



# Programação e Sistemas de Informação

CURSO PROFISSIONAL TÉCNICO DE  
GESTÃO E PROGRAMAÇÃO DE SISTEMAS INFORMÁTICOS

## Introdução à Programação e Algoritmia

v2.1

### MÓDULO 1

Professor: João Martiniano

# O conceito de algoritmo

- Pode ser definido como:

“ Conjunto de instruções que permitem atingir um objetivo ”

- Ou, segundo o dicionário online Infopédia:

“ Conjunto de regras e operações que permitem resolver, num número finito de etapas, um problema ”

in Infopédia [em linha]: Porto Editora, <http://www.infopedia.pt/lingua-portuguesa/algoritmo>

# O conceito de algoritmo

- Basicamente, qualquer programa ou aplicação, informático (quer corra em computadores, telemóveis, ou tablets) é composto por vários algoritmos
- É importante que, antes que um programa seja escrito numa linguagem de programação, os algoritmos sejam analisados e planeados
- Exemplos de algoritmos:
  - uma receita culinária
  - ordenar alfabeticamente um conjunto de nomes
  - determinar numa lista de temperaturas, qual a temperatura mais alta
  - calcular a classificação de um aluno

# Tipos de algoritmos

- Um algoritmo pode ser definido num linguagem natural ou numa linguagem artificial
- Ou seja, um algoritmo pode ser expresso de várias formas, até utilizando símbolos
- Normalmente os algoritmos são definidos numa de quatro formas:
  - descrição narrativa
  - descrição detalhada em linguagem natural
  - pseudocódigo
  - fluxograma

# Tipos de algoritmos

- Exemplo de algoritmo em **linguagem natural**

## Trocar uma lâmpada fundida

1. Preparar lâmpada nova
2. Retirar lâmpada fundida
3. Colocar lâmpada nova

# Tipos de algoritmos

- Exemplo de algoritmo em **pseudocódigo**

## Calcular um preço com desconto

Ler Preço

Ler Desconto

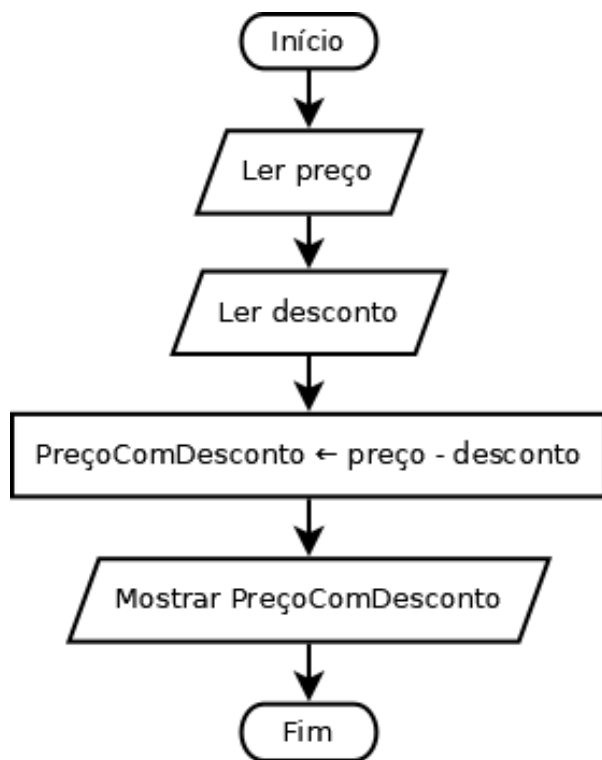
PreçoComDesconto  $\leftarrow$  Preço - Desconto

Escrever PreçoComDesconto

# Tipos de algoritmos

- Exemplo de algoritmo em **fluxograma**

## Calcular preço com desconto



# Tipos de algoritmos

- É importante salientar que um algoritmo pode ser mais ou menos detalhado
- Por exemplo, o algoritmo para troca de uma lâmpada pode ser refinado:

## Trocar uma lâmpada fundida (**versão 1**)

1. Preparar lâmpada nova
2. Retirar lâmpada fundida
3. Colocar lâmpada nova



# Tipos de algoritmos

- É importante salientar que um algoritmo pode ser mais ou menos detalhado
- Por exemplo, o algoritmo para troca de uma lâmpada pode ser refinado:

## Trocar uma lâmpada fundida (**versão 2**)

1. **Desligar interruptor**
2. Preparar lâmpada nova
3. Retirar lâmpada fundida
4. Colocar lâmpada nova
5. **Ligar interruptor para verificar se a nova lâmpada acende**
6. **Se a lâmpada acender, terminar o processo, caso contrário voltar ao passo 1**

# Algoritmos no mundo real

- No mundo real encontramos vários tipos de algoritmos
- Exemplo 1: Receita culinária de esparguete à bolonhesa
  1. Numa panela com água a ferver, coloque o óleo e tempere com sal. Coloque o esparguete a cozer conforme as instruções da embalagem e mexa bem para não colar.
  2. Num tacho, leve ao lume a margarina, a cebola, o alho e as folhas de louro. Deixe refogar durante 5 minutos.
  3. Passado 5 minutos, junte a carne e mexa para fritar bem. Tempere com sal, pimenta e noz-moscada. Quando a carne ficar solta e com cor junte a polpa de tomate e mexa. Junte o cubo de caldo de carne e o vinho branco. Deixe cozinhar entre 10 a 15 minutos até o molho ficar bem apurado.
  4. Depois do esparguete cozido, escorra-o com no escurador. Depois do molho apurado, tempere com orégãos, mexa e apague o lume. Decore o prato com orégãos e sirva o queijo à parte.

<http://www.saborintenso.com/f16/esparguete-bolonhesa-869/>

# Algoritmos no mundo real

- Exemplo 2: Instruções de segurança

Compartimentos onde é utilizado o gás → Se ocorrer uma fuga de gás:

1. Feche o gás na válvula de corte do local
2. Informe o encarregado de segurança e peça-lhe que feche o fluxo de gás nas válvulas de corte gerais
3. Não faça fogo ou chamas
4. Não acione dispositivos eléctricos
5. Abra as portas e janelas
6. Promova a evacuação do local e certifique-se de que alguém não autorizado não entra no local

[https://www.uc.pt/e-prevencao/trabalhadores/seguranca/segurancadeedificios/instrucoes/InstSegParticulares\\_UtilizacaoGas](https://www.uc.pt/e-prevencao/trabalhadores/seguranca/segurancadeedificios/instrucoes/InstSegParticulares_UtilizacaoGas)

# Algoritmos no mundo real

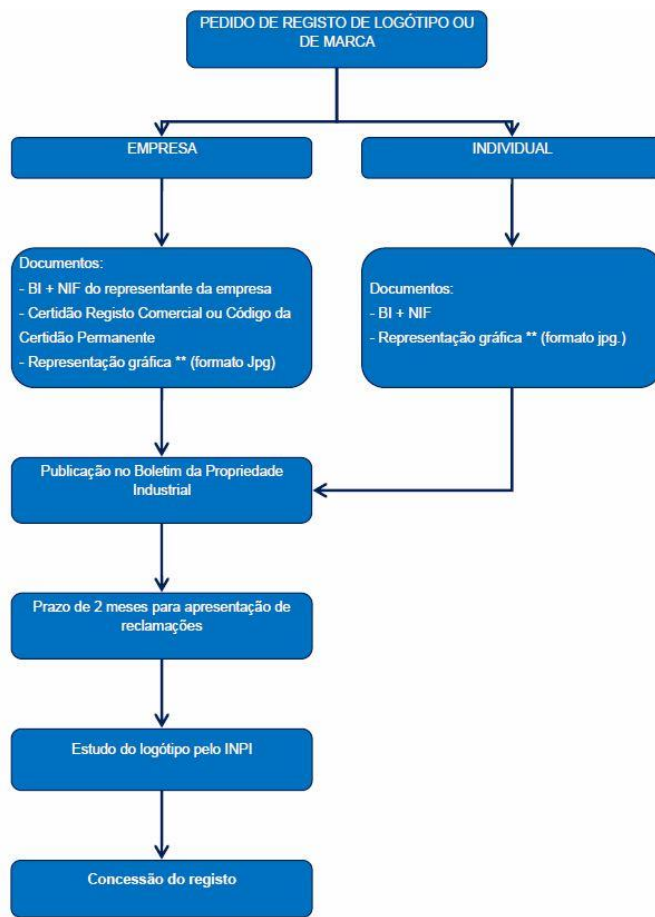
## Exemplo 3: Instruções para matrícula



[http://engeletrica.sites.ufms.br/files/2016/10/Instru%C3%A7%C3%B5es-para-MATRICULA-2016-2\\_FAENG\\_v02-1-1.jpg](http://engeletrica.sites.ufms.br/files/2016/10/Instru%C3%A7%C3%B5es-para-MATRICULA-2016-2_FAENG_v02-1-1.jpg)

# Algoritmos no mundo real

- Exemplo 4: Pedido de registo de marca/logotipo



[http://www.ideram.pt/Content/Images/Passos\\_Processo\\_Pedido\\_Registo.JPG](http://www.ideram.pt/Content/Images/Passos_Processo_Pedido_Registo.JPG)

# Algoritmos no mundo real

## Exercício 1

- Efetue uma pesquisa na Internet e obtenha **dois** algoritmos de tipos diferentes
- Exemplo: 1 fluxograma + 1 algoritmo em linguagem natural

# Tipos de algoritmos

- Existem vários tipos de algoritmos
- Podem ser muito simples ou bastante complexos
- Existem 3 tipos básicos:
  - algoritmos simples (ou *straightforward*)
  - com recurso a decisão
  - com recurso a repetição

# Tipos de algoritmos: Algoritmo simples

- Consiste num conjunto de etapas (ou passos)
- As etapas são executadas sequencialmente
- Exemplo de um algoritmo simples:

## Trocar uma lâmpada fundida

1. Preparar lâmpada nova
2. Retirar lâmpada fundida
3. Colocar lâmpada nova



# Tipos de algoritmos: Algoritmo simples

## Exercício 2

- Escreva um algoritmo simples
- Exemplo: como andar de elevador

# Tipos de algoritmos: Com recurso a decisão

- Também consiste num conjunto de etapas
- Mas algumas etapas apenas são executadas **se** uma determinada condição for **verdadeira**
- Podem ser especificadas etapas a executar **se** a condição for **falsa**

# Tipos de algoritmos: Com recurso a decisão

- Exemplo de um algoritmo com recurso a decisão:

## Procedimentos a efetuar caso a máquina de lavar não inicie o ciclo de lavagem:

1. Pressionou o botão **START**?
  - Se **sim**, passe para o ponto 2
  - Se **não**, pressione o botão **START**
2. A porta do compartimento da roupa está fechada?
  - Se **sim**, passe para o ponto 3
  - Se **não**, feche a porta do compartimento
3. O bloqueio infantil está ativado?
  - Se **sim**, desative o bloqueio infantil
  - Se **não**, passe para o ponto 4
4. Contacte o serviço de assistência técnica

Decisão

# Tipos de algoritmos: Com recurso a decisão

- Exemplo de um algoritmo com recurso a decisão:

## Procedimentos a efetuar caso a máquina de lavar não inicie o ciclo de lavagem:

- Pressionou o botão **START**?
  - Se **sim**, passe para o ponto 2
  - Se **não**, pressione o botão **START**
- A porta do compartimento da roupa está fechada?
  - Se **sim**, passe para o ponto 3
  - Se **não**, feche a porta do compartimento
- O bloqueio infantil está ativado?
  - Se **sim**, desative o bloqueio infantil
  - Se **não**, passe para o ponto 4
- Contacte o serviço de assistência técnica

Executado se a condição  
for **verdadeira**

# Tipos de algoritmos: Com recurso a decisão

- Exemplo de um algoritmo com recurso a decisão:

## Procedimentos a efetuar caso a máquina de lavar não inicie o ciclo de lavagem:

- Pressionou o botão **START**?
  - Se **sim**, passe para o ponto 2
  - Se **não**, pressione o botão **START**
- A porta do compartimento da roupa está fechada?
  - Se **sim**, passe para o ponto 3
  - Se **não**, feche a porta do compartimento
- O bloqueio infantil está ativado?
  - Se **sim**, desative o bloqueio infantil
  - Se **não**, passe para o ponto 4
- Contacte o serviço de assistência técnica

Executado se a condição  
for **falsa**

# Tipos de algoritmos: Com recurso a decisão

## Exercício 3

- Elabore um algoritmo para efetuar login num site
- Considere que o utilizador deverá introduzir a seguinte informação:
  - nome de utilizador
  - password

# Tipos de algoritmos: Com recurso a repetição

- Contém uma ou várias etapas que se **repetem** enquanto uma condição for verdadeira
- Exemplo 1:

