

# ALGORITMIA E ESTRUTURAS DE DADOS

## ESTRUTURAS ITERATIVAS/REPETITIVAS

LICENCIATURA EM  
TECNOLOGIAS E SISTEMAS DE INFORMAÇÃO PARA A WEB  
#ESMAD #P.PORTO

- ❑ Estruturas iterativas/repetitivas
  - ❑ for
  - ❑ while
- ❑ Quebras de ciclo
  - ❑ break
  - ❑ continue



# Python for Loop

## ❖ Estruturas de Repetição | Iterativas

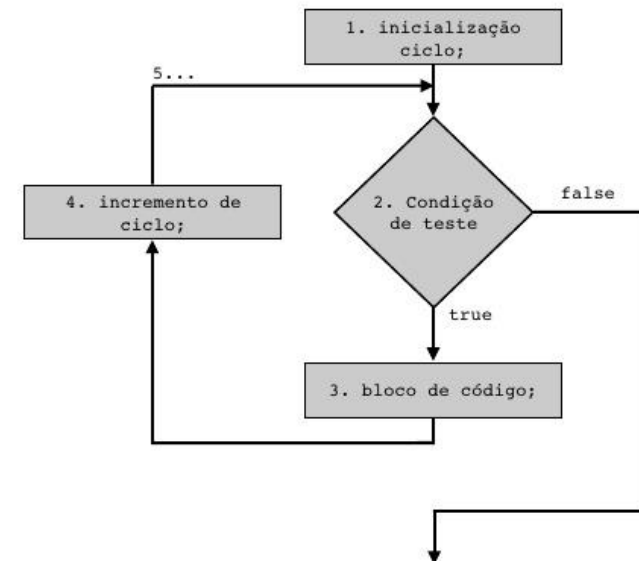
### *for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

A iteração (repetição) de um ciclo *for* pode ser associada a:

- ❑ Valores numéricos (*range*)
- ❑ Texto (*strings*)
- ❑ Sequências encadeadas  
(executar a repetição para cada valor de um sequência:  
array, lista, tupla, etc.)

**for:** como funciona em fluxograma



## ❖ Estruturas de Repetição | Iterativas

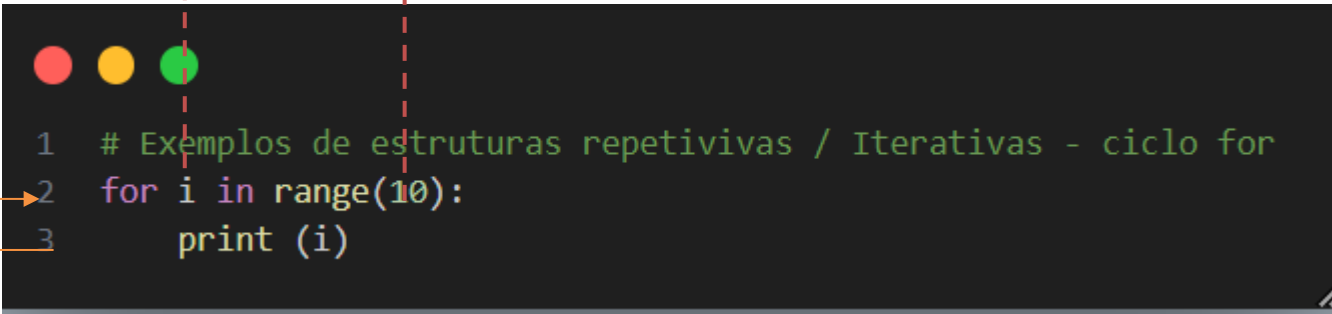
*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

**range**: permite especificar o nº de vezes que o ciclo se repete

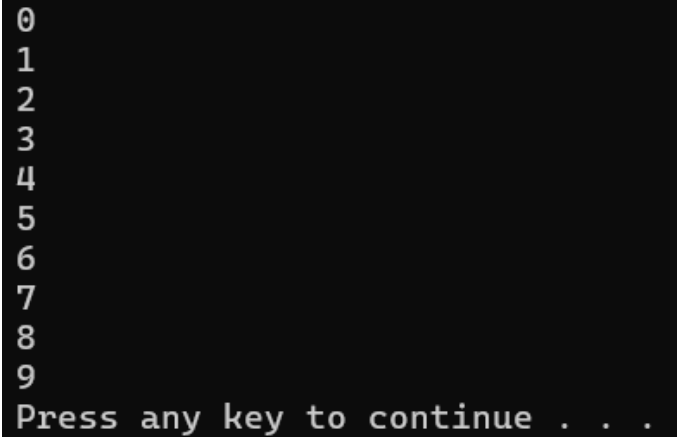
Variável contadora  
do ciclo

Repete o ciclo **10 vezes**. Começa com i a **0** e termina em **9**



```
1 # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2 for i in range(10):
3     print(i)
```

