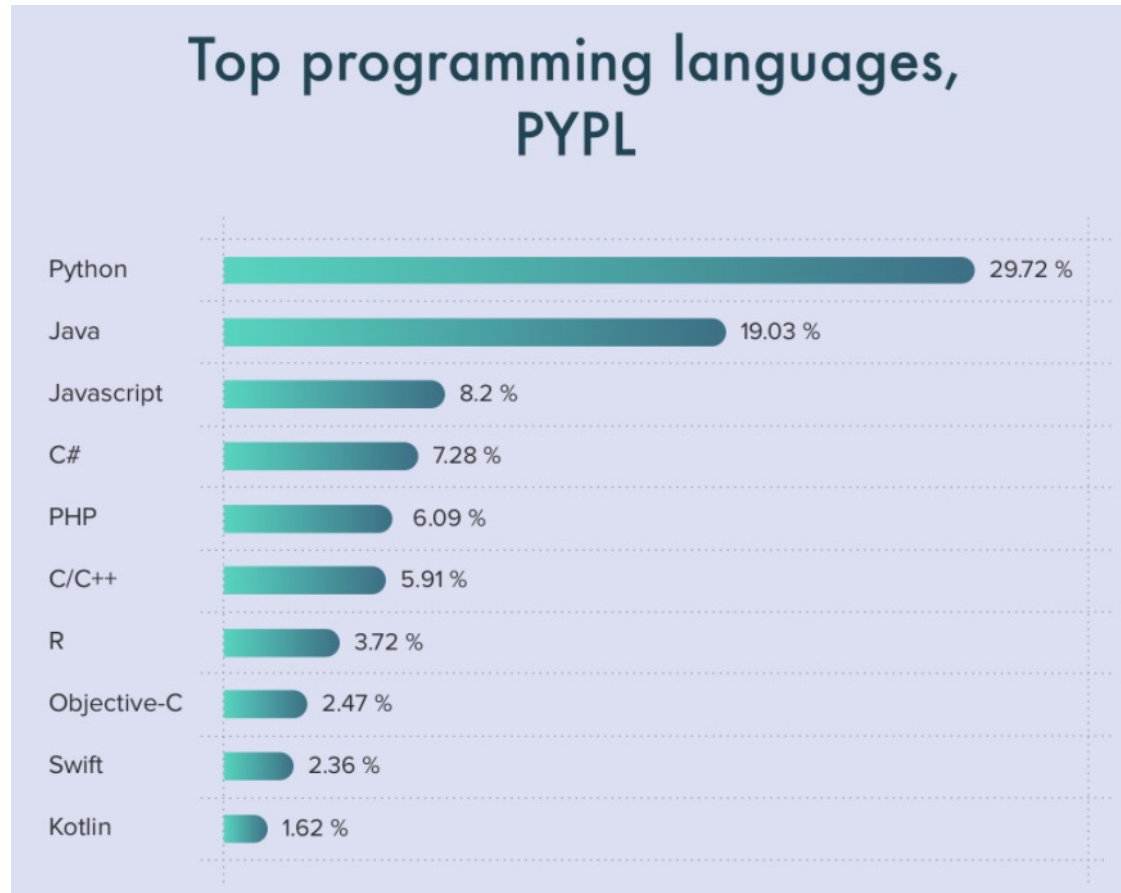




Introdução à Programação em Python

Porquê Python?



Popularity of Programming Language (02-2022):

<https://www.cleveroad.com/blog/programming-languages-ranking>

Documentação

WWW!

w3schools.com

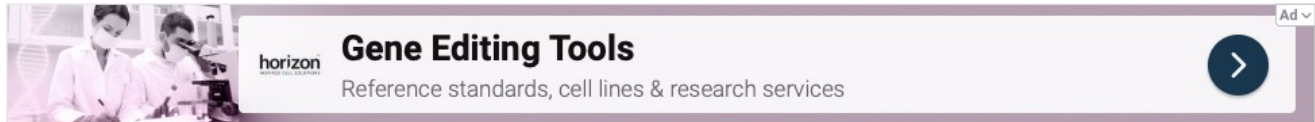
THE WORLD'S LARGEST WEB I

HTML CSS JAVASCRIPT SQL PYTHON PHP BOOTSTRAP HOW TO MORE REFERENCES EXERCISE

Python Tutorial

Python HOME

- Python Intro
- Python Get Started
- Python Syntax
- Python Comments
- Python Variables
- Python Data Types
- Python Numbers
- Python Casting
- Python Strings
- Python Booleans
- Python Operators
- Python Lists
- Python Tuples
- Python Sets
- Python Dictionaries
- Python If...Else
- Python While Loops
- Python For Loops
- Python Functions
- Python Lambda



horizon **Gene Editing Tools**
Reference standards, cell lines & research services

Python Tutorial

< Home

Next >

Python is a programming language.

Python can be used on a server to create web applications.

Start learning Python now >

Learning by Examples

With our "Try it Yourself" editor, you can edit the code and view the result.