## Como colocar combustível no automóvel

1. Indicar montante (em euro) a colocar
2. Retirar mangueira de combustível
3. Enquanto não for atingido o montante definido em 1:
  - apertar o gatilho da agulheta e abastecer o automóvel com combustível
4. Guardar mangueira de combustível

# Tipos de algoritmos: Com recurso a repetição

- Exemplo 2:

## DVD: Instruções para visionar filme

1. Clicar no botão ► (Play) para iniciar o filme
2. Enquanto o utilizador não clicar no botão ■ (Stop) OU o filme não terminar:
  - o filme é visualizado na televisão
3. É apresentado o menu inicial do DVD



# Tipos de algoritmos: Com recurso a repetição

## Exercício 4

- Elabore um algoritmo ou descreva um exemplo da vida real, com recurso a repetição

# PSEUDOCÓDIGO

# Pseudocódigo: Introdução

- Pseudocódigo é uma forma de representar algoritmos, que:

**mistura palavras da linguagem natural...**

por exemplo:

**Ler, Escrever, etc.**

**...com símbolos e notações típicas de uma linguagem de programação**

por exemplo:

**$A \leftarrow B$**

- A utilização de pseudocódigo é um auxiliar útil na escrita de algoritmos
- Facilita o posterior desenvolvimento do programa numa linguagem de programação

# Pseudocódigo: Introdução

- Existe com o objetivo de ser lido e entendido por seres humanos
- Por oposição a um algoritmo escrito numa linguagem de programação, que é lido por um computador
- Neste módulo iremos escrever pseudocódigo utilizando a linguagem **Python**



# Linguagem Python

## Características principais

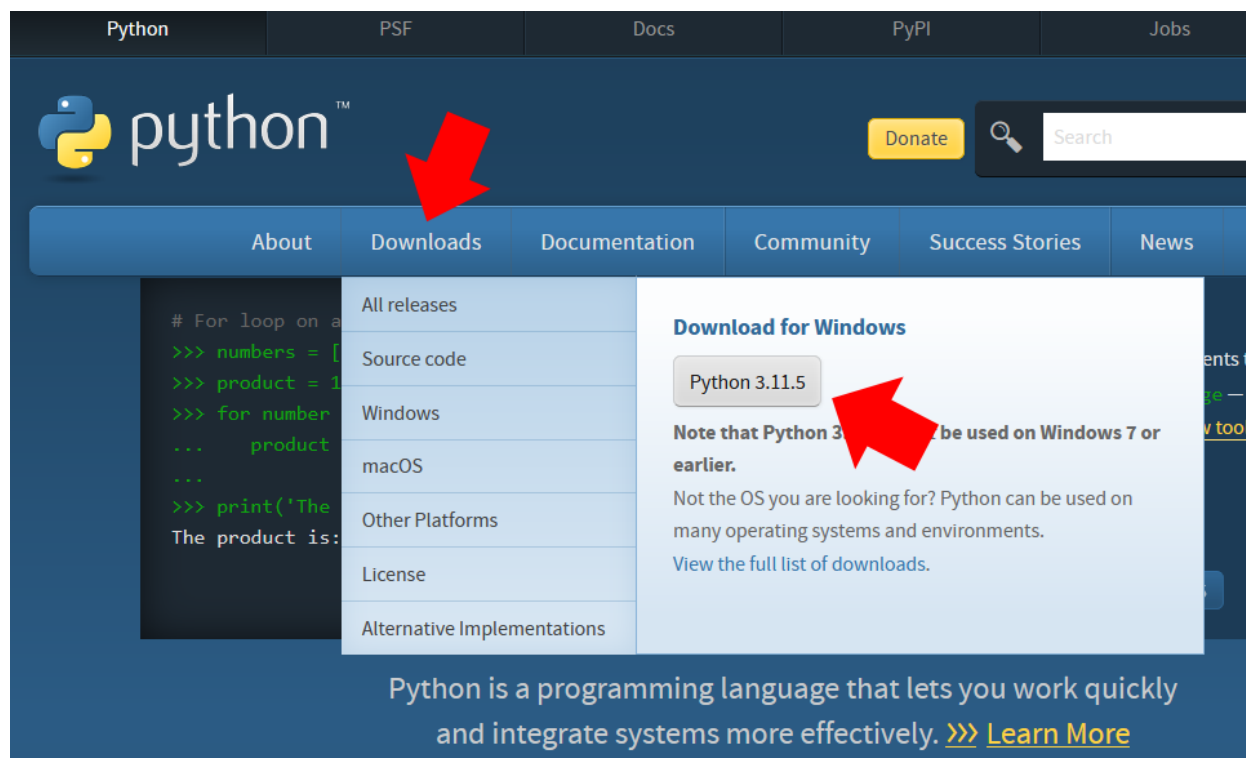
- É uma linguagem de alto nível
- É uma linguagem interpretada
- Os ficheiros terminam em .py

# Linguagem Python



## Instalação

- Ir a <https://www.python.org> e efetuar download

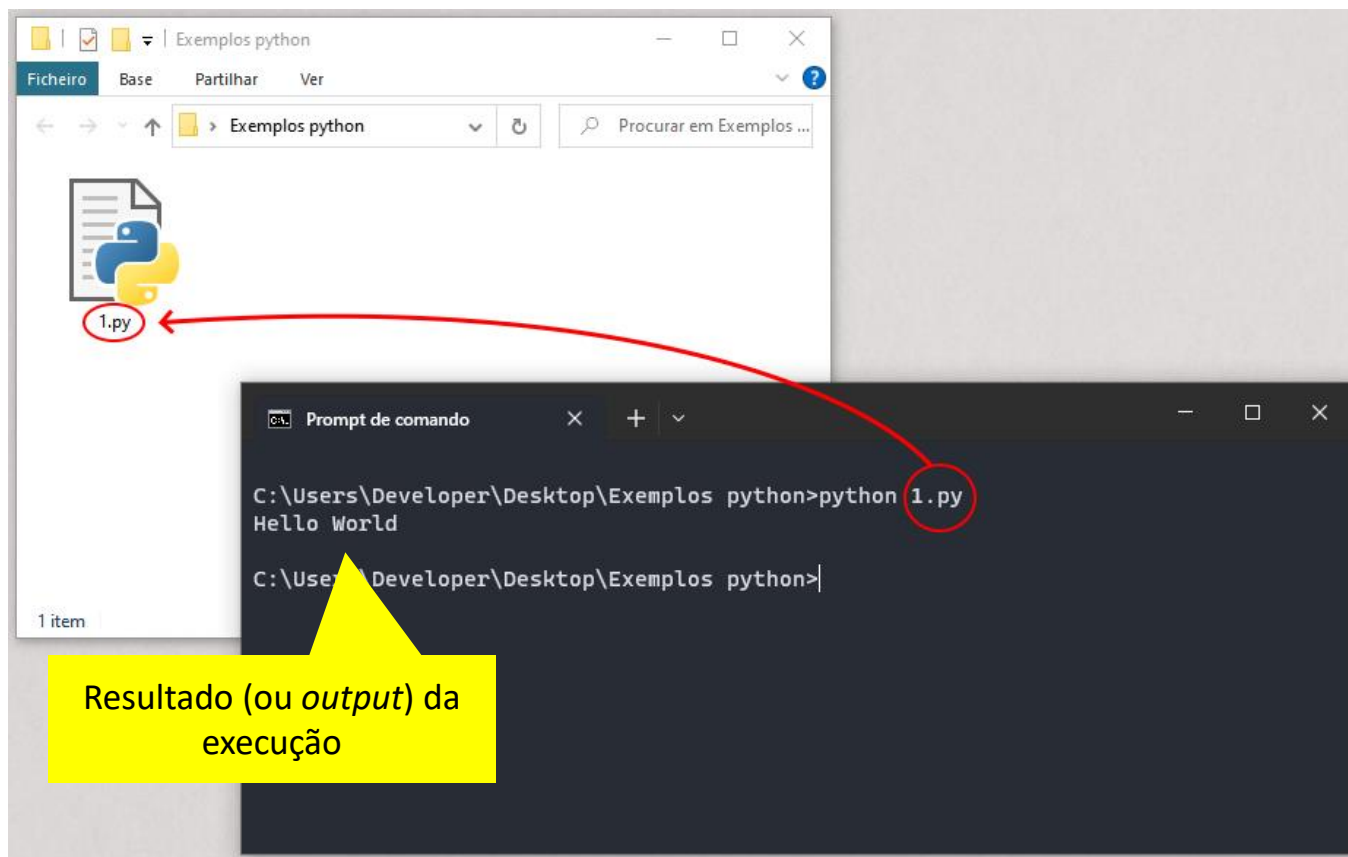




# Linguagem Python

## Executar código Python

- Na linha de comando do Windows (Terminal do Windows):

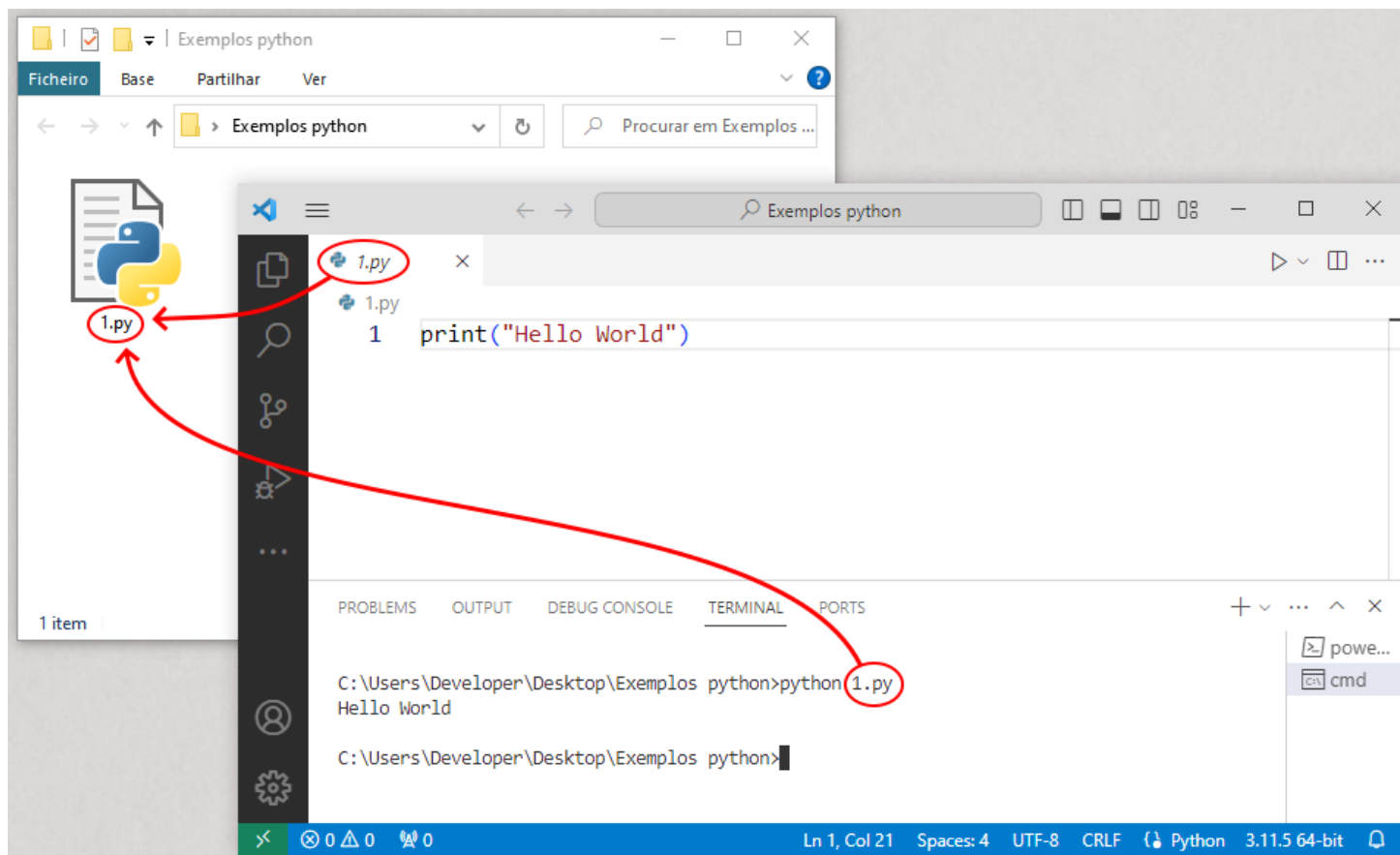


# Linguagem Python



## Executar código Python

- Visual Studio Code:







# Linguagem Python

- Verificar que está instalado e qual a versão

```
python --version
```

- Executar scripts na linha de comando

```
python ficheiro.py
```

(em que **ficheiro** é o nome do ficheiro a executar)



# Comentários

- Os comentários começam com o caracter #
- Podem ser utilizados com dois objetivos:
  - Documentar o código, ou seja, explicar determinadas secções de código:

```
# Calcular o quadrado de 4  
quadrado = 4 * 4
```

- Impedir que o código seja executado

```
# quadrado = 4 * 4  
quadrado = 2 * 2
```

Esta linha de código não é executada

Esta linha de código é executada

- Os comentários podem ser colocados no final de uma linha

```
quadrado = 4 * 4 # Calcular o quadrado de 4
```



# Saída de dados

- A função `print()` permite efetuar a saída de dados
- Ou seja, enviar os dados para fora do algoritmo
- Os dados poderão ir para um qualquer dispositivo (por exemplo: ecrã, impressora, etc.)
- Sintaxe:  

```
print("mensagem")  
print(variável)
```
- Os dados a mostrar podem ser uma mensagem (**string**), o conteúdo de uma variável ou o resultado de uma expressão



# Saída de dados

- Sintaxe:

```
print("mensagem")  
print(variável)
```

- Exemplo: mostrar uma mensagem

```
print("Hello World")
```

- Exemplo: mostrar uma expressão (resultado de um cálculo)

```
print(2 + 2)
```



# Saída de dados

- O caracter `\n` é conhecido como caracter newline e é utilizado para enviar o output para a linha seguinte

- Exemplo:

```
print("Hello World\nLinguagem Python")
```

- Resultado:

```
Hello World  
Linguagem Python
```

- Exemplo:

```
print("Hello World\n\nLinguagem Python")
```

- Resultado:

```
Hello World  
  
Linguagem Python
```

# Variáveis

- As variáveis são utilizadas para guardar valores
- A qualquer momento pode ser atribuído um determinado valor a uma variável
- Qual a utilidade das variáveis?
  - tal como as pessoas utilizam carteiras para transportar objetos de que necessitam no seu dia-a-dia (cartões, dinheiro, etc.)...
  - ...as variáveis permitem armazenar os valores (ou dados) de que os algoritmos necessitam para funcionarem
- Por exemplo, um algoritmo que calcula a soma de dois números, necessita de:
  - uma variável para guardar o 1º número
  - uma variável para guardar o 2º número
  - uma variável para guardar o resultado da soma



# Variáveis

- Para atribuir um valor a uma variável é utilizada a seguinte instrução:

`variável = expressão`

- Exemplo: atribuir à variável `x` o valor `5`

```
x = 5
```

- Exemplo 2: atribuir à variável `cidade` o valor `Coimbra`

```
cidade = "Coimbra"
```

- Exemplo 3: atribuir à variável `preco` o valor `19,99`

```
preco = 19.99
```

**Importante:** repare que é utilizado o caracter **ponto .** e não **vírgula**



# Variáveis

- Exemplo: atribuir o valor lógico **True** (verdadeiro) à variável **casado**

```
casado = True
```

- Exemplo: atribuir a expressão **2 + 2** à variável **y**

```
y = 2 + 2
```

(após esta operação a variável **y** fica com o valor **4**)

- Exemplo: calcular um preço com IVA (23%)

```
precoFinal = preco * 1.23
```

**Importante:** repare que é utilizado o caracter  
\* para a operação de multiplicação





# Variáveis

- Também é possível atribuir a uma variável o valor de outra variável
- Exemplo:

```
x = 5  
y = x
```

- **Questão:** qual o valor de *y*?

- Exemplo:

```
x = 5  
y = 2  
z = x + 5 - y
```

- **Questão:** qual o valor de *z*?



# Variáveis: Regras para os nomes

- Em Python os nomes das variáveis são *case-sensitive*
- Ou seja, há diferenciação quanto a maiúsculas e minúsculas
- Por exemplo, `y` e `Y` são duas variáveis diferentes:

```
y = 5  
Y = 10
```



# Variáveis: Regras para os nomes

- O nome de uma variável deve começar por uma letra ou pelo caracter *underscore* —
- O nome de uma variável não pode começar com um número
- Exemplo incorreto: **2fatura**
- O nome de uma variável apenas pode conter letras (A..Z, a..z), números (0..9) e o caracter *underscore* \_



## Incorreto

- 1aluno
- numero-aluno
- Id#registo
- %IVA



## Correto

- aluno1
- \_numeroAluno
- Numero\_Aluno
- SalaEdificio8
- FATURAS



# Variáveis: Regras para os nomes

## Recomendação

- Apesar de possível, evitar utilizar nos nomes de variáveis:
  - emojis
  - acentuação (~, ´, ` , ç, etc.)



### Em vez de...

- preço
- ÚltimoNome
- número-canção



### Utilizar...

- preco
- UltimoNome
- numero-cancao



# Variáveis

- Para mostrar o valor de uma variável, utilizar a função `print()`
- Exemplo:

```
temperatura = 21.6  
print(temperatura)
```

- Exemplo:

```
cidade = "Coimbra"  
print(cidade)
```



# Variáveis

- Para mostrar o valor de uma ou mais variáveis bem como strings, separar com vírgulas
- Exemplo:

```
cidade = "Coimbra"  
print("Cidade de", cidade)
```

- Resultado:

Cidade de Coimbra

- Exemplo:

```
preco = 19.99  
print("Preço do produto", preco, "€")
```

- Resultado:

Preço do produto 19.99 €



# Tipos de dados

- As variáveis podem armazenar dados/informações de diferentes tipos
- A linguagem Python implementa vários tipos
- Alguns dos tipos mais utilizados:

Tipo	Descrição	Exemplos
str	String: conjunto de caracteres alfanuméricos (letras, algarismos, símbolos)	x = "Hello world" s = ":-)"
int	Valores numéricos inteiros, positivos ou negativos	x = 5 y = -20
float	Números <i>floating point</i> . Podem ser expressos em notação científica com o caracter "e"	x = 20.5 y = 1.0 z = -87.7e100
bool	Valores lógicos, ou booleanos. Valores possíveis: True ou False	x = True y = False



# Operadores

- Durante a execução de um algoritmo, este efetua operações sobre os dados de modo a atingir um resultado
- As operações são feitas utilizando operadores
- Apesar de, em Python, poderem ser utilizados vários tipos de operadores, começamos por analisar os mais simples, que são os **operadores aritméticos básicos**:

Operador	Descrição	Exemplo
+	Soma	5 + 2
-	Subtração	5 - 2
*	Multiplicação	5 * 2
/	Divisão	5 / 2





# Operadores: Concatenação

- O operador `+` quando utilizado com strings permite efetuar a operação de concatenação
- Ou seja, juntar duas ou mais strings
- Exemplo:

```
a = "Olá,"  
b = "bom"  
c = "dia."  
d = a + b + c  
print(d)
```

- Resultado:

```
Olá,bomdia.
```



# Operadores: Concatenação

- Exemplo:

```
a = "Olá,"  
b = "bom"  
c = "dia."  
d = a + " " + b + " " + c  
print(d)
```

- Resultado:

Olá, bom dia.



# Entrada de dados

- A função `input()` permite obter (ou ler) dados
- Ou seja, receber dados para dentro do algoritmo
- Sintaxe:  
`input([mensagem])`
- O parâmetro `mensagem` é opcional
- Se o parâmetro existir, o conteúdo de `mensagem` é mostrado e depois são recebidos os dados
- Exemplo:

```
nome = input("Introduza o seu nome: ")  
print("O seu nome é", nome)
```

- Resultado:

```
Introduza o seu nome: João  
O seu nome é João
```

Introduzido pelo  
utilizador



# Entrada de dados

- Exemplo:

```
nome = input()
print("O seu nome é", nome)
```

- Resultado:

Introduzido pelo  
utilizador

```
João
O seu nome é João
```



# Entrada de dados

- Os dados são recebidos em formato string
- Por esse motivo, os dados numéricos necessitam de ser convertidos para `int` ou `float` utilizando as funções:
  - `int()`
  - `float()`
- Exemplo: ler dois números inteiros, efetuar a respetiva divisão e armazenar o resultado na variável `r`

```
a = int(input())  
b = int(input())  
r = a / b  
print("Resultado:", r)
```

- Resultado: 

Introduzido pelo utilizador

4  
2

Introduzido pelo utilizador

Resultado: 2.0



# Entrada de dados

- Exemplo:
  - ler um valor numérico e converter para `float`
  - somar o valor `1.2`
  - mostrar o resultado

```
temperatura = float(input("Indique a temperatura: "))  
resultado = temperatura + 1.2  
print(resultado)
```

- Resultado:

Indique a temperatura: 37.8  
39.0

Introduzido pelo  
utilizador

# Operadores relacionais

- Os operadores relacionais permitem-nos efetuar comparações
- O resultado é um valor lógico: **True** (verdadeiro) ou **False** (falso)

Operador	Descrição	Exemplo
==	Igualdade	1 == 3 (resultado: <b>False</b> ) 1 == 1 (resultado: <b>True</b> )
!=	Diferente	1 != 3 (resultado: <b>True</b> ) 1 != 1 (resultado: <b>False</b> )
<	Menor que	1 < 3 (resultado: <b>True</b> ) 1 < 1 (resultado: <b>False</b> )
>	Maior que	1 > 3 (resultado: <b>False</b> ) 1 > 1 (resultado: <b>False</b> )
<=	Menor ou a igual a	1 <= 3 (resultado: <b>True</b> ) 1 <= 1 (resultado: <b>True</b> )
>=	Maior ou igual a	1 >= 3 (resultado: <b>False</b> ) 1 >= 1 (resultado: <b>True</b> )

# Instruções condicionais

- As instruções condicionais permitem controlar ou condicionar a execução das instruções de um algoritmo
- Ou seja, torna-se possível executar certas instruções apenas se se verificar uma certa condição
- Sintaxe (simplificada):  
`if condição:`  
`instruções`
- O parâmetro `mensagem` é opcional



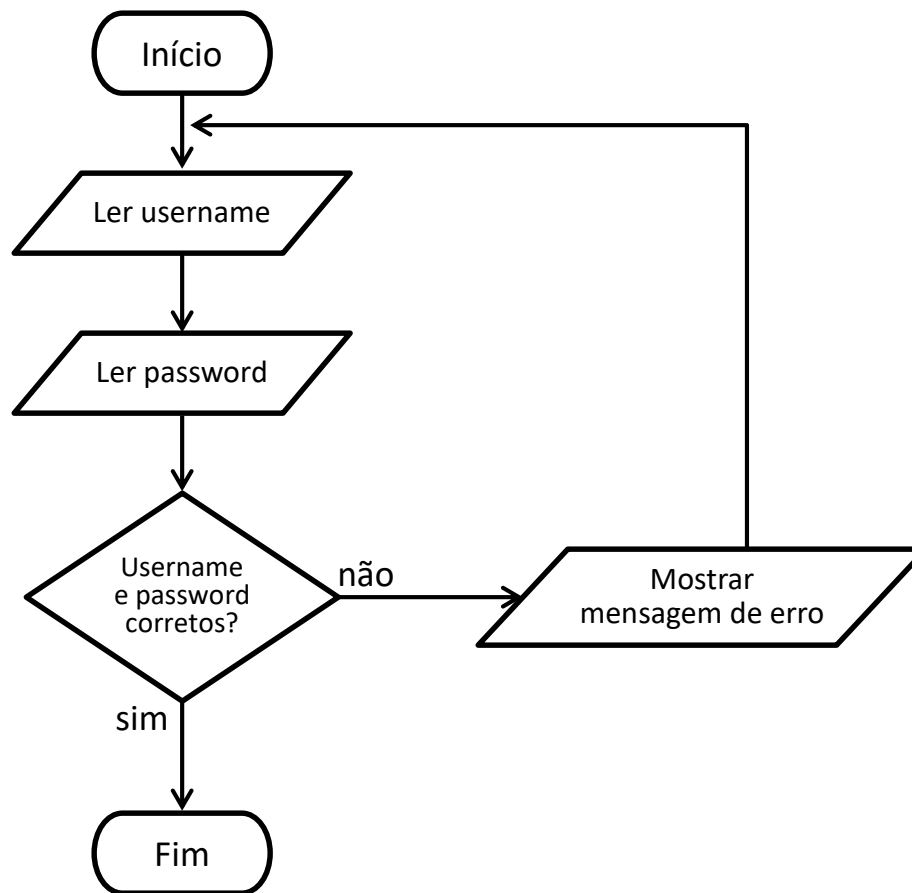
# FLUXOGRAMAS

# Fluxogramas

- Fluxogramas (em Inglês: *flowchart*) são diagramas utilizados para transmitir de forma visual e intuitiva:
  - processos
  - operações
  - algoritmos
  - informações
- Os fluxogramas são constituídos por símbolos
- Cada símbolo tem uma determinada finalidade