The code is shown in a dark-themed editor with three colored window control buttons (red, yellow, green) at the top left. An orange bracket on the left side of lines 2 and 3 indicates the loop body. Two vertical dashed red lines are present: one connects the text 'Variável contadora do ciclo' to the variable 'i' in line 2, and the other connects 'Repete o ciclo 10 vezes. Começa com i a 0 e termina em 9' to the value '10' in line 2.



```
0
1
2
3
4
5
6
7
8
9
Press any key to continue . . .
```

The terminal output shows the numbers 0 through 9 printed on separate lines, followed by the prompt 'Press any key to continue . . .'.

## ❖ Estruturas de Repetição | Iterativas

*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

**range**: permite especificar o nº de vezes que o ciclo se repete

Repete o ciclo **3 vezes**. Começa com i a **0** e termina com i a **2**

```
1 # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2 for i in range(3):
3     numero = int(input("Indique o {:n}º número: ".format(i+1)))
```

```
Indique o 1º número: 10
Indique o 2º número: 20
Indique o 3º número: 30
Press any key to continue . . .
```

## ❖ Estruturas de Repetição | Iterativas

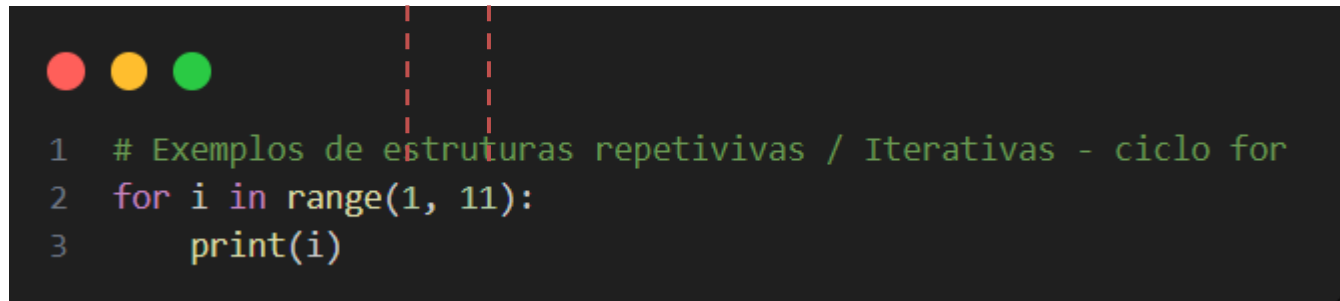
*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

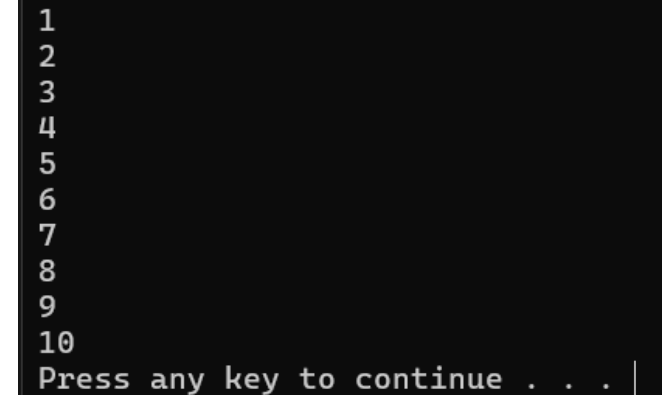
**range**: permite especificar o nº de vezes que o ciclo se repete

Repete o ciclo **10 vezes**. Começa com i a **1** e termina em **10**

*valor inicial*      *valor final* (é excluído)



```
1 # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2 for i in range(1, 11):
3     print(i)
```



```
1
2
3
4
5
6
7
8
9
10
Press any key to continue . . . |
```