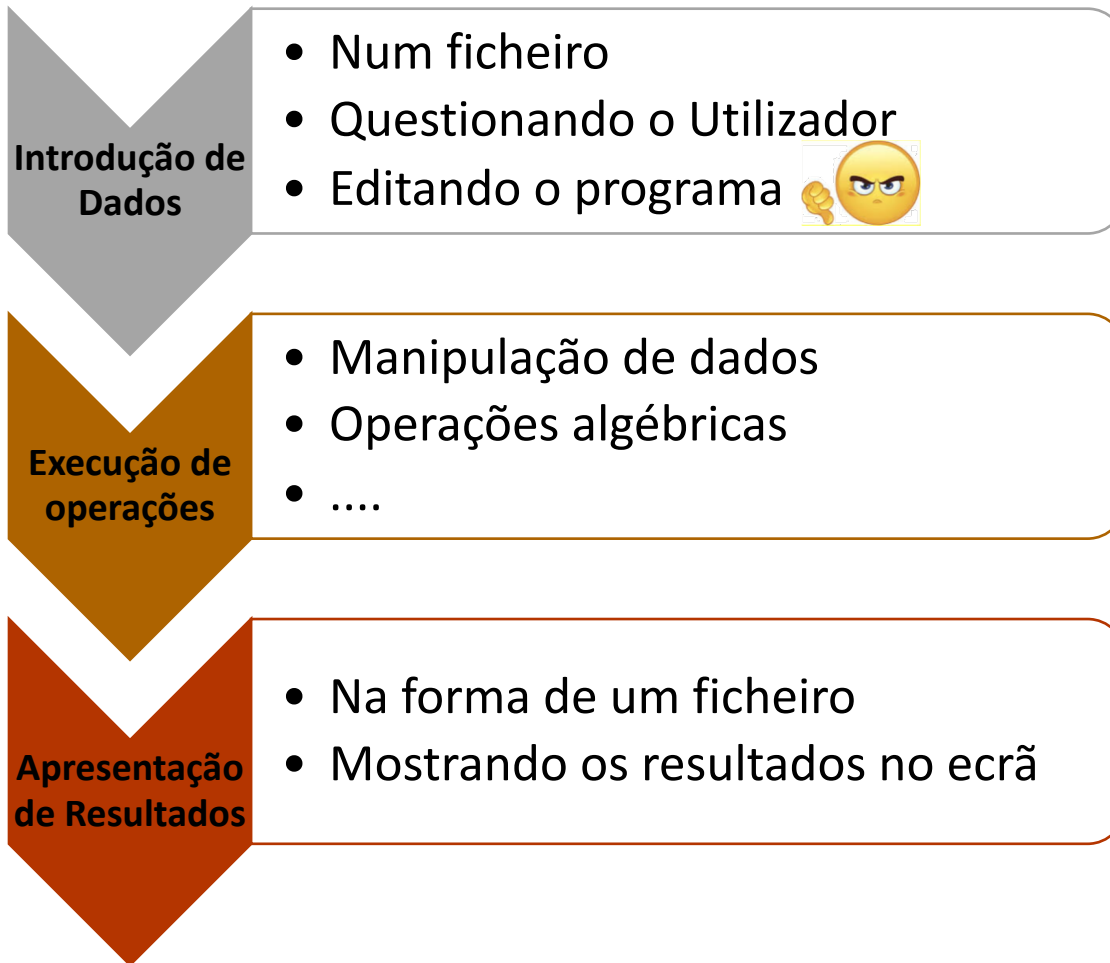
<https://www.w3schools.com/python/default.asp>

O que é um Programa?

É uma sequencia de instruções que realizam uma série de operações, devolvendo no final um resultado ao utilizador.

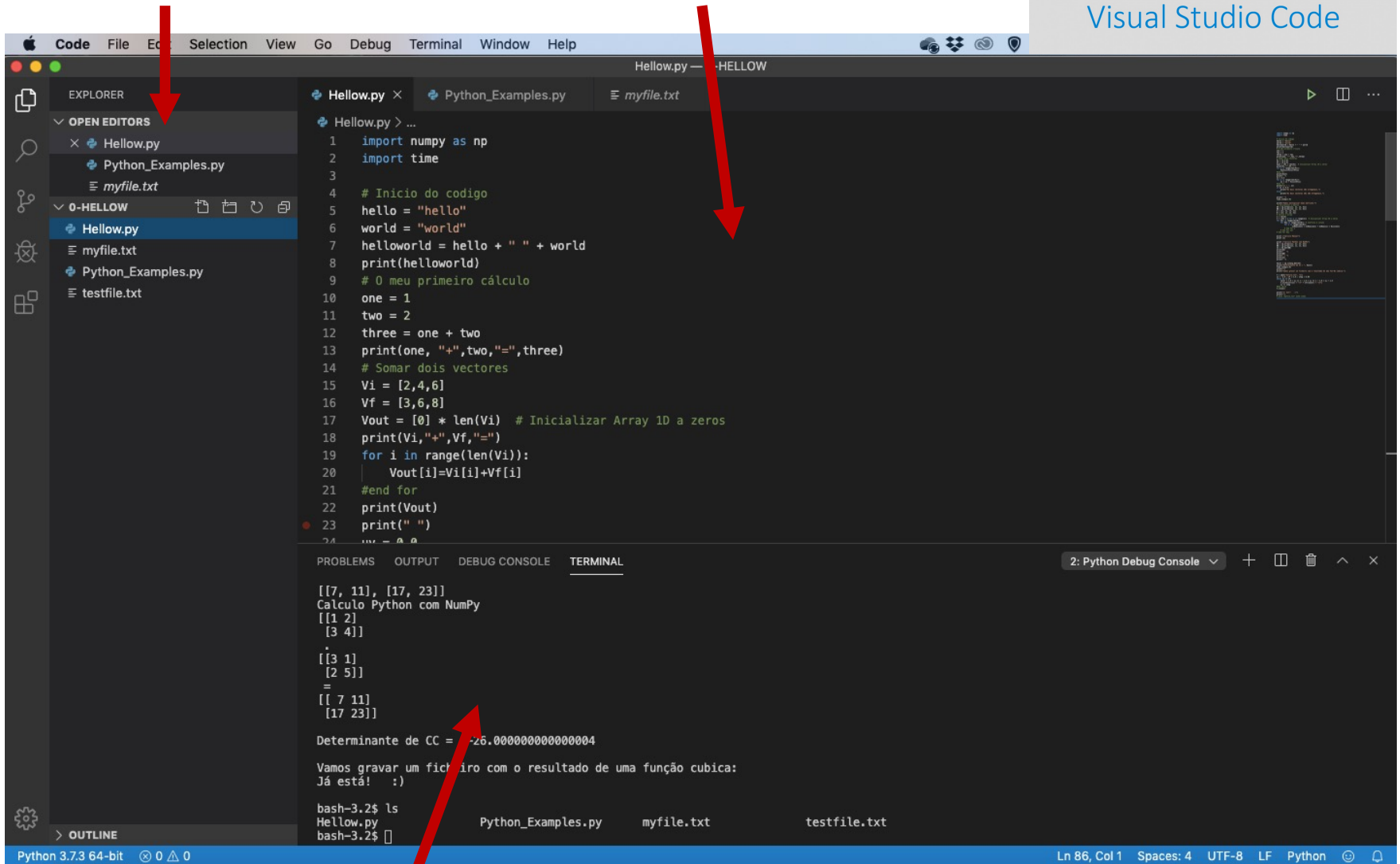
```
print("Química Computacional 2022!")
```

