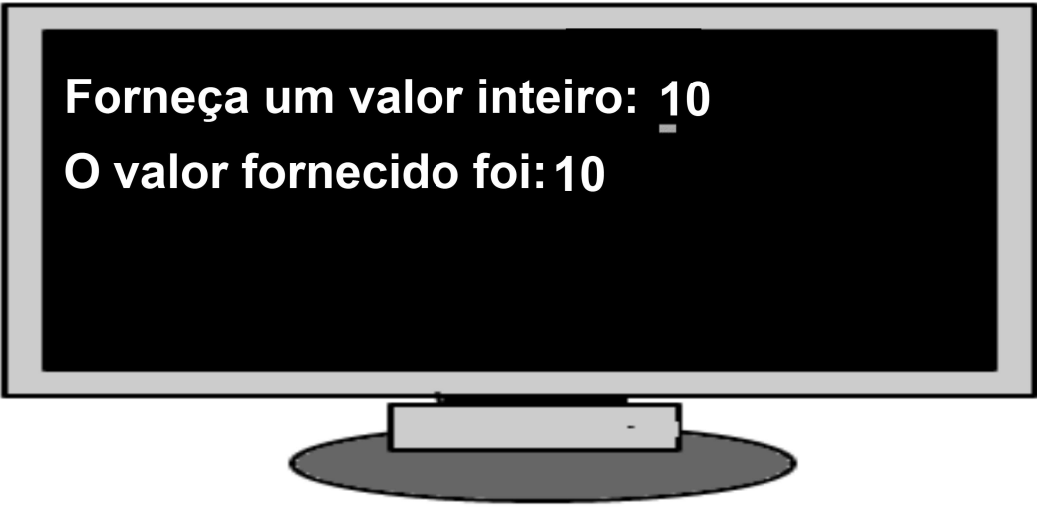
algoritmo "exemplo 1 ajustado"

var

x: inteiro

inicio

fimalgoritmo



Forneça um valor inteiro: 10
O valor fornecido foi: 10

Pseudocódigo – Exemplo

Outro exemplo, seria o pseudocódigo de um algoritmo que recebe um valor inteiro, acresce duas unidades a este, e exhibe o resultado desta manipulação na saída padrão.



Pseudocódigo – Exemplo

Uma variante em pseudocódigo do algoritmo visto anteriormente seria:

```
algoritmo "exemplo 2.2"  
var  
    n: inteiro  
inicio  
    escreva ("Digite um número inteiro: ")  
    leia (n)  
    n ← n+2  
    escreva ("Resultado (número + 2): ", n)  
fimalgoritmo
```

Pseudocódigo – Exemplo

Outra variante em pseudocódigo do algoritmo visto anteriormente seria:

```
algoritmo "exemplo 2.3"  
var  
    n: inteiro  
inicio  
    escreva ("Digite um número inteiro: ")  
    leia (n)  
    escreva ("Resultado (número + 2): ", n+2)  
fimalgoritmo
```

Método para Construção de Algoritmos

Os passos necessários para a construção de um algoritmo são:

- ler atentamente o enunciado do problema, compreendendo-o e destacando os pontos mais importantes;
- definir os dados de entrada, ou seja, quais dados serão fornecidos;
- definir os dados de saída, ou seja, quais dados serão gerados depois do processamento;
- definir o processamento, ou seja, quais cálculos serão efetuados e quais as restrições para esses cálculos. O processamento é responsável pela obtenção dos dados de saída com base nos dados de entrada;
- definir as variáveis necessárias para armazenar as entradas e efetuar o processamento;
- elaborar o algoritmo;
- testar o algoritmo realizando simulações.

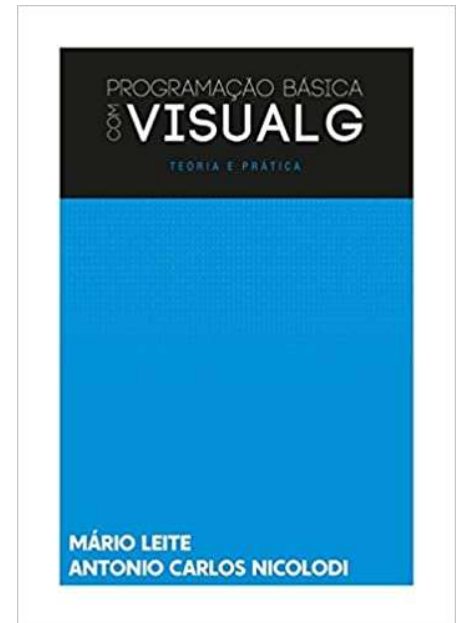
Pseudocódigo – VisuAlg

VisuAlg

O VisuAlg é um programa que permite criar, editar, interpretar e que também executa os algoritmos em português (português estruturado) como se fosse um “programa” normal de computador.