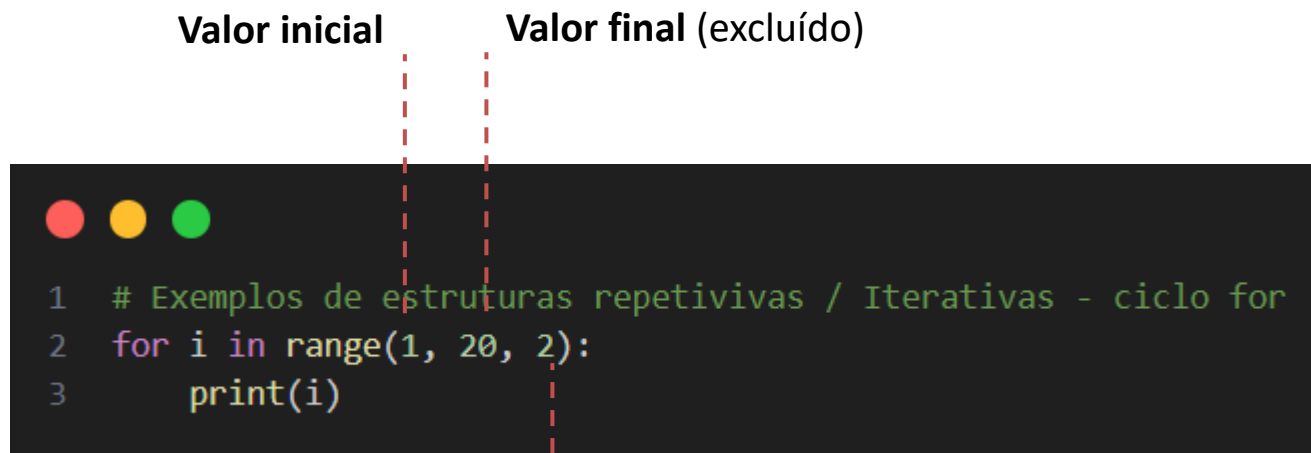


## ❖ Estruturas de Repetição | Iterativas

*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

**range**: permite especificar o nº de vezes que o ciclo se repete

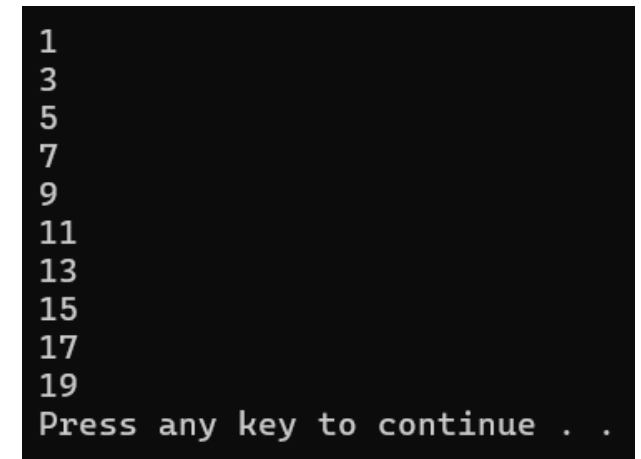


The image shows a code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The code is as follows:

```
1 # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2 for i in range(1, 20, 2):
3     print(i)
```

Annotations with dashed red lines:

- Valor inicial**: points to the first argument '1' in the range function.
- Valor final (excluído)**: points to the second argument '20' in the range function.
- Step / incremento da variável i em cada iteração**: points to the third argument '2' in the range function.



The image shows a terminal window with a dark background. It displays the output of the Python code, which is a list of odd numbers from 1 to 19, one per line. At the bottom, it prompts the user to press a key to continue.

```
1
3
5
7
9
11
13
15
17
19
Press any key to continue . .
```

## ❖ Estruturas de Repetição | Iterativas

*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

**range**: permite especificar o nº de vezes que o ciclo se repete

Valor inicial      Valor final (excluído)

```
1 # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2 for i in range(20, 0, -2):
3     print(i)
```

Step / incremento da variável i em cada iteração

```
20
18
16
14
12
10
8
6
4
2
Press any key to continue . . . |
```



## ❖ Estruturas de Repetição | Iterativas

*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

**strings**: podemos iterar strings, pois consistem em sequências de caracteres



```
1 # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2 nome = "Carla caldeira"
3
4 for caracter in nome:
5     print(caracter)
```

```
C
a
r
l
a

c
a
l
d
e
i
r
a
Press any key to continue . . . |
```

## ❖ Estruturas de Repetição | Iterativas

*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

**strings**: podemos iterar strings, pois consistem em sequências de caracteres

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2  nome = "Carla caldeira"
3
4  for caracter in nome:
5      print(caracter, end="")
6  print("\n")
```

```
Carla caldeira
Press any key to continue . . . |
```

## ❖ Estruturas de Repetição | Iterativas

*for*

Permite implementar um ciclo para executar um conjunto de instruções de forma repetida

**Sequências:** podemos iterar estruturas que representam sequências de dados

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo for
2
3  paises = ["Portugal", "Espanha", "Itália", "Croácia"]
4  for pais in paises:
5      print(pais)
6
```

```
Portugal
Espanha
Itália
Croácia
Press any key to continue . . .
```