Estrutura de um Programa: (Fluxograma)



Gestor de Ficheiros

Editor de Programas



Terminal e Consola de Erros

<https://code.visualstudio.com/>

Como criar um programa no VS Code

1) Abrir o VS Code e criar um diretório, por exemplo

“PROGRAMAS”

2) Criar um ficheiro “NOME_A_ESCOLHA.py” e escreva o código:

```
print(“Química Computacional 2022!”)
```

3) Corra o programa (F5).

ATENÇÃO: o nome do ficheiro deve terminar em “.py”

Variáveis

Em programação, uma variável é um objeto (uma posição, frequentemente localizada na memória do computador) capaz de reter e representar um valor ou expressão.

Em Python, estas são iniciadas na primeira vez que são utilizadas, podendo conter texto, números inteiros, reais, ou ser booleanas.

Variáveis

```
x = 5          (inteiro)
y = "Joao"    (texto)
Z = 0.5        (número de virgula flutuante)
Z1 = Z2 = 6
Flag = True    (booleano)
```

Para pedir um valor para uma variável, *v*, ao utilizador pode usar-se:

```
v = input()
```

Operadores Aritméticos

Operador	Função	Exemplo
+	Adição	$x + y$
-	Subtração	$x - y$
*	Multiplicação	$x * y$
/	Divisão	x / y
**	Expoente	$x ** y$
%	Resto	$x \% y$

Ex: Determinar se um ano é bissexto: 2016, 2020, 2024, ...

Múltiplos de 4

Logo se o resto de $\text{ano}/4 \Rightarrow 0$, é bissexto.

Operadores de Atribuição

Operador	Exemplo	Igual a:
=	$x = 5$	$x = 5$
+=	$x += 3$	$x = x + 3$
-=	$x -= 3$	$x = x - 3$
*=	$x *= 3$	$x = x * 3$
/=	$x /= 3$	$x = x / 3$
%=	$x \% = 3$	$x = x \% 3$
**=	$x ** = 3$	$x = x ** 3$

Ciclos

Ciclos **for**:

```
for i in range(5):  
    Faz algo!!!
```

Ciclos **while**:

```
xi = 0.0 ; xf = 3.0  
step = 0.05
```

```
while xi <= xf:  
    Faz algo !!!  
    xi += step
```

Indentação!!!

Este código:

```
xi = 0.0 ; xf = 3.0
step = 0.05
while xi <= xf:
    print (xi)
    xi += step
```

Não é igual a:

```
xi = 0.0 ; xf = 3.0
step = 0.05
while xi <= xf:
print (xi)
xi += step
```