# Fluxogramas

- Exemplo: processo de autenticação num site

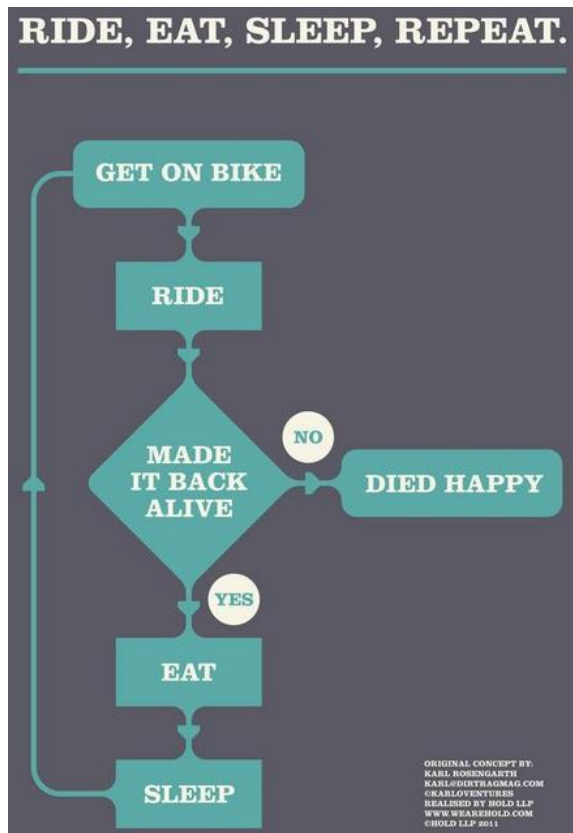


# Fluxogramas

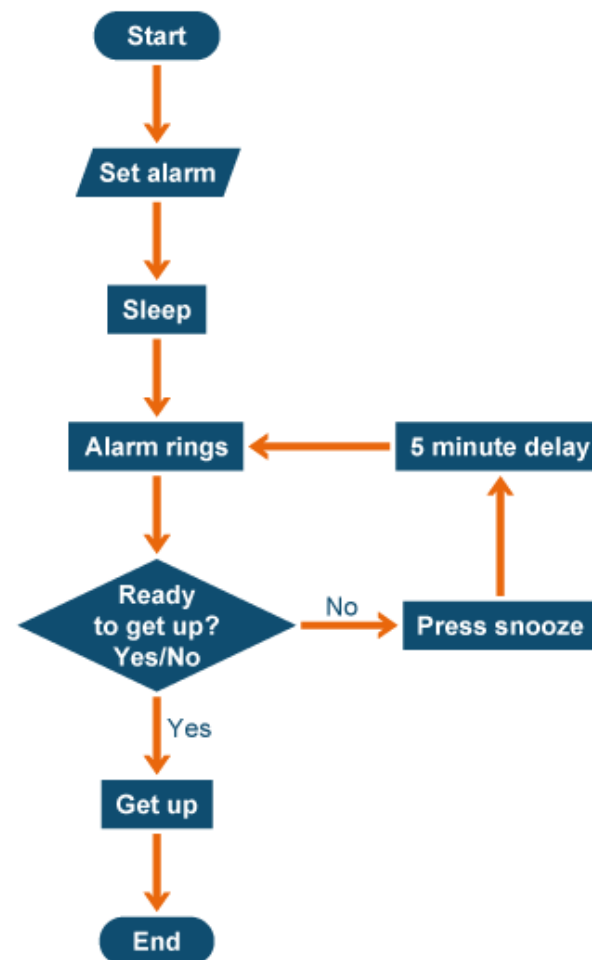
- Os fluxogramas não são apenas utilizados na informática
- Originalmente foram criados para serem utilizados em engenharia mecânica, mas rapidamente foram adotados noutros campos (gestão, por exemplo)
- Fora do contexto informático os fluxogramas são muito utilizados para transmitir vários tipos de informação
- São desenhados de forma mais ou menos livre, utilizando os símbolos, formas e cores que melhor se adaptam à situação

# Fluxogramas

- Exemplos:



[http://images.mentalfloss.com/blogs/wp-content/uploads/2011/06/550\\_ride\\_eat.jpg](http://images.mentalfloss.com/blogs/wp-content/uploads/2011/06/550_ride_eat.jpg)



<http://www.bbc.co.uk/staticarchive/aa37604a60bf83162cd8a15cfd675cb6cb099ca3.png>

# Fluxogramas



<https://s-media-cache-ak0.pinimg.com/736x/e4/8b/7d/e48b7dd4acc7b560717d599077092d69--flowchart-knock-knock.jpg>

# Fluxogramas: Símbolos

- Os seguintes são alguns dos símbolos utilizados para elaborar fluxogramas:



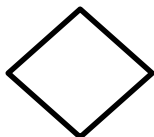
**Terminador:** início ou fim de um fluxograma



**Processo:** uma ação ou operação



**Entrada ou saída de dados:** ler ou mostrar dados



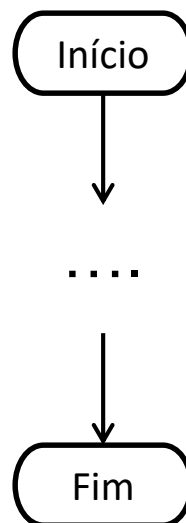
**Decisão:** avaliar uma condição

# Fluxogramas: Início e fim

- O início e o fim de um fluxograma são representados através do símbolo



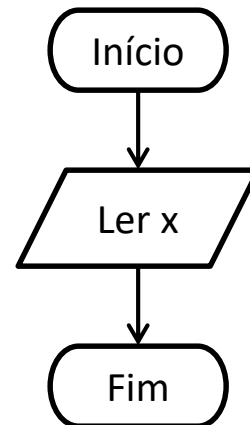
- Exemplo:





# Fluxogramas: Leitura de dados

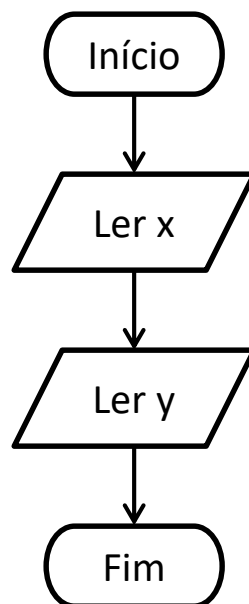
- A leitura de dados consiste em receber dados
- A leitura de dados é feita utilizando o símbolo de entrada ou saída de dados
- Exemplo: obter um número



- Neste caso é obtido um número e este fica armazenado na variável **x**

# Fluxogramas: Leitura de dados

- Exemplo: obter dois números



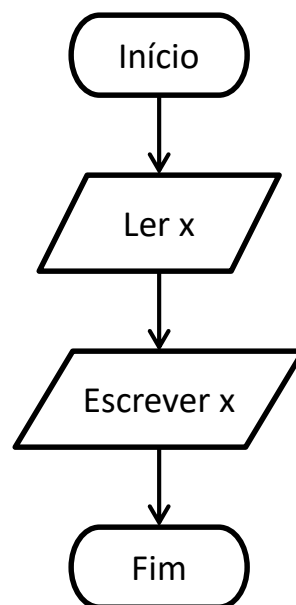
# Fluxogramas: Escrita de dados

- A escrita de dados consiste em enviar os dados para o "exterior"
- Pode ser entendido como:
  - mostrar os dados no ecrã de um computador
  - gravar os dados num suporte de informação (disco, pen USB, etc.)
- A escrita de dados é feita utilizando o símbolo de entrada ou saída de dados



# Fluxogramas: Escrita de dados

- Exemplo: obter um número, armazená-lo na variável **x** e de seguida mostrar **x**



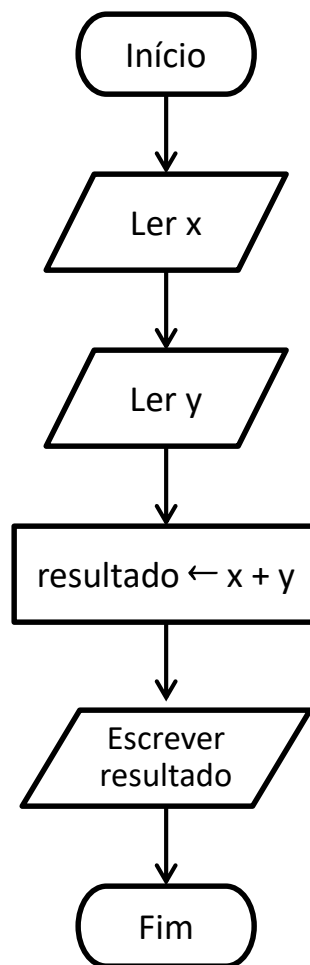
# Fluxogramas: Executar operações

- A execução de operações é feita através do símbolo de processo
- Uma operação é qualquer instrução que não seja:
  - entrada ou saída de dados
  - condicional
- Pode ser, por exemplo:
  - um cálculo
  - atribuir um valor a uma variável
  - etc.



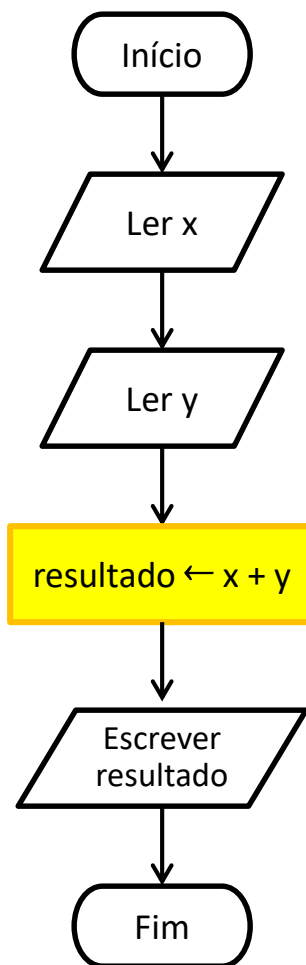
# Fluxogramas: Executar operações

- Exemplo: ler dois números, efetuar a soma e mostrar o resultado



# Fluxogramas: Executar operações

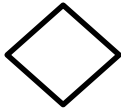
- Exemplo: ler dois números, efetuar a soma e mostrar o resultado

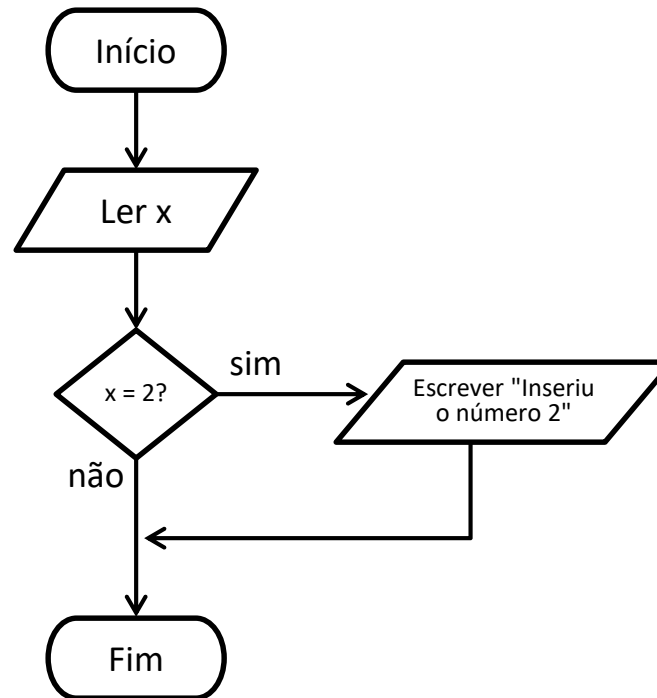


## Explicação

- somar  $x$  e  $y$
- armazenar o resultado na variável `resultado`

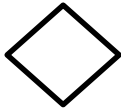
# Fluxogramas: Efetuar uma decisão

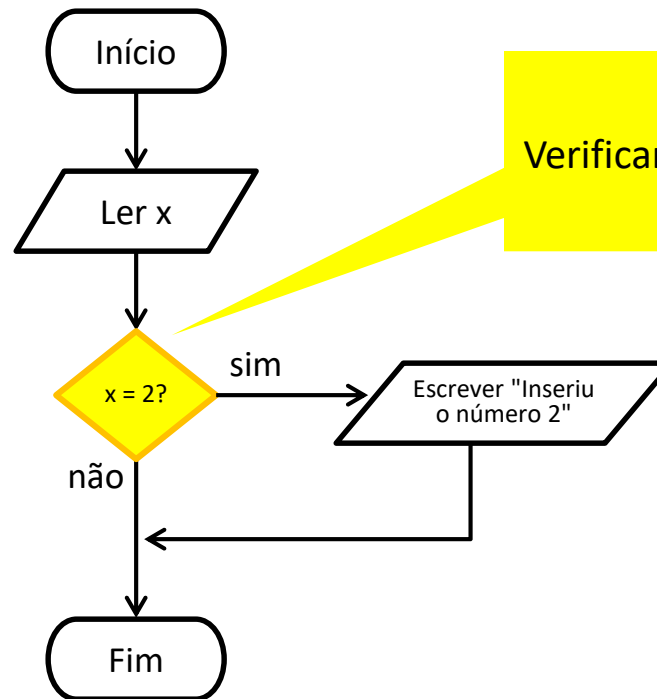
- As tomadas de decisão são efetuadas utilizando o símbolo 
- Equivale, em pseudocódigo, à instrução SE
- Exemplo: ler um número e verificar se é igual a 2





