

# Algoritmos - 1

Alexandre Diehl

Departamento de Física - UFPel

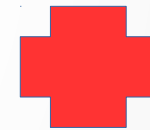
# Conceitos básicos

Computador: conceito clássico (até quando????)



**HARDWARE**

(partes físicas)

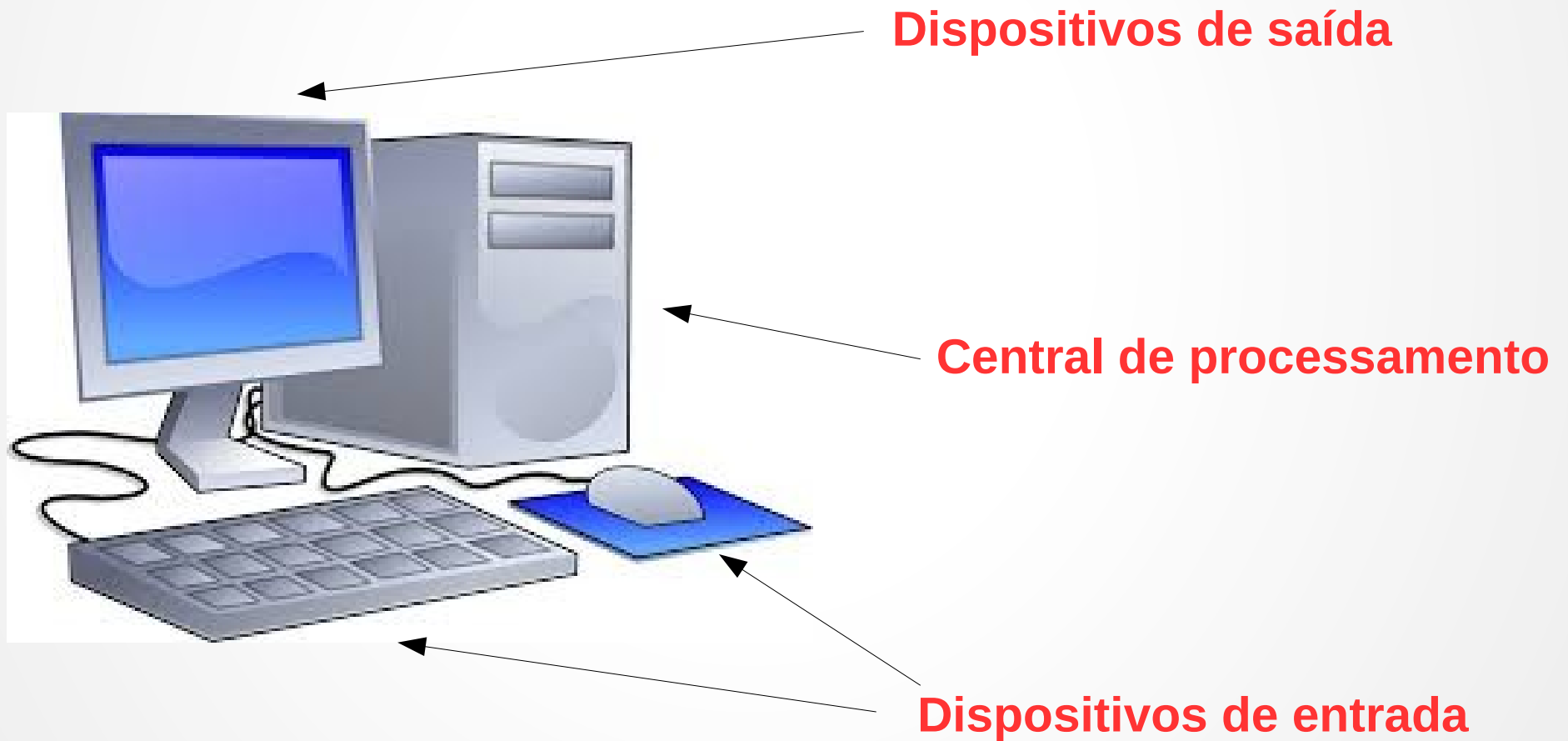


**SOFTWARE**

(programas)

# Conceitos básicos

## Computador: conceito clássico (até quando????)



SCEF

# Conceitos básicos

## Computador: conceito clássico (até quando????)



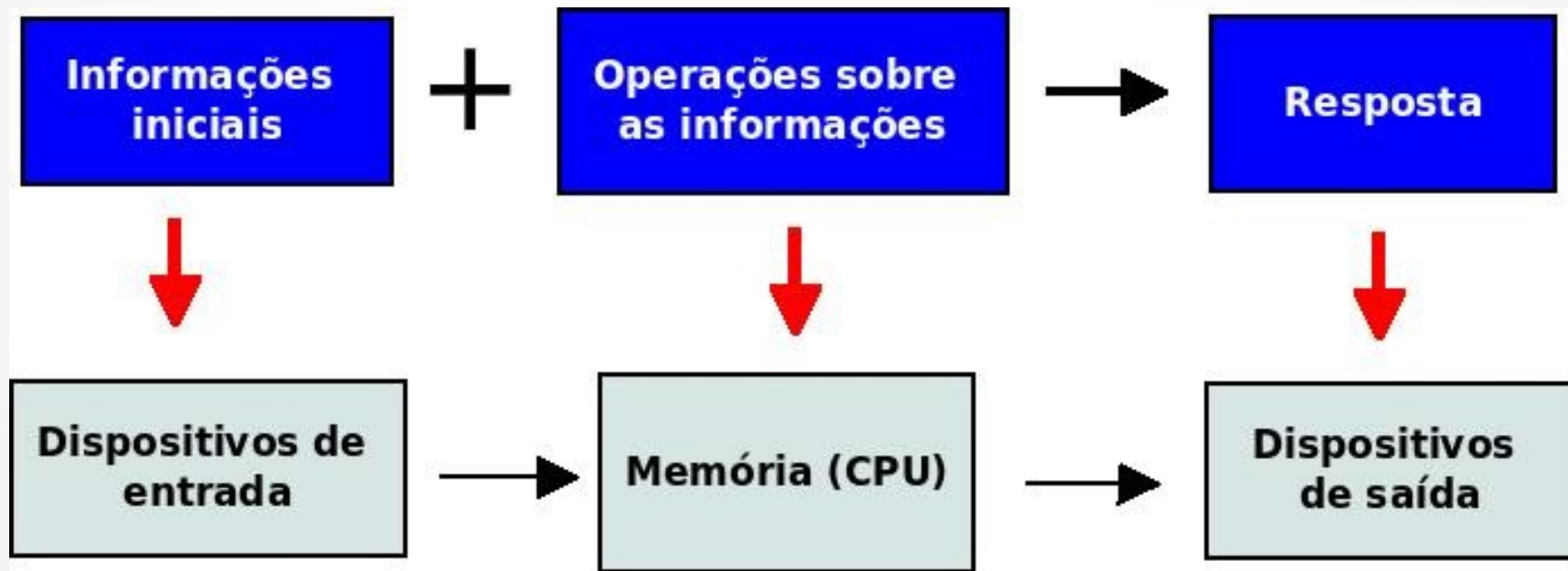
A finalidade de um computador é receber, manipular e armazenar dados.