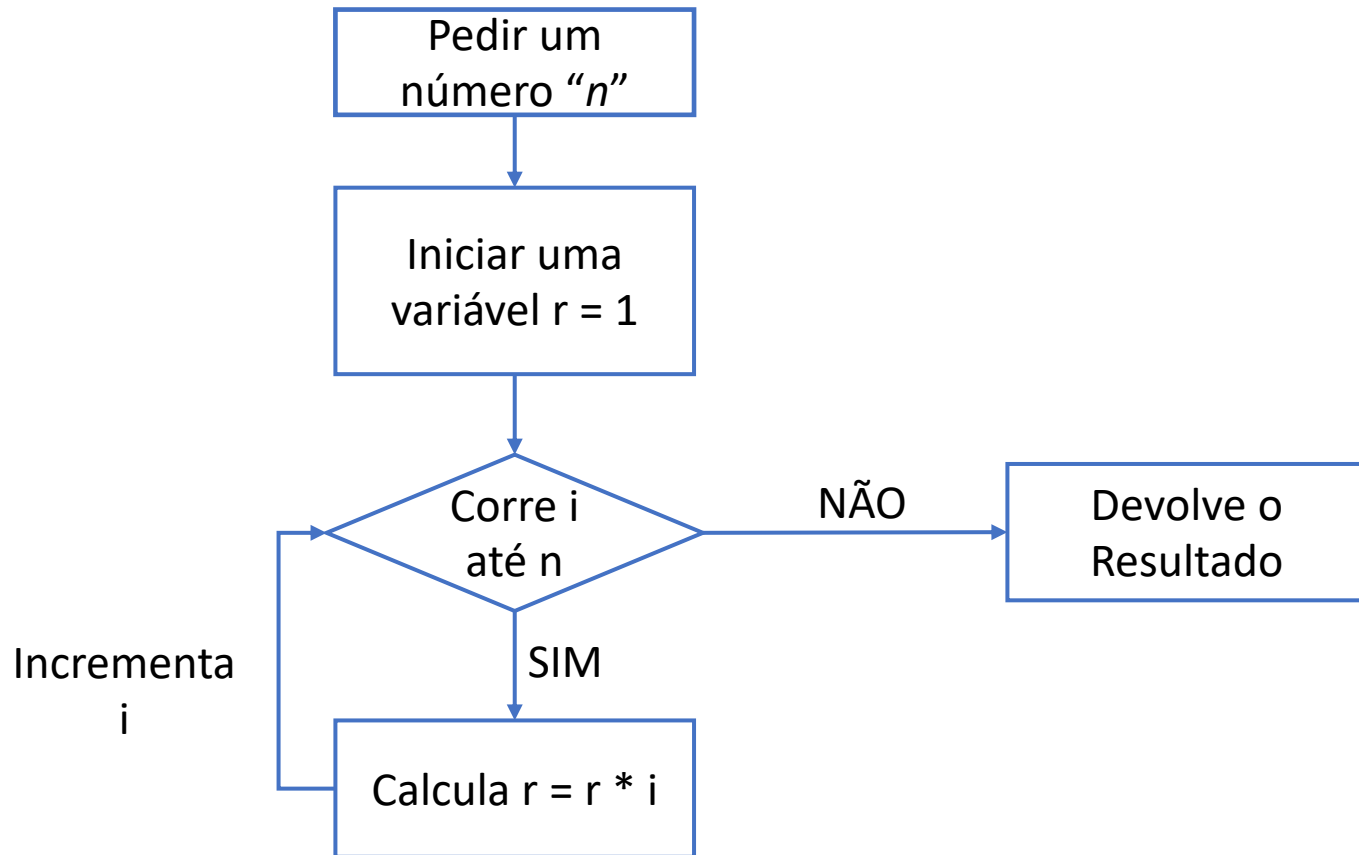
Exemplo

1) Criar um programa que calcule o fatorial de um número.

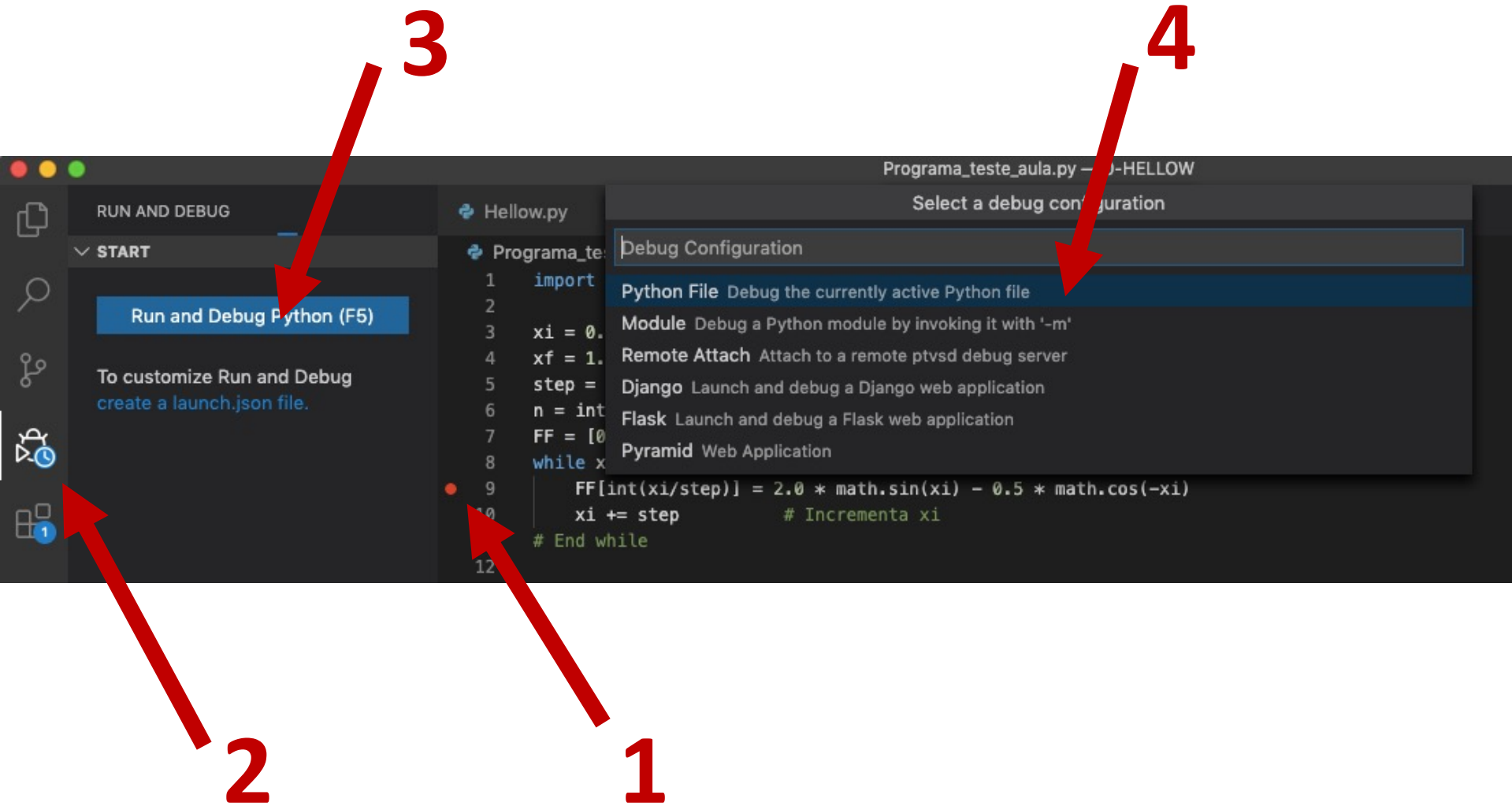
$$n! = \prod_{k=1}^n k = n \times (n - 1) \times (n - 2) \times \dots \times 3 \times 2 \times 1, \quad \forall n \in \mathbb{N}$$