# Fluxogramas: Efetuar uma decisão

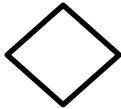
- As tomadas de decisão são efetuadas utilizando o símbolo 
- Equivale, em pseudocódigo, à instrução SE
- Exemplo: ler um número e verificar se é igual a 2

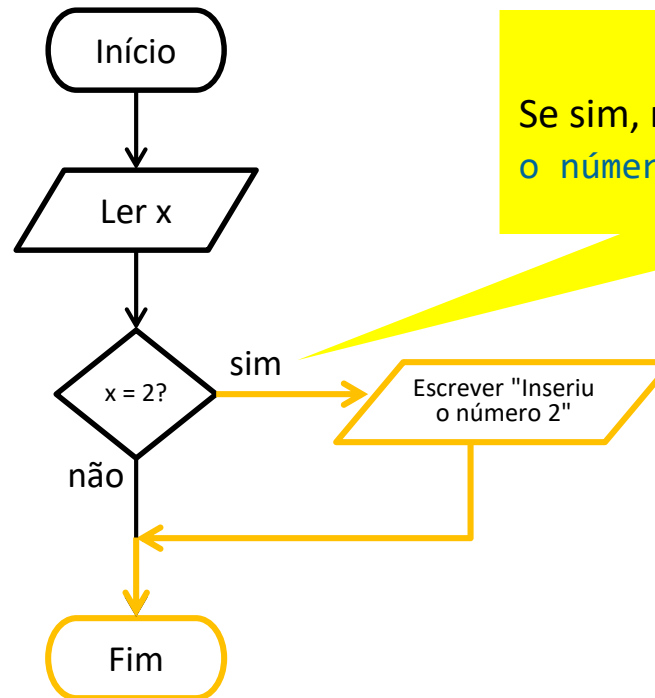


## Explicação

Verificar se  $x$  tem o valor 2

# Fluxogramas: Efetuar uma decisão

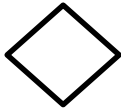
- As tomadas de decisão são efetuadas utilizando o símbolo 
- Equivale, em pseudocódigo, à instrução SE
- Exemplo: ler um número e verificar se é igual a 2

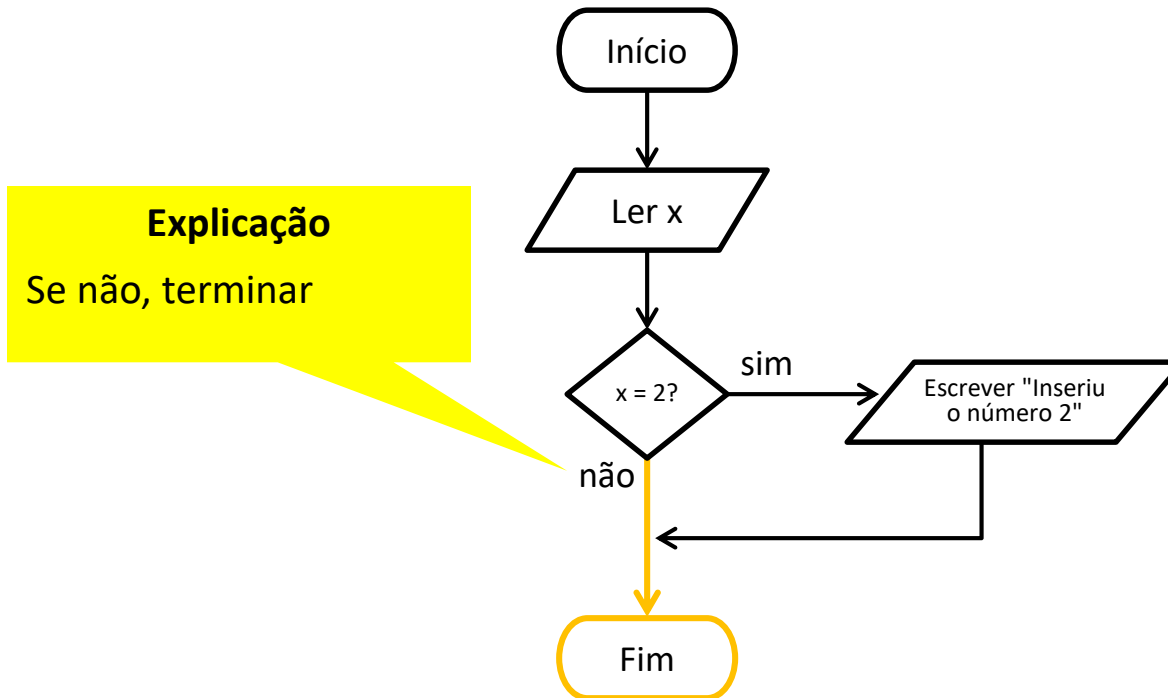


## Explicação

Se sim, mostrar a mensagem "Inseriu o número 2" e terminar

# Fluxogramas: Efetuar uma decisão

- As tomadas de decisão são efetuadas utilizando o símbolo 
- Equivale, em pseudocódigo, à instrução SE
- Exemplo 1: ler um número e verificar se é igual a 2



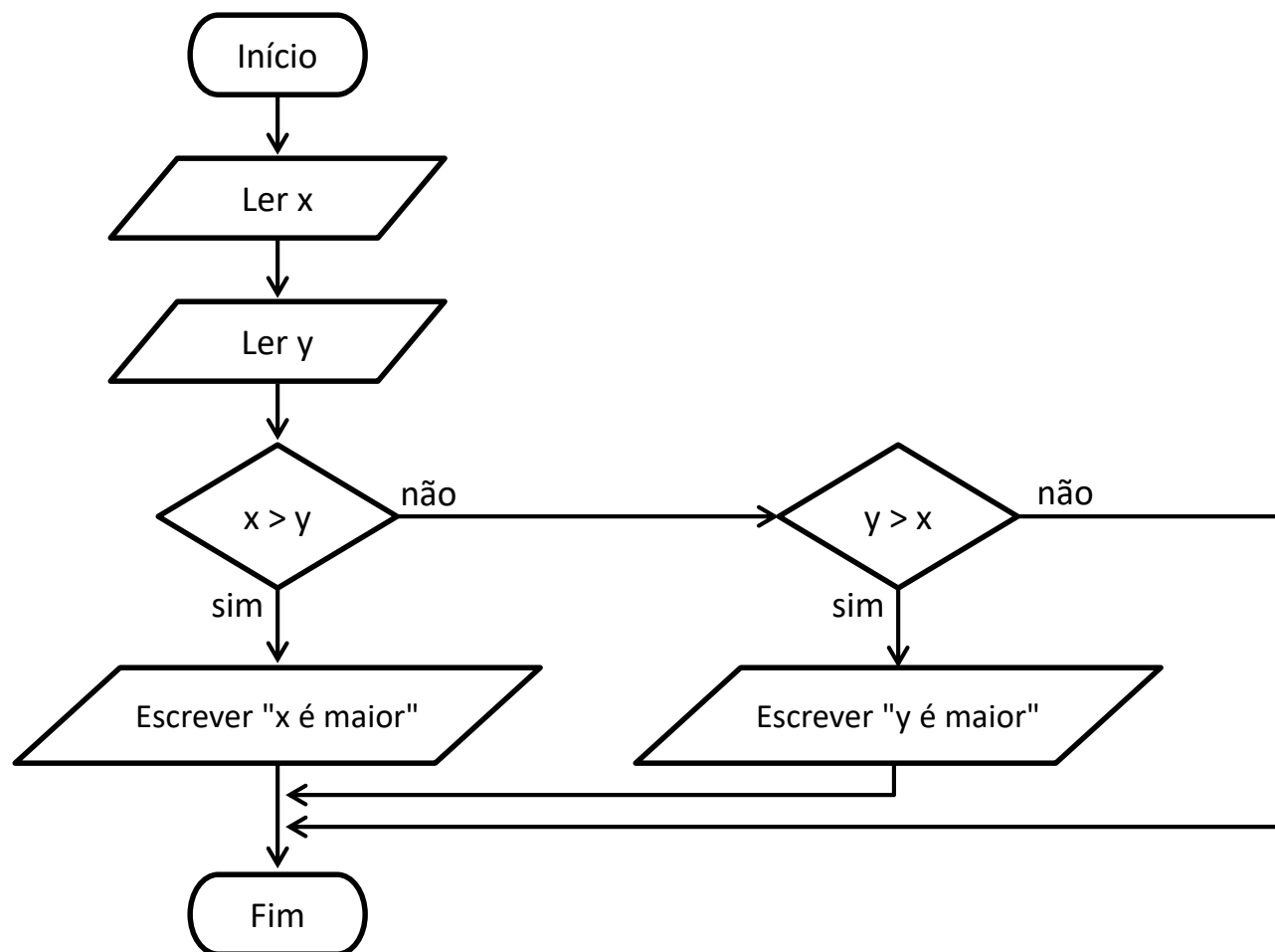
# Fluxogramas

## Exercício 8

- Desenhe um fluxograma que ilustre o processo de encomenda de uma pizza, de acordo com o seguinte algoritmo:
  - Em primeiro lugar o cliente deve especificar o serviço pretendido
    - Take away
      - Neste caso o passo seguinte consiste em especificar a loja onde é levantada a encomenda
    - Entrega ao domicílio
      - Neste caso o passo seguinte consiste em especificar a morada onde a encomenda é entregue
  - O cliente deve especificar a hora em que vai levantar a encomenda ou a hora em que a encomenda deve ser entregue ao domicílio
  - De seguida o cliente deve escolher:
    - a(s) pizza(s) pretendida(s)
    - a(s) bebida(s)
  - Para finalizar o cliente deverá aceitar o resumo que lhe é apresentado bem como o total a pagar