A finalidade de um computador é realizar o **processamento de dados**.

# Conceitos básicos

Computador: conceito clássico (até quando????)

Processamento de dados



Fonte: Ascencio e Campos

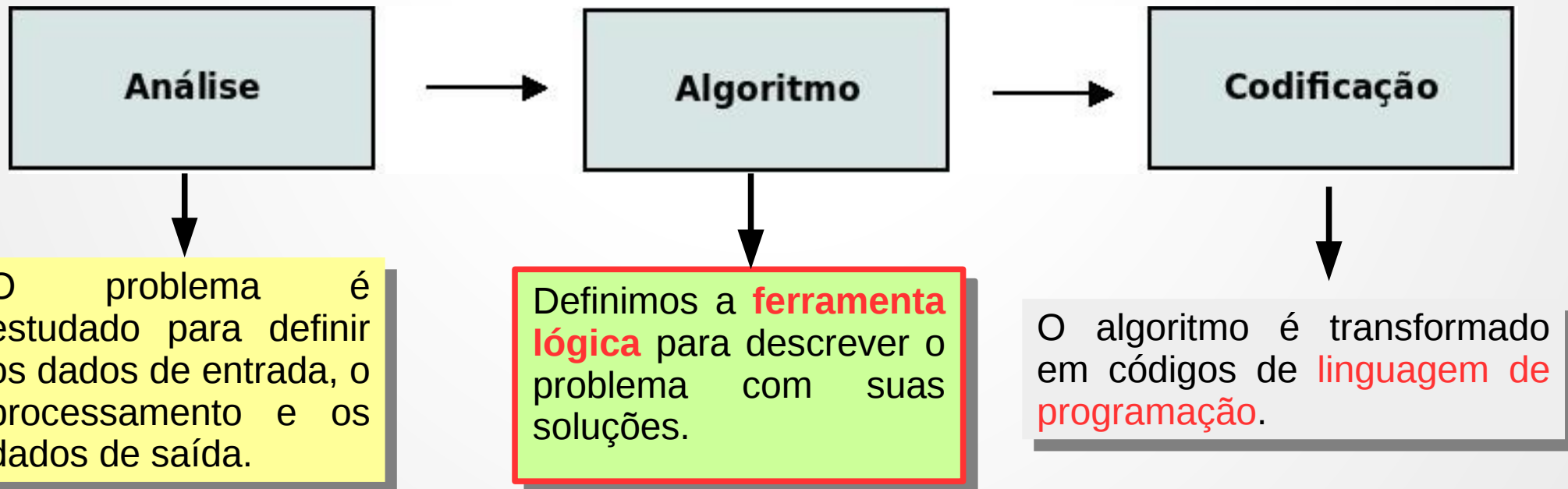
# Conceitos básicos

Computador: conceito clássico (até quando????)

## Processamento de dados

Realizado através da execução de um **programa computacional** ou vários programas interligados.

## Etapas para o desenvolvimento de um programa



# Algoritmo: Definição 1

**ALGORITMO** é uma sequência ordenada e finita de operações para a realização de uma tarefa qualquer (Ascencio, 1999).

## Exemplo: Realização de um experimento de Física Experimental 1

**Passo 1:** Reunir os equipamentos necessários para o experimento.

**Passo 2:** Montar o aparato experimental.

**Passo 3:** Realizar o experimento.

**Passo 4:** Fazer a análise dos dados experimentais.

**Passo 5:** Fazer o relatório do experimento.

**Passo 6:** Entregar o relatório.

# Algoritmo: Definição 2

**ALGORITMO** é uma sequência ordenada e finita de instruções ou operações para a solução de um problema computacional.

**Exemplo: Calcular a média das idades dos alunos da turma**

**Passo 1:** Reunir (ou obter) as idades dos alunos.

**Passo 2:** Calcular a média das idades.

**Passo 3:** Apresentar o resultado do cálculo da média.



# Algoritmo: Definição 3

**ALGORITMO** é a descrição de um conjunto de comandos que, obedecidos, resultam numa sucessão finita de ações.

**AÇÃO** é um acontecimento que, a partir de um estado inicial, após um período de tempo finito, produz um estado final previsível e bem definido (Farrer, 1999).

Todo **ALGORITMO** tem um início bem definido e um fim após um período finito de tempo.

# Algoritmo: Definição 3

**ALGORITMO** é a descrição de um conjunto de comandos que, obedecidos, resultam numa sucessão finita de ações.

**AÇÃO** é um acontecimento que, a partir de um estado inicial, após um período de tempo finito, produz um estado final previsível e bem definido (Farrer, 1999).