Exemplo (fluxograma)

$$n! = \prod_{k=1}^n k = n \times (n-1) \times (n-2) \times \dots \times 3 \times 2 \times 1, \quad \forall n \in \mathbb{N}$$



DEBUG



Testes Lógicos

Função **IF**:

```
if V > 0:
```

```
↔ # Faz algo se a condição é verdadeira
```

```
else:
```

```
↔ # Faz algo se a condição é falsa
```



Cuidado com a
Indentação

Operadores de Comparação

Operador	Nome	Exemplo
==	Igual	$x == y$
!=	Diferente	$x != y$
>	Maior que	$x > y$
<	Menor que	$x < y$
>=	Maior ou igual	$x >= y$
<=	Menor ou igual	$x <= y$

Outros Operadores

Operador	Descrição	Exemplo
and	Devolve “true” se duas condições são verdadeiras	$x > 5$ and $x < 10$
or	Devolve “true” se uma das condições for verdadeiras	$x < 5$ or $x > 10$
not	Reverso do resultado, i.e., “False” se a condição é verdadeira	not($x < 5$ and $x > 10$)
is	Devolve “true” se os valores são iguais	x is y
is not	Devolve “true” se os valores não são iguais	x is not y

Exemplo

Criar um programa que calcule o fatorial de um número.

$$n! = \prod_{k=1}^n k = n \times (n - 1) \times (n - 2) \times \dots \times 3 \times 2 \times 1, \quad \forall n \in \mathbb{N}$$

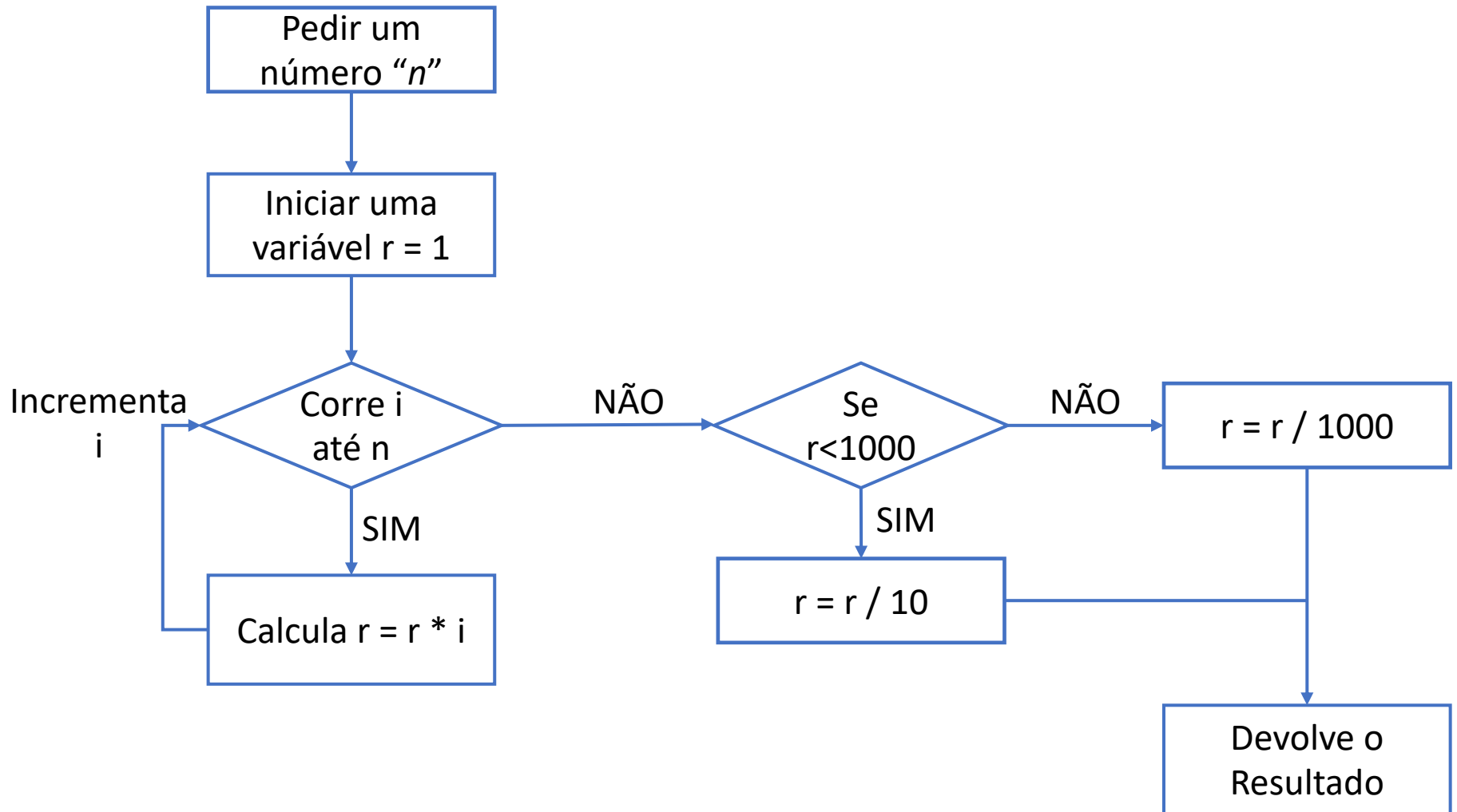
e que recalcule o resultado de acordo com:

se <1000 divide o valor por 10

se >1000 divide o valor por 1000

Exemplo (fluxograma)

$$n! = \prod_{k=1}^n k = n \times (n-1) \times (n-2) \times \dots \times 3 \times 2 \times 1, \quad \forall n \in \mathbb{N}$$



Funções

The image shows a screenshot of the Microsoft Excel interface. At the top, the title bar displays "Book1 - Excel" and the "AutoSave" toggle is set to "Off". The "Home" ribbon is active, showing the "Clipboard" group with "Undo", "Paste", "Cut", "Copy", and "Format Painter" options. The "Font" group shows font size "11" and bold, italic, and underline options. Below the ribbon, the formula bar contains the text "=average(" and a tooltip for the AVERAGE function is visible, showing the syntax "AVERAGE(number1, [number2], ...)". In the worksheet grid, cell A1 is selected and contains the text "=average(".