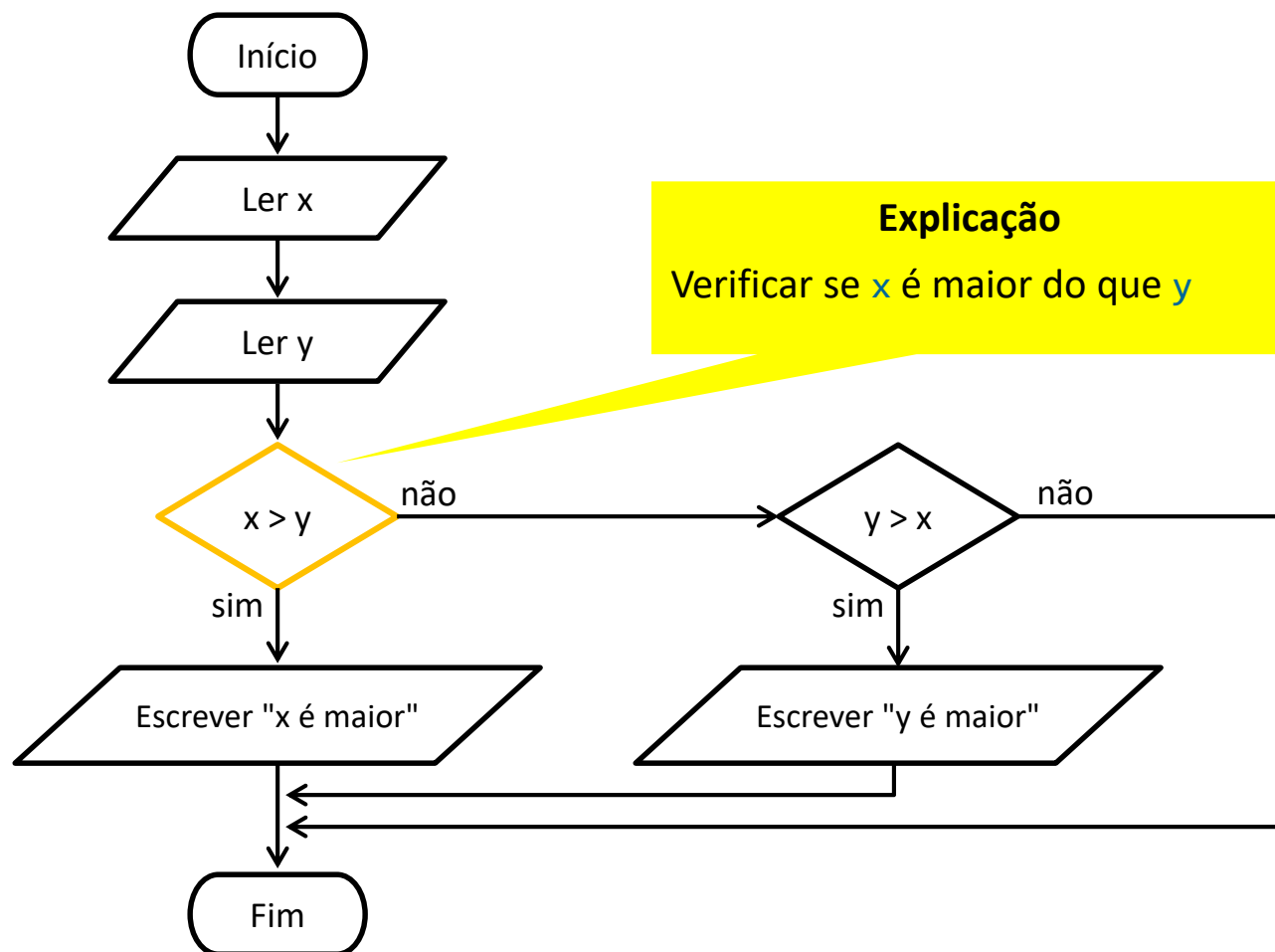
# Fluxogramas: Efetuar uma decisão

- Exemplo 2: obter 2 números e determinar qual é o maior



# Fluxogramas: Efetuar uma decisão

- Exemplo 2: obter 2 números e determinar qual é o maior

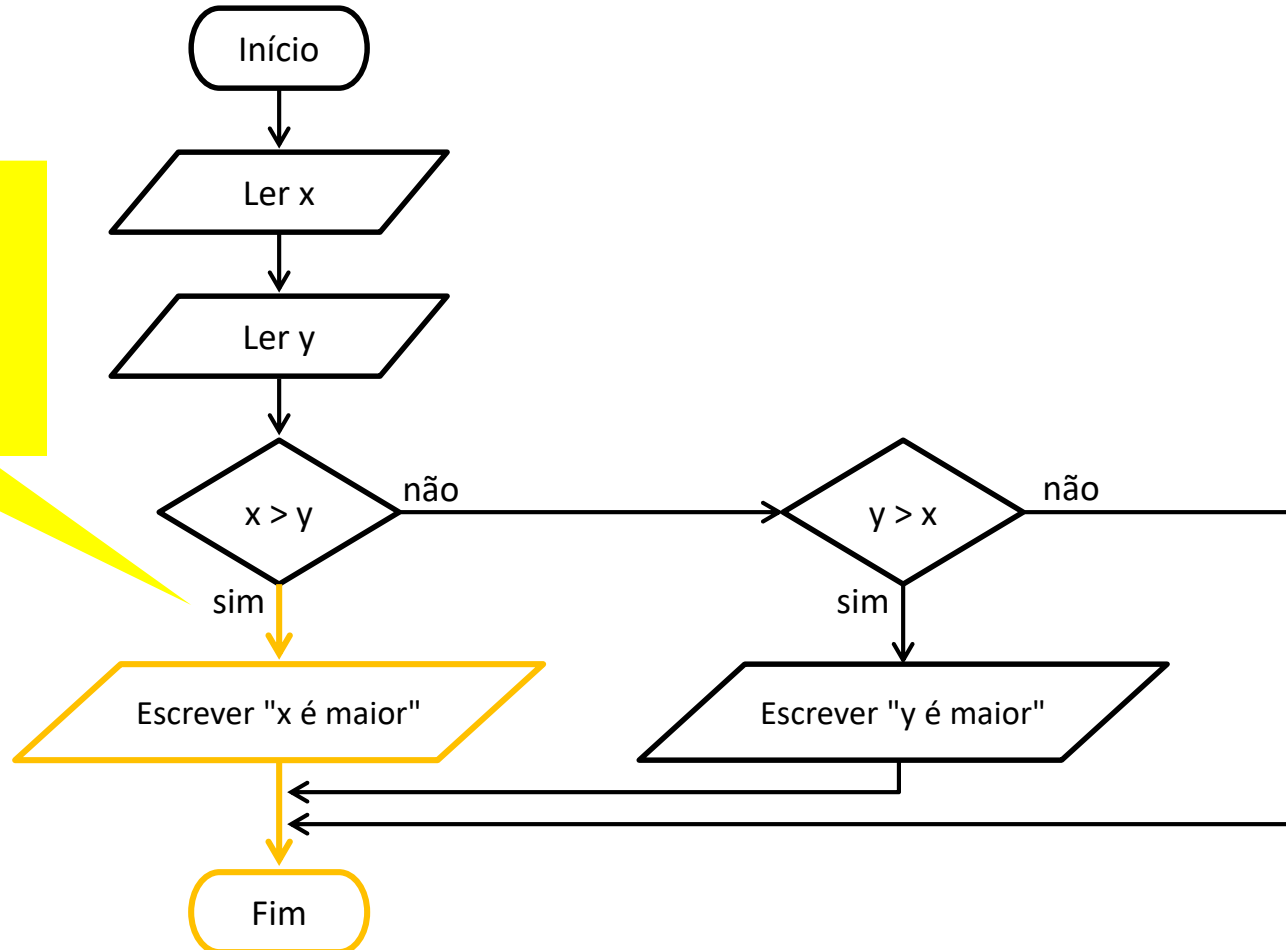


# Fluxogramas: Efetuar uma decisão

- Exemplo 2: obter 2 números e determinar qual é o maior

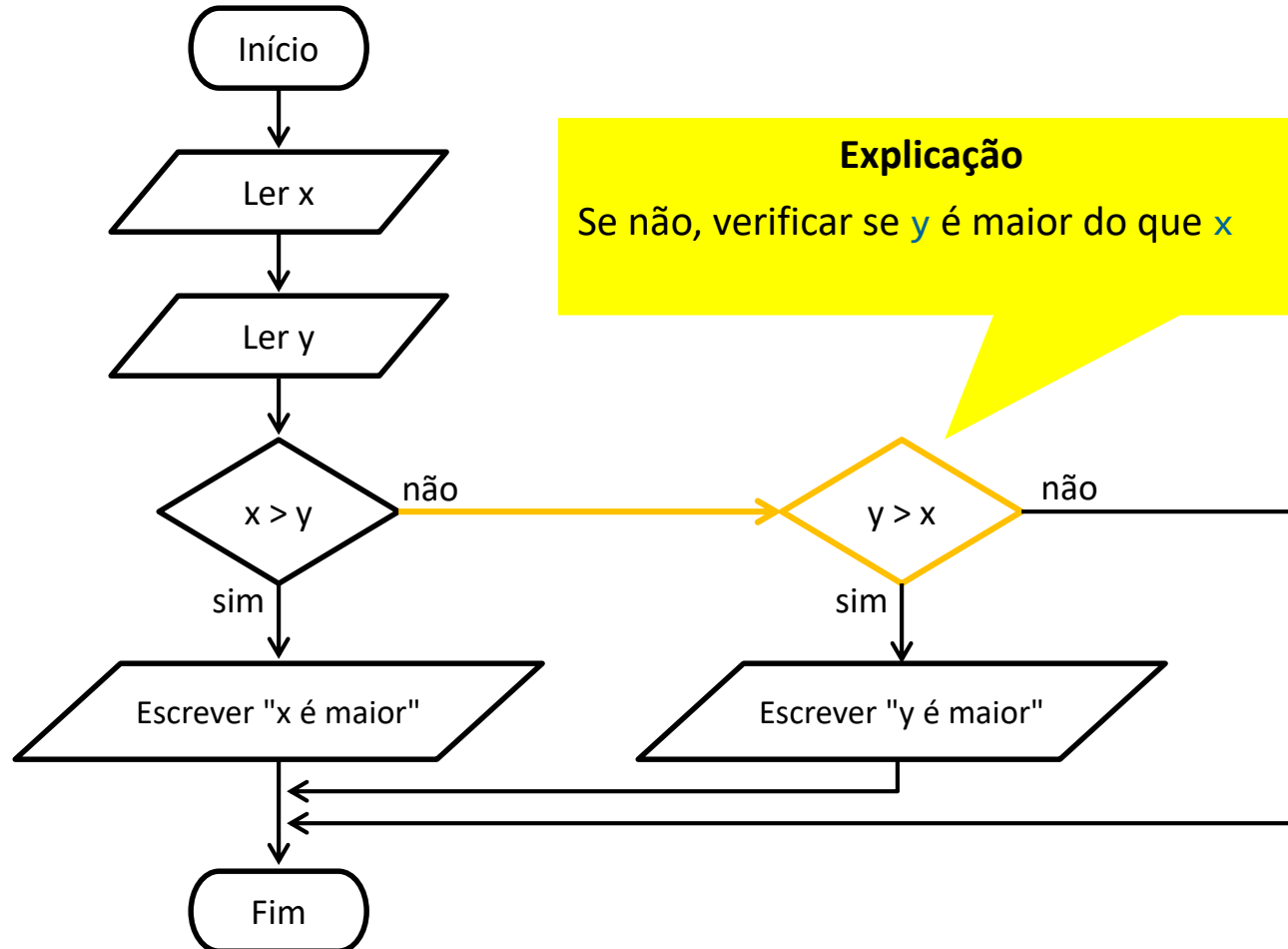
## Explicação

Se sim, mostrar a mensagem "x é maior" e terminar



# Fluxogramas: Efetuar uma decisão

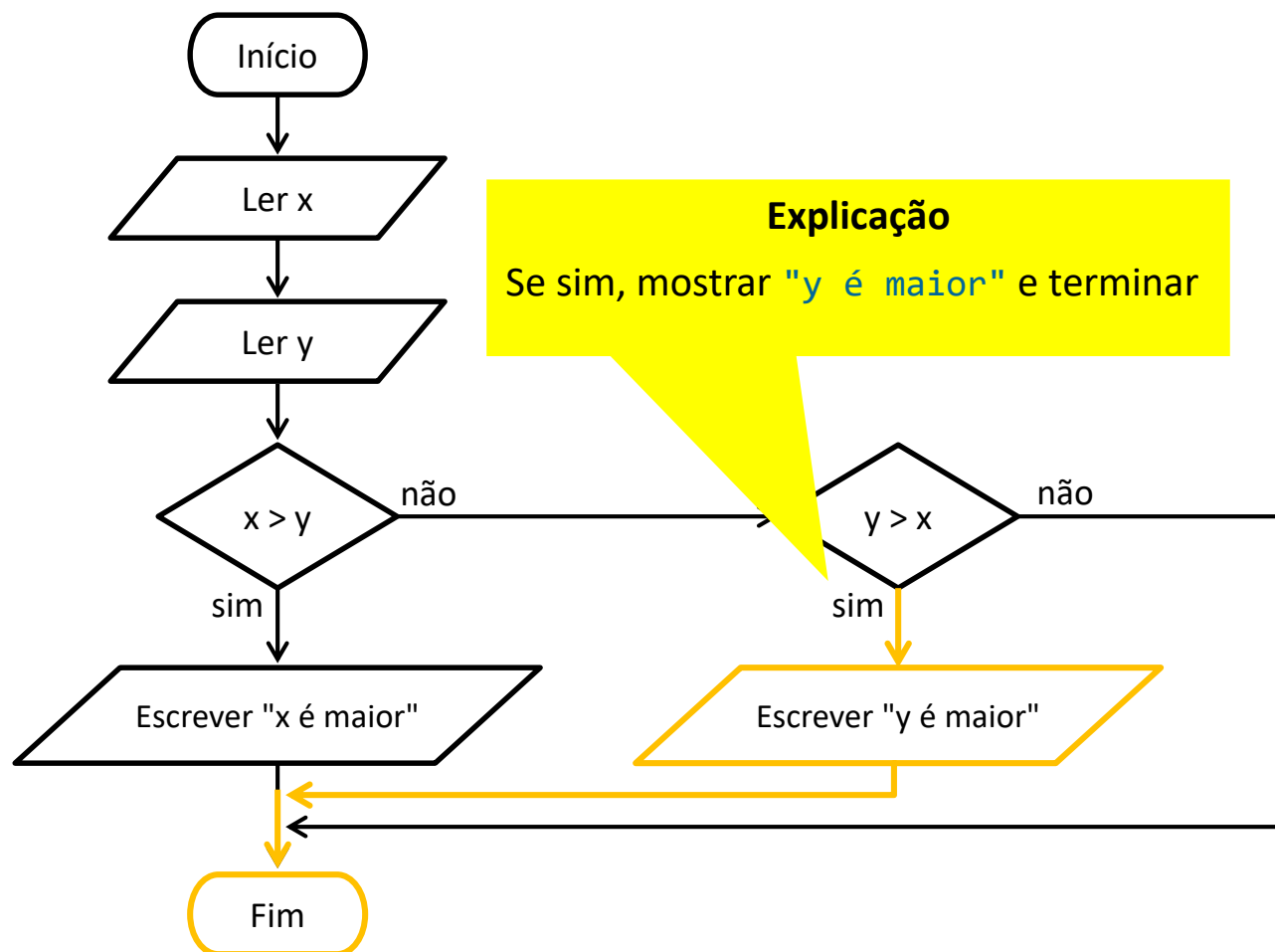
- Exemplo 2: obter 2 números e determinar qual é o maior