O **ALGORITMO** não é a solução do problema, mas sim a forma ou o meio de obtê-la.

# Algoritmo: Definição 3

**ALGORITMO** é a descrição de um conjunto de comandos que, obedecidos, resultam numa sucessão finita de ações.

**AÇÃO** é um acontecimento que, a partir de um estado inicial, após um período de tempo finito, produz um estado final previsível e bem definido (Farrer, 1999).

Não existe uma única forma para um **ALGORITMO**: um problema pode ser resolvido de várias maneiras.

# Formas de representação

## Descrição narrativa

A sequência de ações para a solução do problema é montada usando uma **linguagem natural** (língua portuguesa, por exemplo).

- **Vantagem:** não é necessário aprender nenhum conceito novo, pois a língua natural já é bem conhecida.
- **Desvantagem:** a língua natural abre espaço para várias interpretações, o que posteriormente dificultará a transcrição do algoritmo para uma linguagem de programação propriamente dita (Fortran, C, C++).

# Formas de representação

## Fluxogramas

Forma gráfica de apresentação dos algoritmos, onde as ações são representadas por formas geométricas predefinidas.

- **Vantagem:** o entendimento de elementos gráficos é mais fácil que o entendimento de textos.
- **Desvantagem:** é necessário aprender a simbologia dos fluxogramas e, além disso, o algoritmo resultante não apresenta detalhes, dificultando a sua transcrição para um programa em linguagem computacional.

# Formas de representação

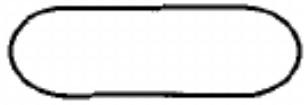
## Pseudocódigo

Também chamada de linguagem estruturada, reúne um conjunto de ações (ou comandos), usando uma linguagem simples e genérica, não associada com uma linguagem de programação específica.

- **Vantagem:** a passagem do algoritmo para qualquer linguagem de programação é quase imediata, bastando conhecer as palavras reservadas da linguagem de programação que será utilizada.
- **Desvantagem:** é necessário aprender as regras do pseudocódigo.

# Fluxogramas

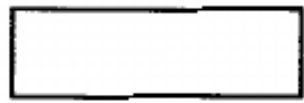
## Símbolos gráficos mais usados



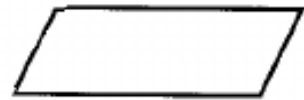
Símbolo utilizado para indicar o início e o fim do algoritmo.



Permite indicar o sentido do fluxo de dados. Serve exclusivamente para conectar os símbolos ou blocos existentes.



Símbolo utilizado para indicar cálculos e atribuições de valores.



Símbolo utilizado para representar a entrada de dados.



Símbolo utilizado para representar a saída de dados.



Símbolo que indica que deve ser tomada uma decisão, indicando a possibilidade de desvios.

# Exemplos

**Exemplo 1:** Faça um algoritmo para mostrar o resultado da multiplicação de dois números.

## Descrição narrativa

**Passo 1:** Receber (ou obter) os dois números que serão multiplicados.

**Passo 2:** Multiplicar os números.

**Passo 3:** Mostrar o resultado obtido na multiplicação.

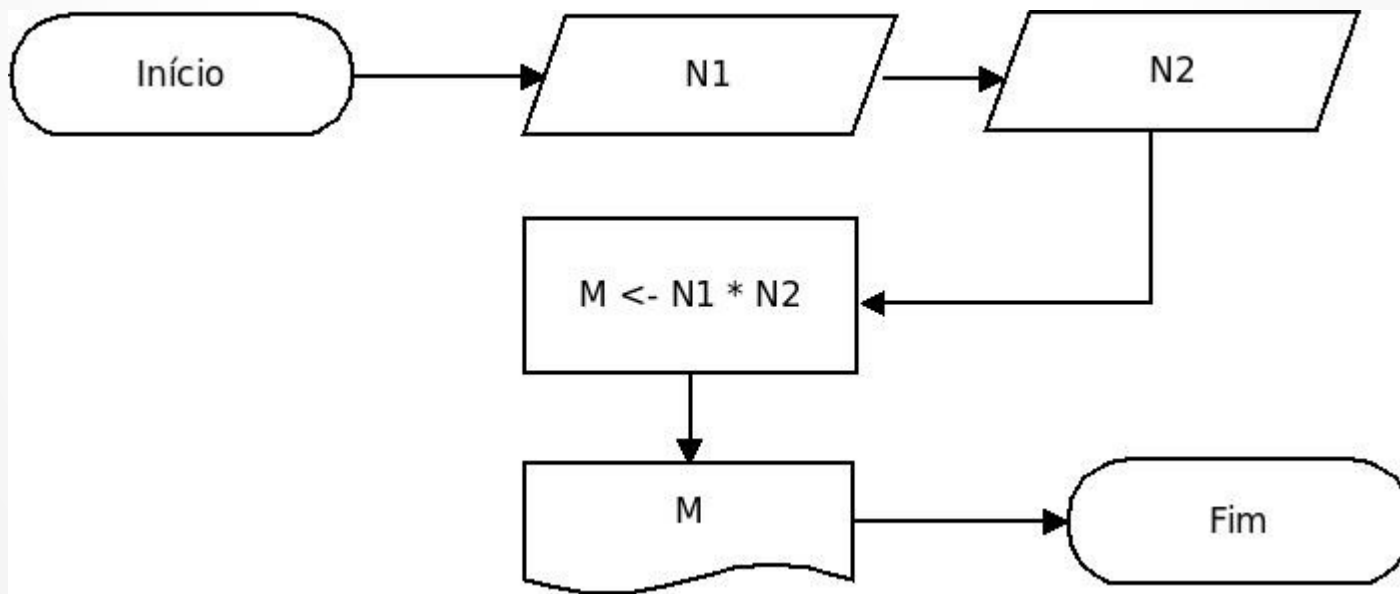


# Exemplos

**Exemplo 1:** Faça um algoritmo para mostrar o resultado da multiplicação de dois números.

## Fluxograma

Gerado com programa **dia** (linux)



SCEF

# Exemplos

**Exemplo 2:** Faça um algoritmo para mostrar o resultado da divisão de dois números.

## Descrição narrativa

**Passo 1:** Receber os dois números que serão divididos.

**Passo 2:** Se o segundo número for igual a zero, não poderá haver divisão, pois não existe divisão por zero; caso contrário, dividir os números.