**Nota:** voltaremos a estes casos mais tarde, quando usarmos estruturas de dados deste tipo

## ❖ Estruturas de Repetição | Iterativas

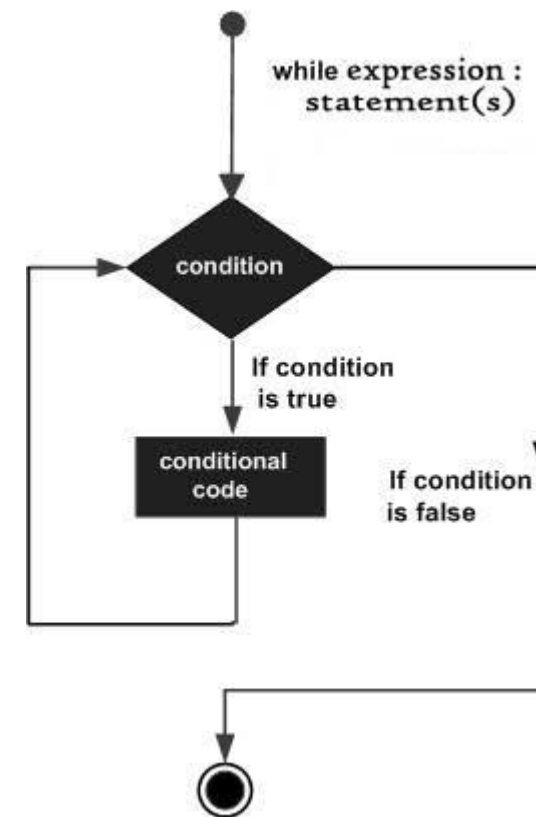
### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

A condição é testada, repetidamente, antes de iniciar cada iteração do ciclo

Quando a condição falha, a execução segue para a linha de código imediatamente a seguir ao fim do ciclo while

```
while expression:  
    statement(s)
```



## ❖ Estruturas de Repetição | Iterativas

### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

A condição é testada, repetidamente, antes de iniciar cada iteração do ciclo

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  numero=1
4  while numero<=10:
5      print(numero)
6      numero+=1
```

```
1
2
3
4
5
6
7
8
9
10
Press any key to continue . . .
```

## ❖ Estruturas de Repetição | Iterativas

### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  numero=int(input("Indique um número entre [0-20]: "))
4
5  while numero<=0 or numero >20:
6      print("Indicou um número inválido. Tente novamente!\n")
7      numero=int(input("Indique um número entre [0-20]: "))
8
```

```
Indique um número entre [0-20]: 21
Indicou um número inválido. Tente novamente!

Indique um número entre [0-20]: -1
Indicou um número inválido. Tente novamente!

Indique um número entre [0-20]: 15
Press any key to continue . . . |
```

## ❖ Estruturas de Repetição | Iterativas

### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  tabuada=int(input("Imprimir a tabuada dos: "))
4  numero=1
5  while numero<11:
6      print("{:n} * {:n} = {:n}".format(tabuada, numero, tabuada*numero))
7      numero+=1
```

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  tabuada=int(input("Imprimir a tabuada dos: "))
4  numero=1
5  while numero<11:
6      print(f'{tabuada} * {numero} =', tabuada*numero)
7      numero+=1
8
```

```
Imprimir a tabuada dos: 7
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
Press any key to continue . . . |
```



## ❖ Estruturas de Repetição | Iterativas

### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  import random    # biblioteca que permite gerar números aleatoriamente
4
5  numeroGerado=random.randint(0,20)  # gera um nº inteiro entre 0 e 20
6                                     # inclui os limites inferior e superior
7  palpite = int(input("Indique o seu palpite:"))
8  while numeroGerado != palpite:
9      print("Não acertou! :( tente novamente!\n")
10     palpite = int(input("Indique o seu palpite:"))
11
```

## ❖ Estruturas de Repetição | Iterativas

### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  import random  # biblioteca que permite gerar números aleatoriamente
4
5  numeroGerado=random.randint(0,20)  # gera um nº inteiro entre 0 e 20
6                                     # inclui os limites inferior e superior
7  palpite = int(input("Indique o seu palpite:"))
8  while numeroGerado != palpite:
9      print("Não acertou! :( tente novamente!\n")
10     palpite = int(input("Indique o seu palpite:"))
11
```

```
Indique o seu palpite:10
Não acertou! :( tente novamente!

Indique o seu palpite:12
Não acertou! :( tente novamente!

Indique o seu palpite:18
Não acertou! :( tente novamente!

Indique o seu palpite:
```