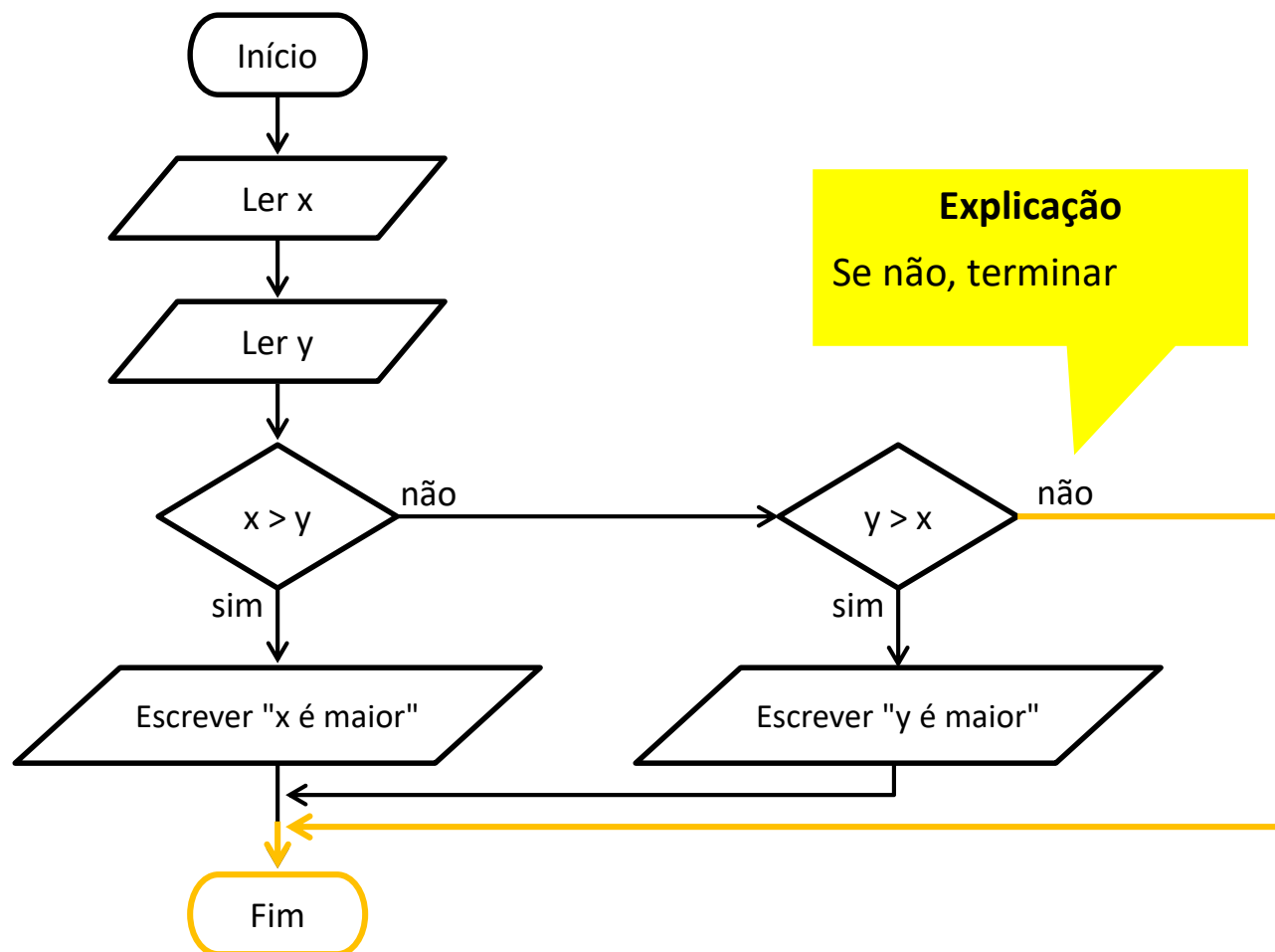
# Fluxogramas: Efetuar uma decisão

- Exemplo 2: obter 2 números e determinar qual é o maior



# Fluxogramas: Efetuar uma decisão

- Exemplo 2: obter 2 números e determinar qual é o maior

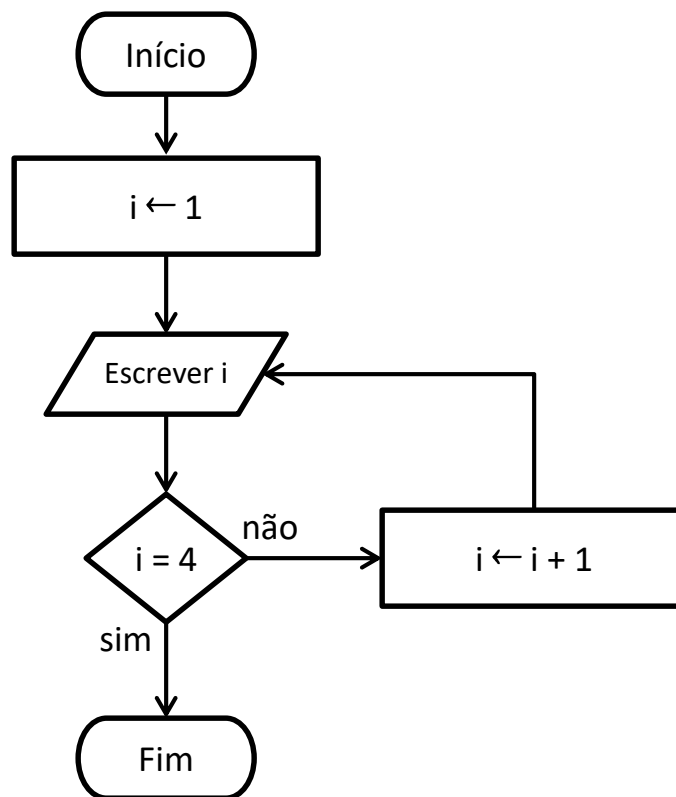


# Fluxogramas: Repetição

- É possível efetuar repetições, ou seja, repetir certos passos durante um certo número de vezes ou até que uma certa condição se verifique
- Para tal basta utilizar os símbolos já abordados

# Fluxogramas: Repetição

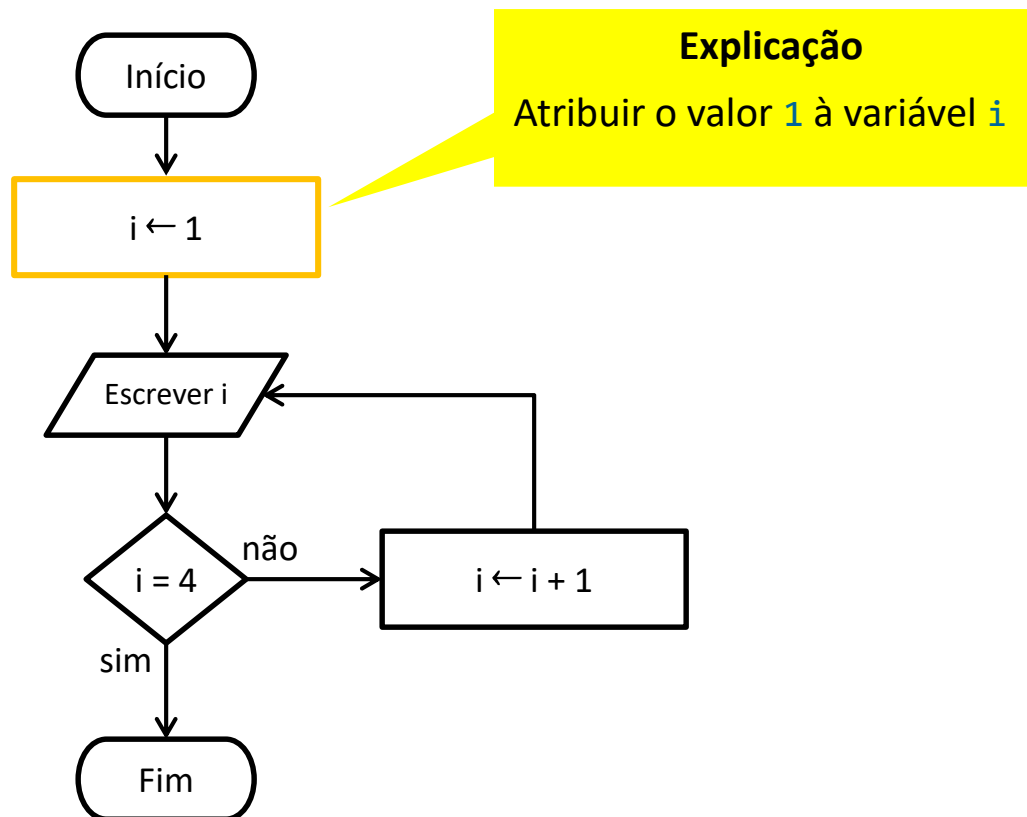
- Exemplo: mostrar os números de 1 a 4



# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

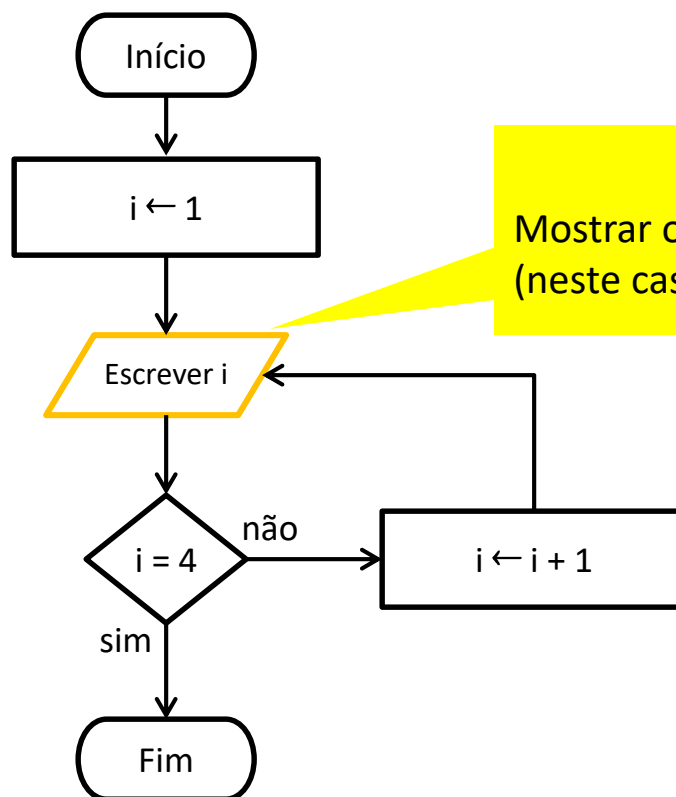
$i = 1$



# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 1$



## Explicação

Mostrar o valor da variável  $i$   
(neste caso, é 1)