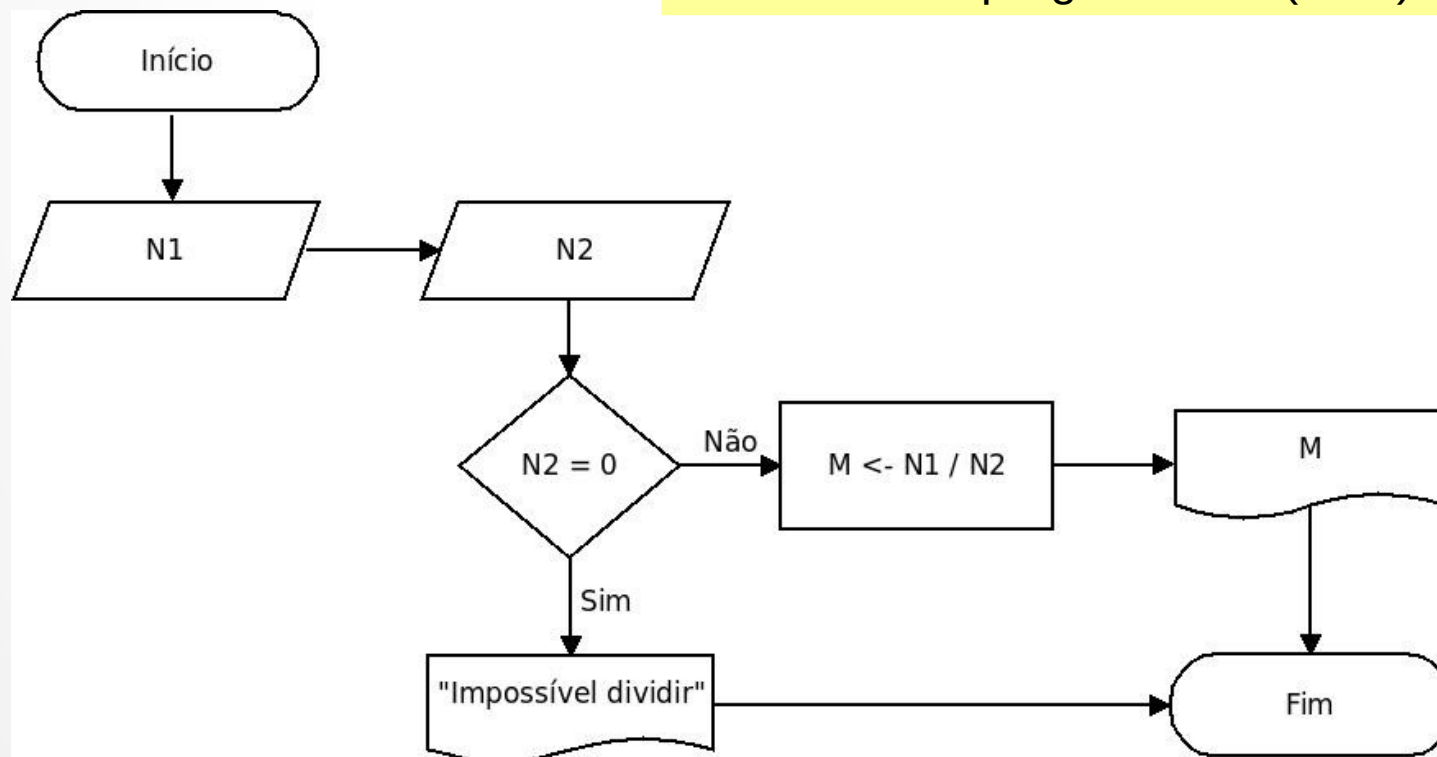
**Passo 3:** Mostrar o resultado obtido na divisão.

# Exemplos

**Exemplo 2:** Faça um algoritmo para mostrar o resultado da divisão de dois números.

## Fluxograma

Gerado com programa **dia** (linux)



# Pseudocódigo

## Pseudocódigo

Também chamada de linguagem estruturada, reúne um conjunto de ações (ou comandos), usando uma linguagem simples e genérica, não associada com uma linguagem de programação específica.

## PORTUGOL

**Pseudo linguagem** ou **método de codificação em língua portuguesa** de um algoritmo.

Acrônimo formado pelas palavras **PORTU**guês + al**GO**l + Pasca**L**, também conhecido como **português estruturado**, criado por Antonio Carlos Nicolodi e Antonio Manso em 1986.