## ❖ Estruturas de Repetição | Iterativas

### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  import random  # biblioteca que permite gerar números aleatoriamente
4
5  numeroGerado=random.randint(0,20)  # gera um nº inteiro entre 0 e 20
6                                     # inclui os limites inferior e superior
7  palpite = int(input("Indique o seu palpite:"))
8  tentativas=1
9  while numeroGerado != palpite:
10     print("Não acertou! :( tente novamente!\n")
11     palpite = int(input("Indique o seu palpite:"))
12     tentativas+=1
13  print(f"Acertou em {tentativas} tentativas")
```

## ❖ Estruturas de Repetição | Iterativas

### *while*

Permite implementar um ciclo para executar um conjunto de instruções, repetidamente, enquanto se verificar uma determinada condição

```
1  # Exemplos de estruturas repetitivas / Iterativas - ciclo while
2
3  import random  # biblioteca que permite gerar números aleatoriamente
4
5  numeroGerado=random.randint(0,20)  # gera um nº inteiro entre 0 e 20
6                                     # inclui os limites inferior e superior
7  palpite = int(input("Indique o seu palpite:"))
8  tentativas=1
9  while numeroGerado != palpite:
10     print("Não acertou! :( tente novamente!\n")
11     palpite = int(input("Indique o seu palpite:"))
12     tentativas+=1
13  print(f"Acertou em {tentativas} tentativas")
```

```
Indique o seu palpite:1
Não acertou! :( tente novamente!

Indique o seu palpite:2
Não acertou! :( tente novamente!

Indique o seu palpite:3
Não acertou! :( tente novamente!

Indique o seu palpite:4
Não acertou! :( tente novamente!

Indique o seu palpite:5
Acertou em 5 tentativas
Press any key to continue . . . |
```

## ❖ Estruturas de Repetição | Iterativas

### *Quebras de ciclo*

Permitem interromper / quebrar ciclos repetitivos

#### ❑ break

Permite quebrar o ciclo e transfere a execução para a primeira instrução imediatamente a seguir ao ciclo

#### ❑ continue

Permite continuar diretamente para a próxima iteração de um ciclo sem executar as instruções seguintes da iteração corrente

## ❖ Estruturas de Repetição | Iterativas

### *Quebras de ciclo*

Permitem interromper / quebrar ciclos repetitivos

Exemplo1.py > ...

```
1  # Exemplos de quebras de ciclos
2
3  tabuada = int(input("Imprimir a tabuada dos: "))
4  numero = 0
5  while numero < 10 :
6      numero+=1
7      if numero == 6:
8          break
9      print(tabuada, "*", numero, "=", tabuada * numero)
10
11
12
```

C:\WINDOWS\py.exe

```
Imprimir a tabuada dos: 7
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35

```

## ❖ Estruturas de Repetição | Iterativas

### *Quebras de ciclo*

Permitem interromper / quebrar ciclos repetitivos

```
Exemplo1.py > ...
1  # Exemplos de quebras de ciclos
2
3  tabuada = int(input("Imprimir a tabuada dos: "))
4  numero = 0
5  while numero < 10 :
6      numero+=1
7      if numero == 6:
8          continue
9      print(tabuada, "*", numero, "=", tabuada * numero)
10
11
12
```

```
C:\WINDOWS\py.exe
Imprimir a tabuada dos: 7
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
```



## ❖ Estruturas de Repetição | Iterativas

### *Avalia o teu conhecimento*

#### ❑ for / while

Implemente um programa que leia 10 número e no final indique o maior e a média.

Nota: se indicar um número superior a 20, o seu programa deve ignorá-lo!



C:\WINDOWS\py.exe

```
Indique um número: 2
Indique um número: 4
Indique um número: 6
Indique um número: 8
Indique um número: 10
Indique um número: 1
Indique um número: 3
Indique um número: 5
Indique um número: 7
Indique um número: 9
A média é 5.5
o maior é 10
```

#### ❑ for/while

Elabore um programa que simule a função fatorial, isto é, que leia um número e determine o fatorial desse número.

Exemplo: Fatorial de 5 =  $5 * 4 * 3 * 2 * 1 = 120$

Note que  $0! = 1$

Nota: não utilizar a função `math.factorial()`

O objetivo é desenvolvermos a nossa própria função fatorial!



C:\WINDOWS\py.exe

```
Indique um número: 5
Fatorial de 5 é 120
```