	A	B	C		G
1	=average(

Funções

- Uma função é um bloco de código que só é executado quando é chamado.
- É possível passar dados, conhecidos como parâmetros, para uma função.
- Uma função pode retornar dados como resultado.

```
def my_function(fname):  
    print(fname + " Refsnes")  
    return 10
```

Cuidado com a
Indentação

Exemplo

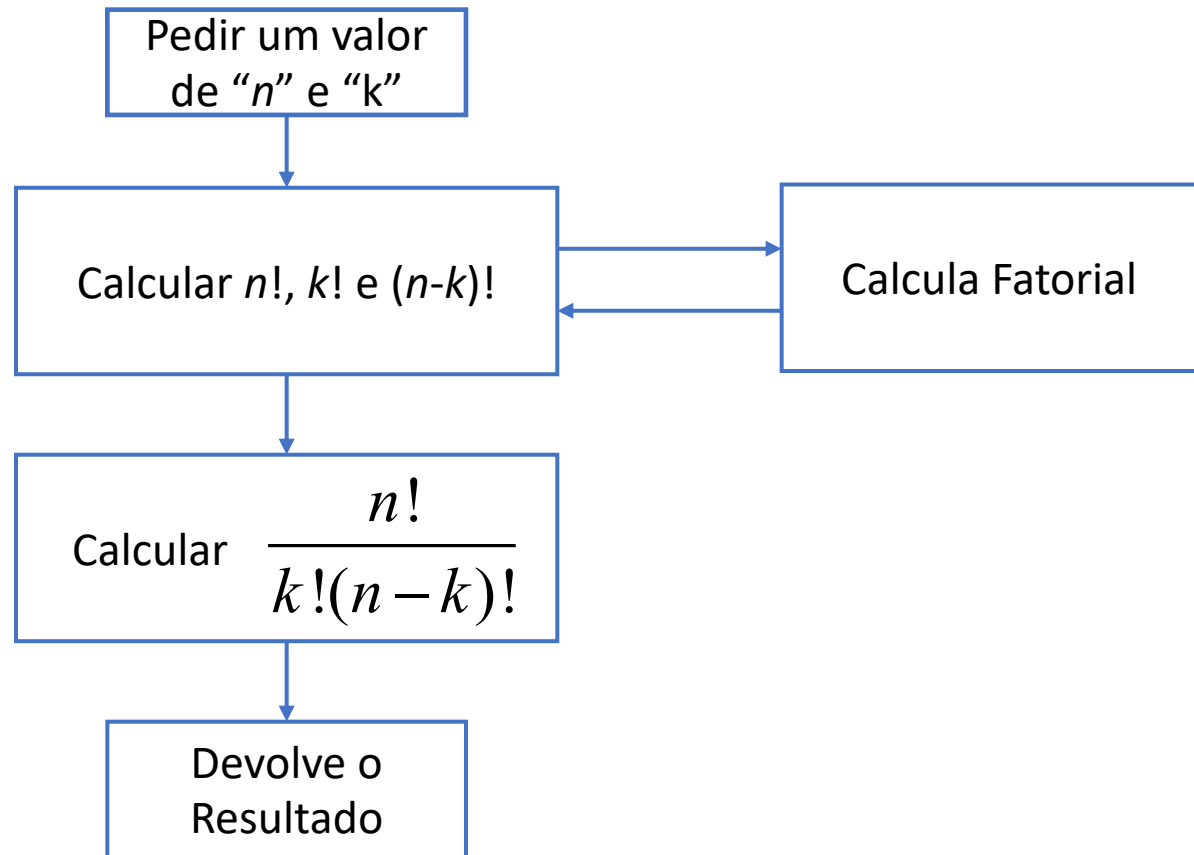
O **coeficiente binomial**, também chamado de **número binomial**, de um número n , na classe k , consiste no número de combinações de n termos, k a k , definido como:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Construir um programa que calcule número de combinações de n termos, k a k .

Exemplo

(fluxograma)



Encadeamento de Funções

```
if r>100:  
    j="TR"  
elif r>10:  
    j="BR"  
elif r>0:  
    j="JR"
```

```
for j in range(10):  
    for k in range(5):  
        r = j + k
```

Exemplo:

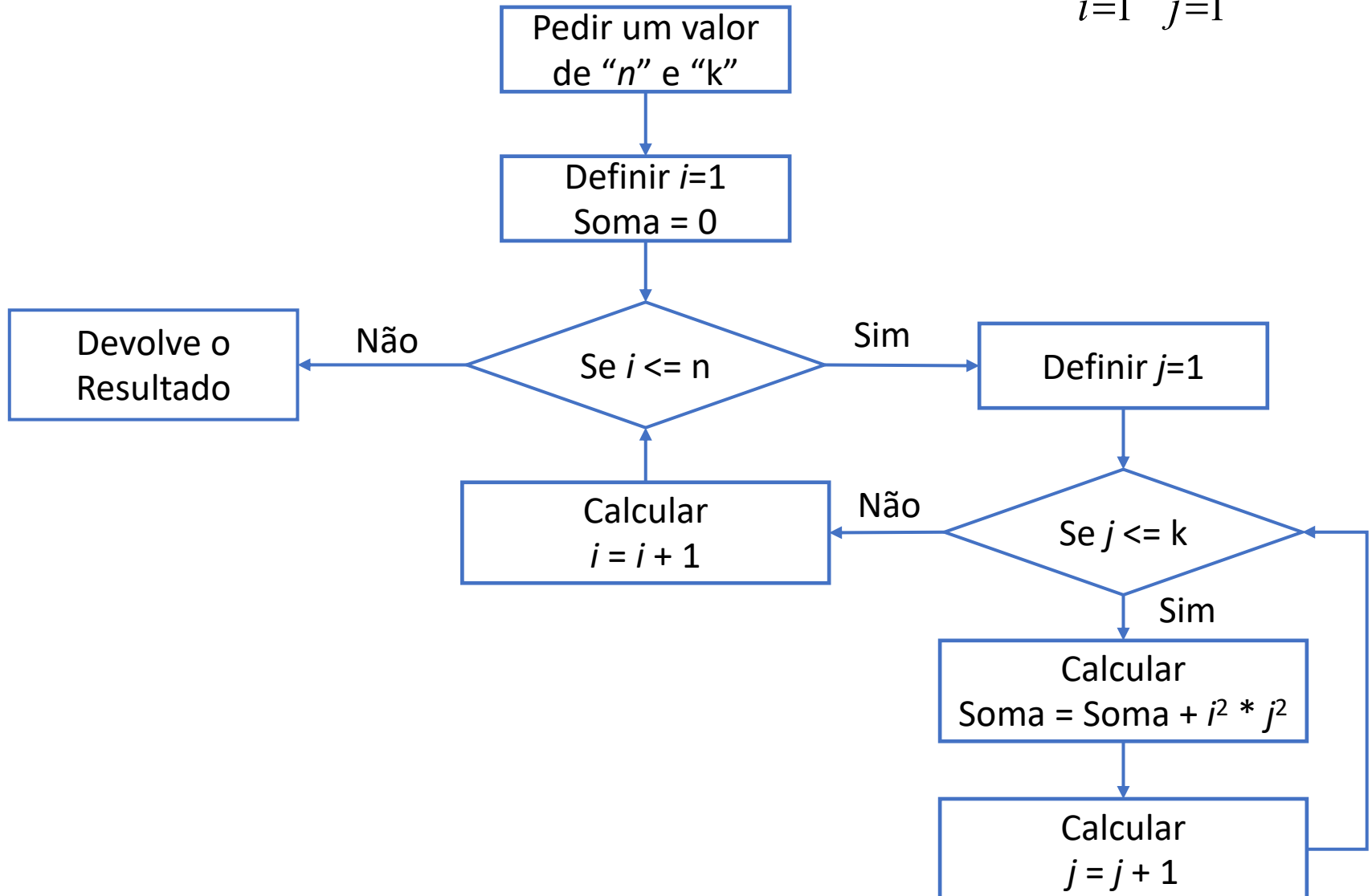
Construir um programa em Python para calcular:

$$\sum_{i=1}^n \sum_{j=1}^k i^2 \cdot j^2$$

Exemplo

(fluxograma)

$$\sum_{i=1}^n \sum_{j=1}^k i^2 \cdot j^2$$



Arrays

Arrays são utilizados para guardar múltiplos valores numa única variável.

```
Numero = [1.0, 2.0, 3.1]
```

Arrays

Quando necessários, os valores são “chamados” indicando a sua posição no array.

```
carro = ["Ford", "Volvo", "BMW"]  
         0       1       2
```

```
print(carro[1]) → Devolve "Volvo"
```

Atenção: os índices dos valores num array são numerados entre 0 e N-1, onde N é o tamanho do array.

Para saber o tamanho de um array usa-se:

```
len(carro)
```

Arrays Multidimensionais

Array's NxM:

$$A = \begin{matrix} & \underbrace{\hspace{2cm}}_{N=0} & & \underbrace{\hspace{2cm}}_{N=1} \\ & [1, 2], & & [3, 4] \\ & \uparrow \quad \uparrow & & \uparrow \quad \uparrow \\ M = & 0 \quad 1 & & 0 \quad 1 \end{matrix}$$

Neste caso os valores são utilizados como:

```
print("A22 = ", A[1][1]) → Devolve "4"
```

Arrays

No entanto, na maioria das vezes não se sabe quantos valores serão colocados no array antes de iniciar o programa.

Nestes casos, utiliza-se, por exemplo:

`n = 2`

`D = [[0] for i in range(n)]` → Array com N entradas

`C = [[0] * n for i in range(n)]` → Array N x N

Gravar Ficheiros

Gravar dados num ficheiro:

w – escreve o ficheiro
r – lê o ficheiro
a – adiciona dados a
ficheiro já existente

1) Abrir o ficheiro:

```
f = open("testfile.txt", "w")
```

Nome do ficheiro

2) Gravar linha de texto:

```
f.write(str(xi) + "\t" + str(valor) + "\n")
```

tab

"Fecha" a linha

3) Fechar o ficheiro:

```
f.close()
```

Converte número para texto.
A função inversa é **float**

Funções Matemáticas

É necessário importar bibliotecas de matemática. Para o efeito, no início do programa adicionar:

```
import math
```

Por exemplo, utilizar as funções como:

<code>math.cos(x)</code>	Cosseno de x , em radianos
<code>math.sin(x)</code>	Seno de x , em radianos
<code>math.acos(x)</code>	Arco cosseno de x , em radianos
<code>math.asin(x)</code>	Arco seno de x , em radianos
<code>math.tan(x)</code>	Tangente de x , em radianos
<code>math.atan(x)</code>	Arco tangente de x , em radianos
<code>math.exp(x)</code>	Exponencial de base e
<code>math.log(x[, base])</code>	Logaritmo de x (para a base e)
<code>math.sqrt(x)</code>	Raiz quadrada de x