# Pseudocódigo

## PORTUGOL

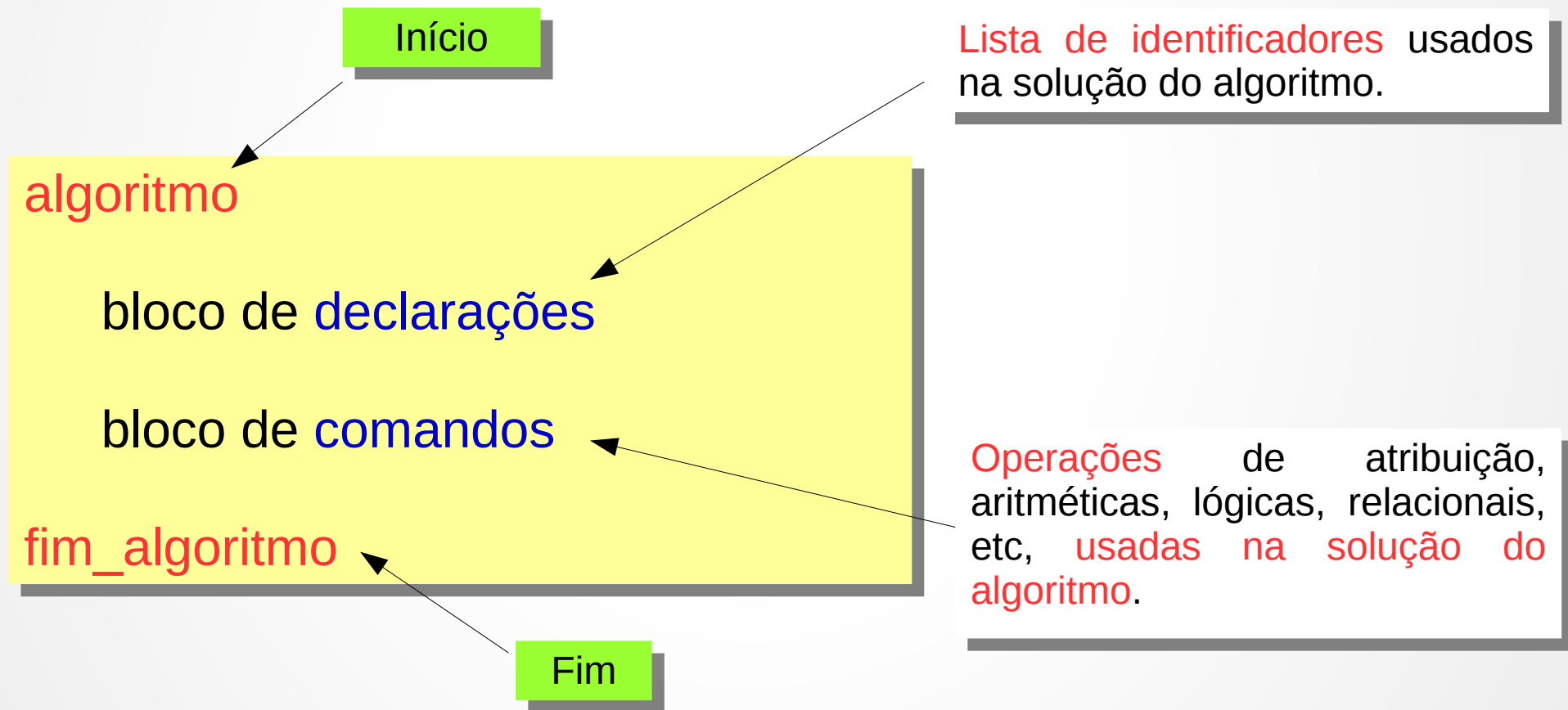
Por não se tratar de uma linguagem de programação propriamente dita, existem variações na codificação do Portugol.

- **Portugol Studio** (<http://lite.acad.univali.br/portugol/>)
- **VisualG** (<http://visualg3.com.br/>)
- **Portugol IDE** (<http://www.dei.estt.ipt.pt/portugol/>)
- **Portugol Online** (<https://vinyanalista.github.io/portugol/>)

**Ambiente de desenvolvimento integrado** (IDE) de algoritmos, criado por Medeiros e Soares, que usa a codificação do Portugol proposta por Ascencio e Campos em 2007.

# Pseudocódigo

## Formato de um pseudocódigo usando Portugol Online



# Pseudocódigo

## Estruturas básicas num pseudocódigo

- Identificador

**Nome dado** para uma constante ou variável.

### Regras para criação de identificadores

- Podem ser usados **números**, **letras** (maiúsculas ou minúsculas) e o **caractere** **\_**.
- Deve começar por uma **letra** (maiúscula ou minúscula) ou pelo **caractere** **\_**.

*underscore  
ou traço  
interior*

Portugol é **case sensitive**: letra maiúscula é diferente de letra minúscula

**Exemplo:** identificador com dado literal **Maria** é diferente de outro com dado literal **maria**

# Pseudocódigo

## Estruturas básicas num pseudocódigo

- **Identificador**

**Nome dado** para uma constante ou variável.

### Regras para criação de identificadores

- Não podem ser usados símbolos como \$, #, !, ?, &, +, -
- Não podem ser usados espaços em branco.
- Não pode ser usado o caractere – (hífen).
- Não podem ser usadas palavras reservadas do pseudocódigo (leia, escreva, repita, etc)

# → cerquilha ou number sign ou hash ou pound sign  
& → ampersand ou e comercial ou eitza



# Pseudocódigo

## Estruturas básicas num pseudocódigo

- **Constantes e Variáveis**

Espaços reservados na memória do computador para armazenar elementos de um certo conjunto ou tipo de dados.

- **Expressões**

Durante a execução do algoritmo, combinam os valores armazenados nas variáveis e constantes para calcular novos valores.

# Pseudocódigo

## Estruturas básicas num pseudocódigo

- **Constante**

Durante a execução do algoritmo, o **valor fixo da constante não muda** ao longo do tempo.

- **Variável**

Durante a execução do algoritmo, o **valor (ou conteúdo) da variável pode mudar** ao longo do tempo.

# Pseudocódigo

## Tipos de dados (Constantes ou Variáveis)

### Numérico

→ Usa o sistema decimal, podendo ser um número sem (**inteiro**) ou com (**real**) a parte fracionária (identificada por um ponto).

### Lógico

→ O dado só pode ser **verdadeiro** ou **falso**.

### Literal

→ O dado pode ser qualquer sequência de **caracteres** (letras, números ou símbolos especiais), **definidos entre aspas** (exemplo: "a", "bola").

# Pseudocódigo

## Declaração de Constantes ou Variáveis

Uma vez definida uma variável ou constante, o identificador usado só pode armazenar dados de um mesmo tipo.