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

Livro: Programação Básica Com VisuAlg
setembro 2020





Área dos algoritmos (Edição do código fonte) -> Nome do arquivo: [semnome]

```

1 Algoritmo "semnome"
2 // Disciplina : Algoritmo e Programação para Computação
3 // Professor : Marcelo Linder
4 // Descrição : Aqui você descreve o que o programa faz! (função
5 // Autor(a) : Nome do(a) aluno(a)
6 // Data atual : 11/03/2021
7 Var
8 // Seção de Declarações das variáveis
9 |
10
11 Início
12 // Seção de Comandos, procedimento, funções, operadores, etc...
13
14
15 Fimalgoritmo

```

Áreas das variáveis de memória (Globais e Locais)

Escopo	Nome	Tipo	Valor

Área de visualização dos resultados

Empty area for displaying results.

Pseudocódigo - Exercícios

Parte 1

Pseudocódigo – Exercício 1

Construa um pseudocódigo para representar um algoritmo que efetue a multiplicação de dois inteiros quaisquer.

Os passos necessários para a construção de um algoritmo são:

- ler atentamente o enunciado do problema, compreendendo-o e destacando os pontos mais importantes;
- definir os dados de entrada, ou seja, quais dados serão fornecidos;
- definir os dados de saída, ou seja, quais dados serão gerados depois do processamento;
- definir o processamento, ou seja, quais cálculos serão efetuados e quais as restrições para esses cálculos. O processamento é responsável pela obtenção dos dados de saída com base nos dados de entrada;
- definir as variáveis necessárias para armazenar as entradas e efetuar o processamento;
- elaborar o algoritmo;
- testar o algoritmo realizando simulações.

Pseudocódigo – Exercício 1

A solução apresentada tem variantes, por exemplo:

```
algoritmo " exercício 1.2 "  
var  
    n1, n2: inteiro  
inicio  
    escreva ("Digite o multiplicando inteiro: ")  
    leia (n1)  
    escreva ("Digite o multiplicador inteiro: ")  
    leia (n2)  
    escreva ("Resultado da multiplicação: ", n1*n2)  
fimalgoritmo
```

Pseudocódigo – Exercício 1

Outra variante da solução seria:

```
algoritmo " exercício 1.3 "  
var  
    n1, n2: inteiro  
inicio  
    escreva ("Digite o multiplicando e em seguida ")  
    escreva (" o multiplicador (ambos inteiros): ")  
    leia (n1, n2)  
    escreva ("Resultado da multiplicação: ", n1*n2)  
fimalgoritmo
```

Pseudocódigo – Exercício 1

Temos mais um exemplo de variante da solução:

```
algoritmo "exercício 1.4"  
var  
    n1, n2: inteiro  
inicio  
    escreva ("Digite o multiplicando e em seguida ")  
    escreva (" o multiplicador (ambos inteiros): ")  
    leia (n1, n2)  
    escreva (n1, "*", n2, "=", n1*n2)  
fimalgoritmo
```


Pseudocódigo – Exercício 2

Gere um pseudocódigo que aplique um desconto de 30% sobre o valor de um produto, recebido como entrada, e retorne o resultado da manipulação na saída padrão.

```
algoritmo "exercício 2.1"  
var  
    valor, resultado: real  
inicio  
    escreva ("Entre com o valor do produto: ")  
    leia (valor)  
    resultado <- valor * 0.7  
    escreva ("Valor com desconto de 30%: ", resultado)  
fimalgoritmo
```

Pseudocódigo – Exercício 2

Primeira solução alternativa.

algoritmo "exercício 2.2"

var valor: real

 19.99

inicio

 escreva ("Entre com o valor do produto: ")

 leia (valor)

 valor <- valor * 0.7

 escreva ("Valor com desconto de 30%: ", valor:6:2)

fimalgoritmo

Pseudocódigo – Exercício 2

Segunda solução alternativa.

```
algoritmo "exercício 2.3"  
var valor: real  
inicio  
    escreva ("Entre com o valor do produto: ")  
    leia (valor)  
    escreva ("Valor com desconto de 30%: ")  
    escreva (valor * 0.7:6:2)  
fimalgoritmo
```

Pseudocódigo – Exercício 3

Faça um pseudocódigo de um algoritmo que receba 4 notas, calcule e mostre a média aritmética entre elas.