Exemplo

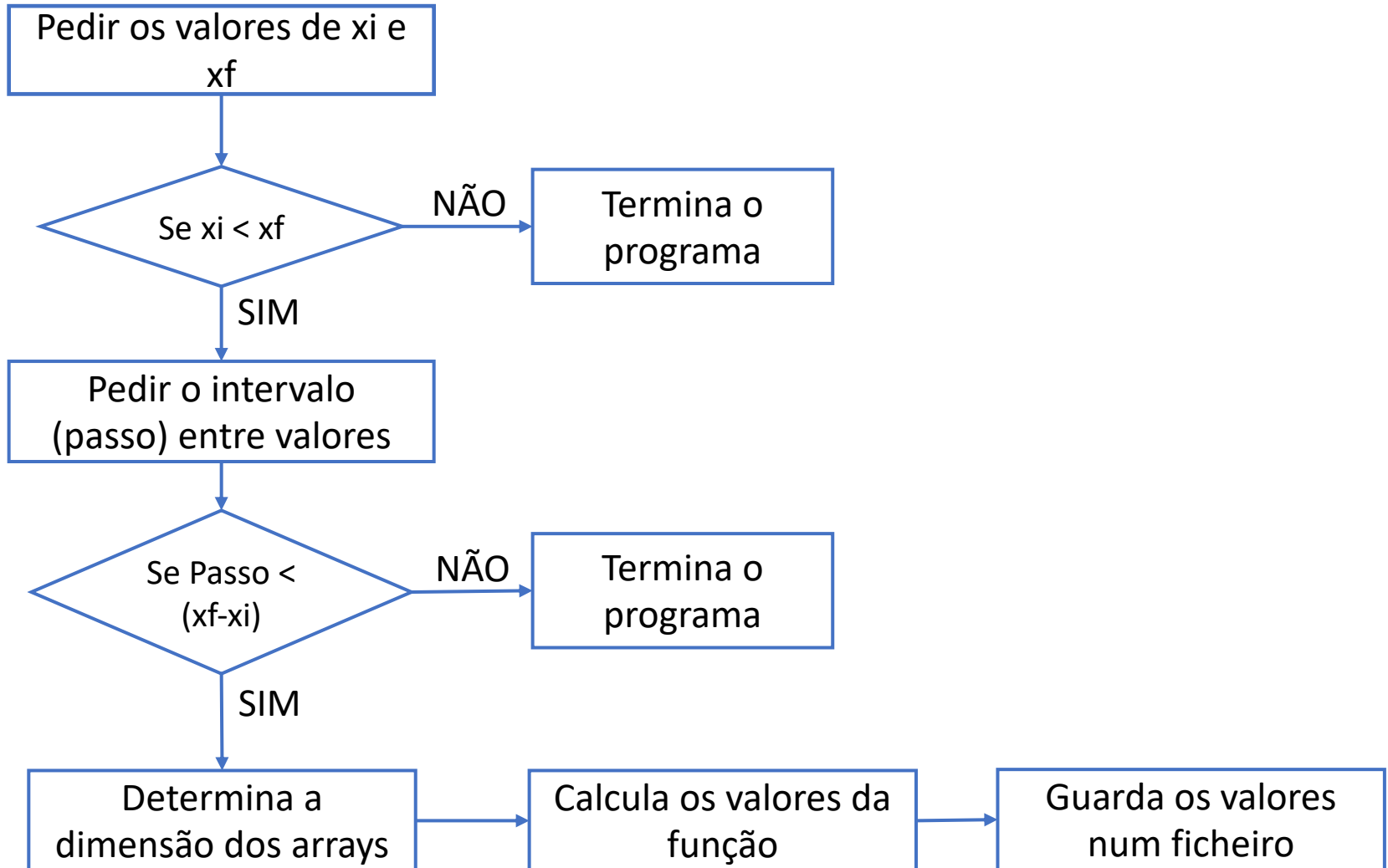
1) Criar um programa que calcule

$$f(x) = 1.0 \sin(x) - 0.5 \cos(-2x)$$

no intervalo de x_i a x_f , em intervalos δx , e que guarde os resultados num ficheiro.

Exemplo

(fluxograma)



Programa de Exemplo

```
import math # Importa as bibliotecas de matemática

print ("Valor Xi = ") # Pergunta pelo valor de Xi
xi = float(input()) # Pedes o valor Xi ao
                    # utilizador e converte para valor

print ("Valor Xf = ") # Pergunta pelo valor de Xf
xf = float(input()) # Pedes o valor Xf ao
                    # utilizador e converte para valor

if xf < xi: # Verifica a validade dos valores
introduzidos
    print ("xf deve ser menor que xi. Tente
novamente!")
    quit() # Termina o programa

print ("Passo = ") # Pergunta intervalo entre
                    # valores calculados

step = float(input()) # Defini o step em cada
                       # ciclo e converte para valor

if xf-xi < step: # Verifica a validade do step
introduzido
    print("Utilizar um step mais pequeno.")
    quit() # Termina o programa

n = int((xf-xi)/step) # Determina o tamanho do
                       # array que vai necessitar

n = int((xf-xi)/step) # Determina o tamanho do
                       # array que vai necessitar
FF = [0] * n # Aloca a memoria necessária para o
              # array FF

i = 0 # Variável de controlo do ciclo

while i < n: # Calcula os valores para a função:
              # 1.0 sin(x) - 0.5 cos(-2x)
    x = xi + i * step
    FF[i] = 1.0 * math.cos(x) - 0.5 *
            math.cos(-2*x)
    i += 1 # Incrementa i

# Grava os dados num ficheiro
f = open("testfile.txt", "w") # Abre o ficheiro
for i in range(len(FF)): # Vai correr todos os
                          # valores no array
    x = xi + i * step # Calcula o valor de x
    # Escreve cada linha com o valor de x e f(x)
    f.write(str(x) + "\t" + str(FF[i]) + "\n")

f.close() # Fecha o ficheiro
print ("DONE.")
```