**declare** <lista de identificadores> **tipo de dado**

Palavra reservada

Relação de constantes ou variáveis que definem os dados, separadas por vírgula.

Palavra-chave que identifica o tipo de dado: **numérico**, **lógico** ou **literal**.

# Pseudocódigo

## Declaração de Constantes ou Variáveis

A declaração do identificador apenas define o tipo de dado.

Não é atribuído nenhum valor ao identificador nesta etapa.

**declare** <lista de identificadores> **tipo de dado**

**algoritmo**

**declare** i, j, k **numerico**  
flag **logico**  
nome **literal**

**fim\_algoritmo**

} **Bloco de declarações**

# Pseudocódigo

## Declaração de Constantes ou Variáveis

Podem ser usados **comentários** na declaração dos identificadores.

O comentário é iniciado por //

**declare** <lista de identificadores> **tipo de dado**

**algoritmo**

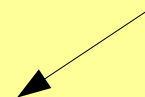
**declare** i, j, k **numerico**

flag **logico**

nome **literal** // nome do aluno

**fim\_algoritmo**

Comentário



# Pseudocódigo

## Operadores

**Meios pelos quais se realizam operações** sobre as variáveis e constantes.

### Tipos de operadores

- ✓ Operadores de **atribuição**
- ✓ Operadores **aritméticos**
- ✓ Operadores **relacionais**
- ✓ Operadores **lógicos**

# Pseudocódigo

## Operadores de atribuição

Usados para **atribuir valores para os identificadores** ou operações entre eles.

Usamos o símbolo  $\leftarrow$  para atribuir um valor ao identificador.

**algoritmo**

**declare** x **numerico**  
          y **literal**  
          teste **logico**

x  $\leftarrow$  4  
x  $\leftarrow$  x + 2  
y  $\leftarrow$  "aula"  
teste  $\leftarrow$  falso



**Bloco de atribuições**

**fim\_algoritmo**



# Pseudocódigo

## Operadores aritméticos

Usados para **operações com valores numéricos** entre constantes e variáveis.

### Básicos

Menor  prioridade  ↓  Maior	Operador	Símbolo	Exemplo
	soma	+	$a \leftarrow b + c$
	subtração	-	$a \leftarrow b - c$
	multiplicação	*	$a \leftarrow b * c$
	divisão	/	$a \leftarrow b / c$

# Pseudocódigo

## Operadores aritméticos

Usados para **operações com valores numéricos** entre constantes e variáveis.

### Pré-definidos

Função	Descrição	Exemplo
arredonda(x)	Retorna o <b>inteiro mais próximo</b> do número real x	$i \leftarrow \text{arredonda}(1.7)$
parte_inteira(x)	Retorna a <b>parte inteira</b> do número real x	$i \leftarrow \text{parte\_inteira}(2.4)$
resto(x,y)	Retorna o resto da divisão do número x pelo número y	$r \leftarrow \text{resto}(8,3)$

# Pseudocódigo

## Operadores aritméticos

Usados para **operações com valores numéricos** entre constantes e variáveis.

### Pré-definidos

Função	Descrição	Exemplo
seno(x)	Retorna o seno do ângulo x (expresso em radianos)	ang ← seno(3.1415)
cosseno(x)	Retorna o cosseno ângulo x (expresso em radianos)	ang ← cosseno(4.1415)

# Pseudocódigo

## Operadores aritméticos

Usados para **operações com valores numéricos** entre constantes e variáveis.

### Pré-definidos

Função	Descrição	Exemplo
potencia(a,b)	Retorna o número a elevado ao número b	$p \leftarrow potencia(4,2)$
raiz_quadrada(x)	Retorna a raiz quadrada do número x	$r2 \leftarrow raiz\_quadrada(16)$
raiz_enesima(n,x)	Retorna a raiz n do número x	$r3 \leftarrow raiz\_enesima(3,8)$

# Pseudocódigo

## Operadores relacionais

Usados na **comparação entre valores ou expressões**, retornando como resultado um valor lógico (falso ou verdadeiro).

Operador	Símbolo	Exemplo
Igual	=	a = b
Maior que	>	a > b
Menor que	<	a < b
Maior ou igual a	>=	a >= b
Menor ou igual a	<=	a <= b
Diferente de	<>	a <> b

# Pseudocódigo

## Operadores lógicos

**Relacionam entre si valores ou expressões lógicas**, resultando em valores lógicos (falso ou verdadeiro).

Operador	Símbolo	Exemplo
<b>Conjunção:</b> a conjunção de a e b é verdadeira se e somente se a e b são ambos verdadeiros.	<b>e</b>	a <b>e</b> b
<b>Disjunção:</b> a disjunção de a e b é verdadeira se e somente se, pelo menos, um deles é verdadeiro.	<b>ou</b>	a <b>ou</b> b
<b>Negação:</b> resulta na inversão do valor de a: se a é verdadeiro, resulta em falso; se a for falso, resulta em verdadeiro	<b>nao</b>	<b>nao</b> a

# Pseudocódigo

## Prioridade entre os operadores



$(a + b) > (c + d) \text{ e } (e = 0)$

SCEF

# Pseudocódigo

## Prioridade entre os operadores

### Algoritmo

**declare** x, y, z **numerico**  
          nome **literal**  
          flag **logico**

x ← 2  
y ← 5  
z ← 9  
nome ← "Maria"  
flag ← falso

**fim\_algoritmo**

Qual é o resultado desta operação?

**( x + y > z ) e ( nome = "Maria" )**

2 + 5  
└───┘

7 > 9  
└───┘

falso e verdadeiro

O resultado é falso



# Pseudocódigo

## Prioridade entre os operadores

### Algoritmo

```
declare x, y, z numerico  
       nome literal  
       flag logico
```

```
x ← 2  
y ← 5  
z ← 9  
nome ← "Maria"  
flag ← falso
```

```
fim_algoritmo
```

Qual é o resultado desta operação?