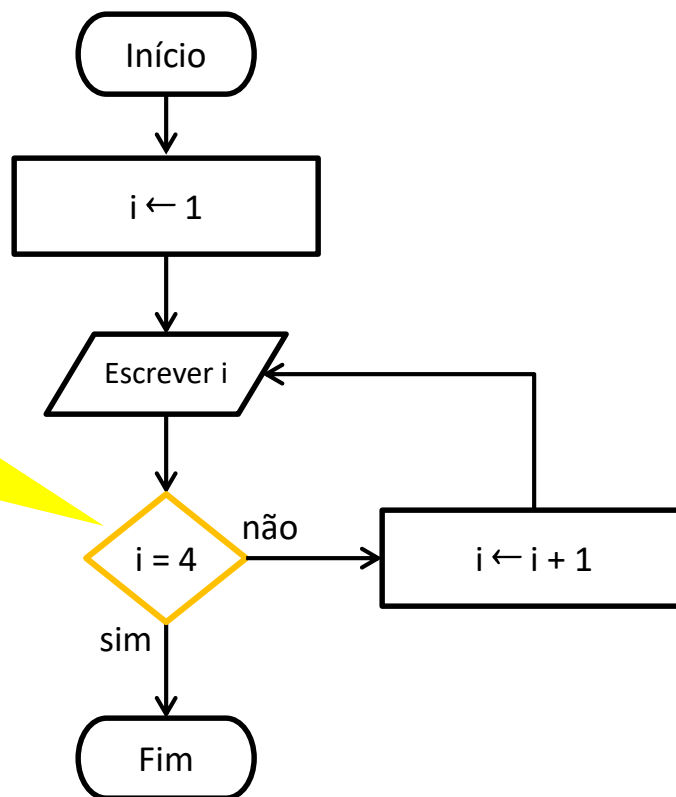
# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 1$

## Explicação

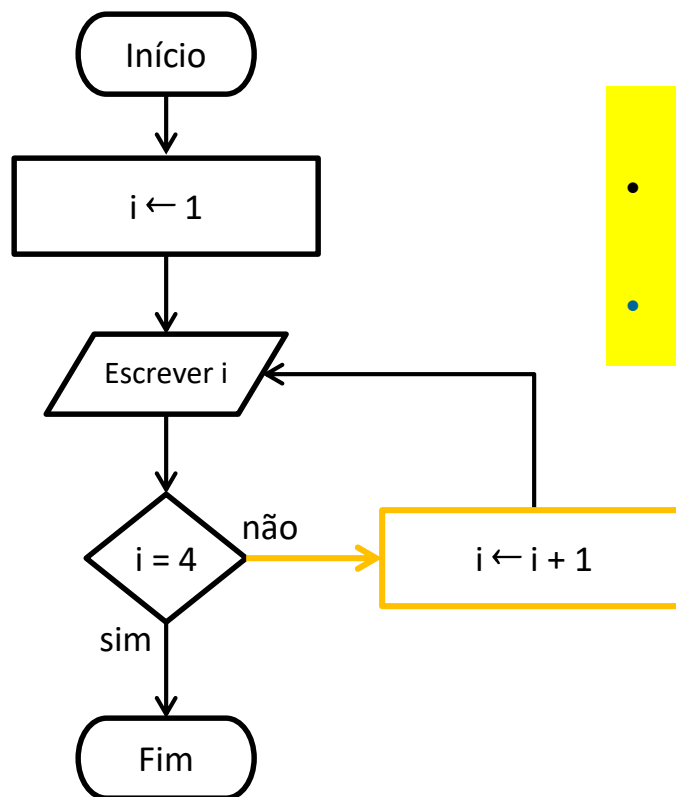
Verificar se  $i$  é igual a 4  
(neste caso não é, tem o valor 1)



# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 2$



## Explicação

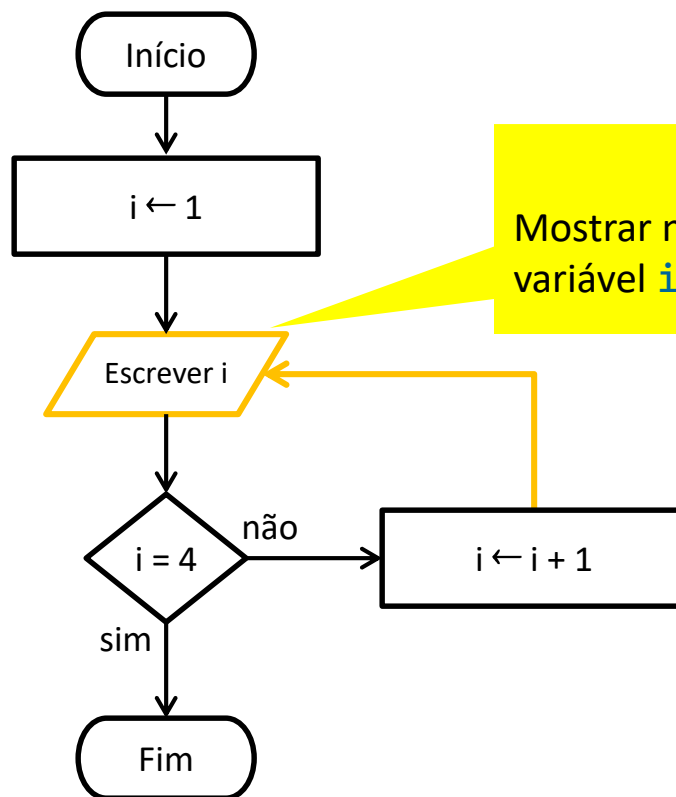
- como  $i$  não é igual a 4, então somar 1 a  $i$
- $i$  passa a ter o valor 2



# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 2$



## Explicação

Mostrar novamente o valor da variável  $i$  (neste caso, é 2)

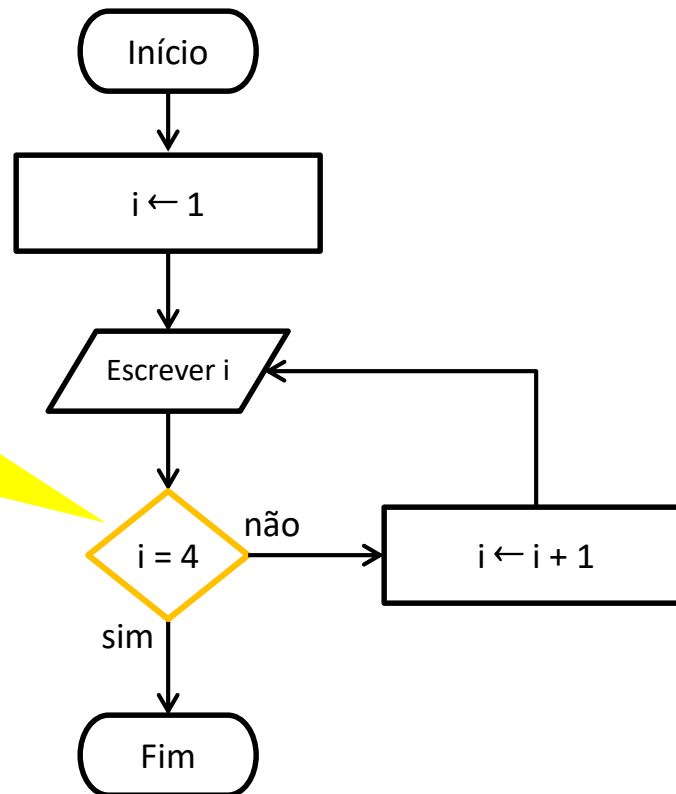
# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 2$

## Explicação

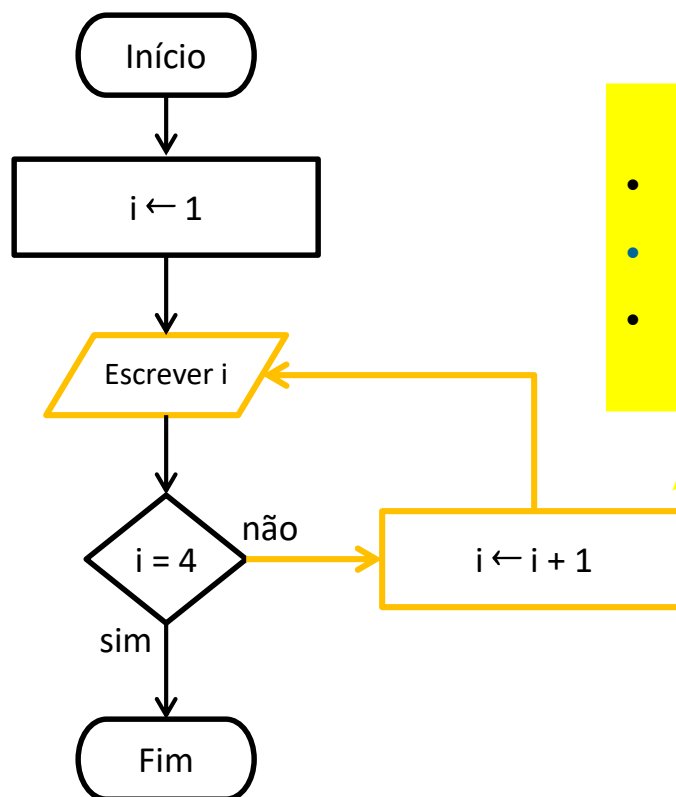
Verificar novamente se  $i$  é igual a 4 (neste caso não é, tem o valor 2)



# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 3$



## Explicação

- somar novamente 1 a  $i$
- $i$  passa a ter o valor 3
- mostrar novamente o valor de  $i$

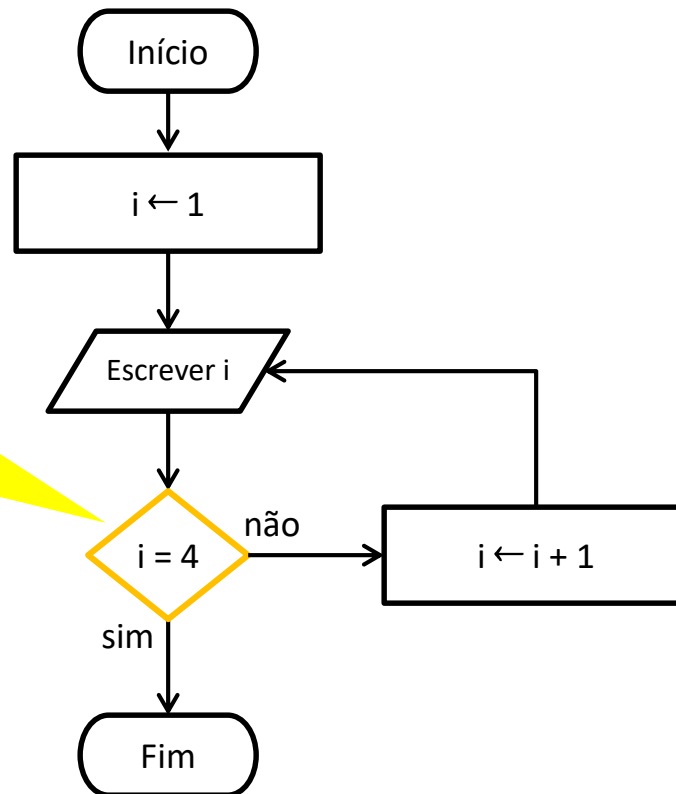
# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 3$

## Explicação

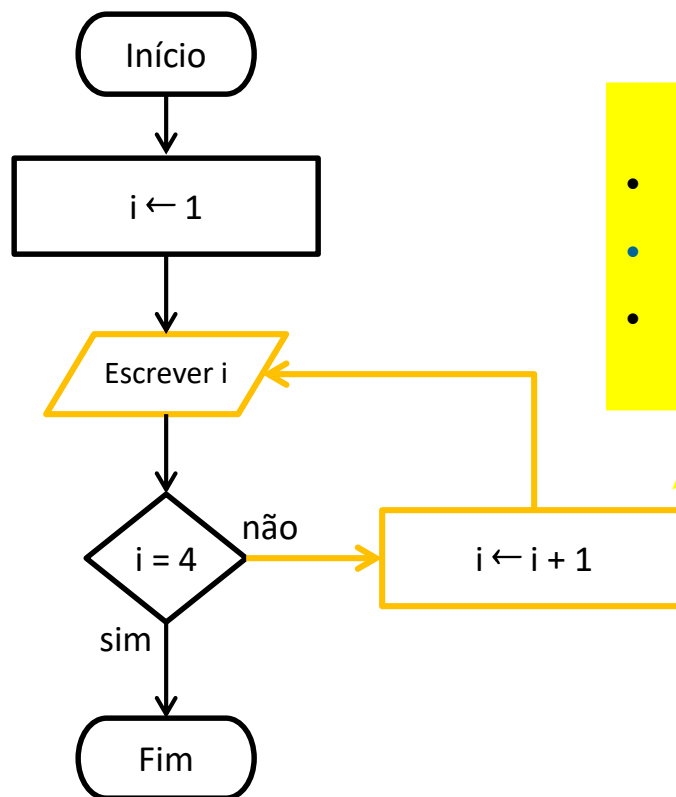
Verificar novamente se  $i$  é igual a 4 (neste caso não é, tem o valor 3)



# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 4$



## Explicação

- somar novamente 1 a  $i$
- $i$  passa a ter o valor 4
- mostrar novamente o valor de  $i$

# Fluxogramas: Repetição

- Exemplo: mostrar os números de 1 a 4

$i = 4$

## Explicação

Verificar se  $i$  é igual a 4:  
como é verdade, então  
terminar