( flag ) ou ( y >= x )

5 >= 2



falso ou verdadeiro



O resultado é verdadeiro

# Pseudocódigo

## Prioridade entre os operadores

Qual é o resultado desta operação?

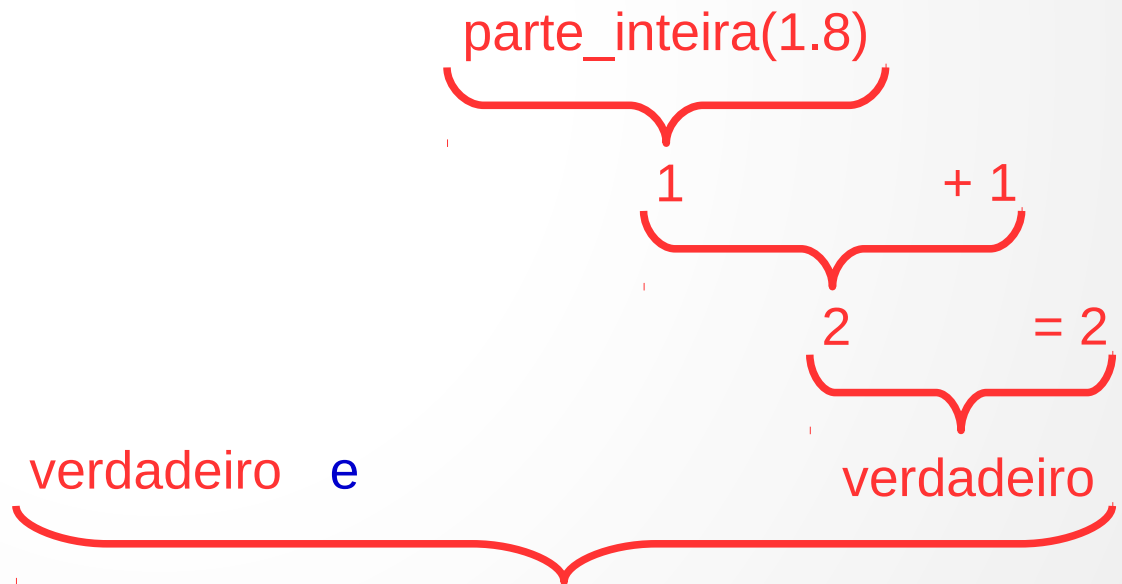
### Algoritmo

```
declare x, y, z numerico  
       nome literal  
       flag logico
```

```
x ← 2  
y ← 5  
z ← 9  
nome ← "Maria"  
flag ← falso
```

```
fim_algoritmo
```

( **nao** flag ) e ( parte\_inteira(z / y) + 1 = x )



O resultado é verdadeiro

# Pseudocódigo

## Prioridade entre os operadores

### Algoritmo

**declare** a, b, c **numerico**  
teste **logico**

a ← 1  
b ← 4.0  
c ← 8  
teste ← verdadeiro

**fim\_algoritmo**

Qual é o resultado desta operação?

( **nao** teste ) **ou** ( resto(b,2) = 0.0 )



O resultado é verdadeiro

# Pseudocódigo

## Prioridade entre os operadores

### Algoritmo

**declare** a, b, c **numerico**  
teste **logico**

a ← 1  
b ← 4.0  
c ← 8  
teste ← verdadeiro

**fim\_algoritmo**

Qual é o resultado desta operação?

( teste ) **e** ( **nao** teste )

verdadeiro **e** falso

O resultado é falso

# Pseudocódigo

## Prioridade entre os operadores

### Algoritmo

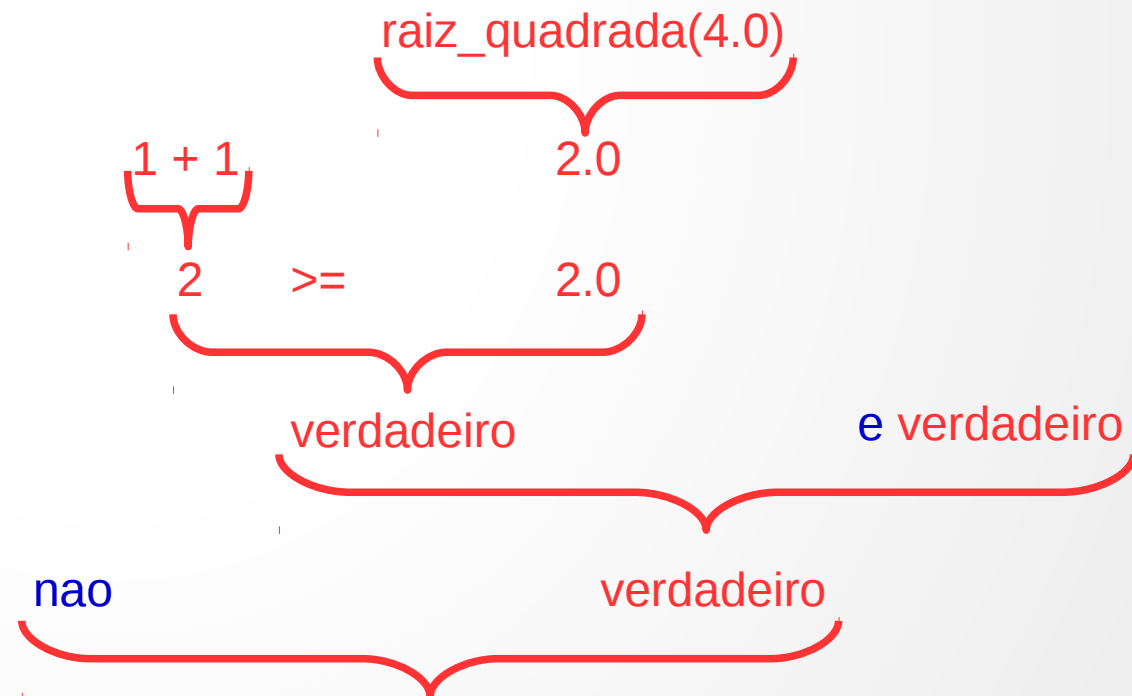
**declare** a, b, c **numerico**  
teste **logico**

a ← 1  
b ← 4.0  
c ← 8  
teste ← verdadeiro

**fim\_algoritmo**

Qual é o resultado desta operação?

**nao** ( a + 1 >= raiz\_quadrada(b) e teste )



O resultado é falso

# Pseudocódigo

## Entrada de dados num pseudocódigo

Definida pelo comando **leia** seguido do **nome do identificador** que recebe o valor de entrada.

**leia** nome do identificador

Cada instrução **leia** corresponde a uma entrada de dados através do dispositivo de entrada (teclado).

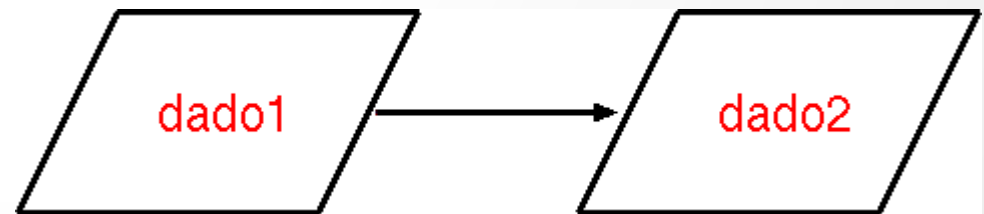
**algoritmo**

**declare** dado1, dado2 **numerico**

**leia** dado1

**leia** dado2

**fim\_algoritmo**



# Pseudocódigo

## Entrada de dados num pseudocódigo

Definida pelo comando **leia** seguido dos **nomes dos identificadores** que recebem os valores de entrada.

**leia** identificador\_1, identificador\_2, ..., identificador\_n

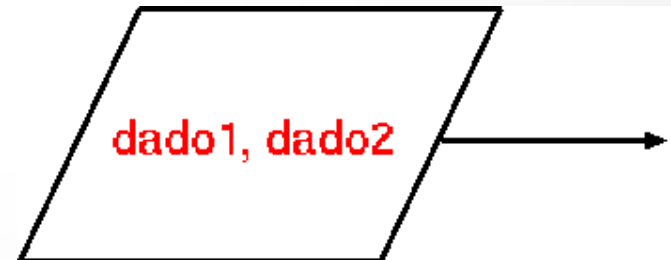
Os dados são **fornecidos um de cada vez**, através do dispositivo de entrada (teclado).

**algoritmo**

**declare** dado1, dado2 **numerico**

**leia** dado1, dado2

**fim\_algoritmo**



# Pseudocódigo

## Saída de dados num pseudocódigo

Definida pelo comando **escreva** seguido do **nome do identificador** que terá seu valor de saída apresentado.

**escreva** nome do identificador

Os dados são apresentados através do dispositivo de saída (monitor), com cada instrução escreva em uma linha diferente.

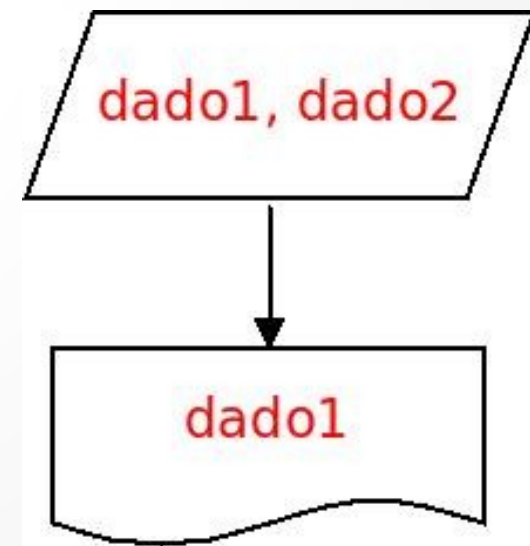
**algoritmo**

**declare** dado1, dado2 **numerico**

**leia** dado1, dado2

**escreva** dado1

**fim\_algoritmo**





# Pseudocódigo

## Saída de dados num pseudocódigo

Pode ser usado **texto na saída dos dados**. O **texto** deve ser colocado **entre aspas**, separado por uma vírgula do nome do identificador.

**escreva** "texto", nome do identificador

**algoritmo**

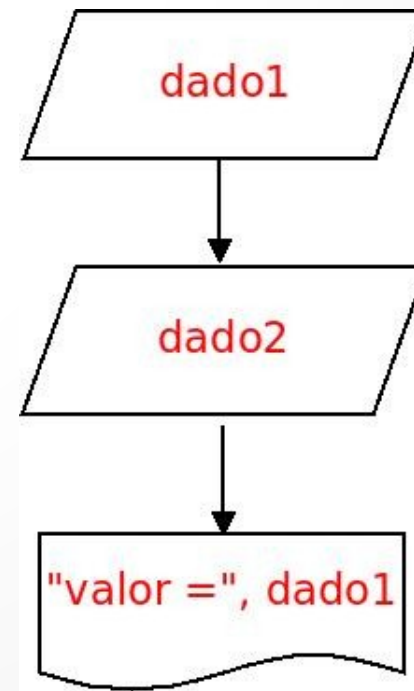
**declare** dado1, dado2 **numerico**

**leia** dado1

**leia** dado2

**escreva** "valor =", dado1

**fim\_algoritmo**



# Pseudocódigo

## Saída de dados num pseudocódigo

Podem ser apresentados **mais de um dado de saída**, com cada um dos dados separados por vírgula.

**escreva** identificador\_1, identificador\_2,..., identificador\_n

### algoritmo

**declare** dado1, dado2 **numerico**

**leia** dado1, dado2

**escreva** "valor 1 =", dado1, " valor 2 =", dado2

**fim\_algoritmo**

