P.PORTO

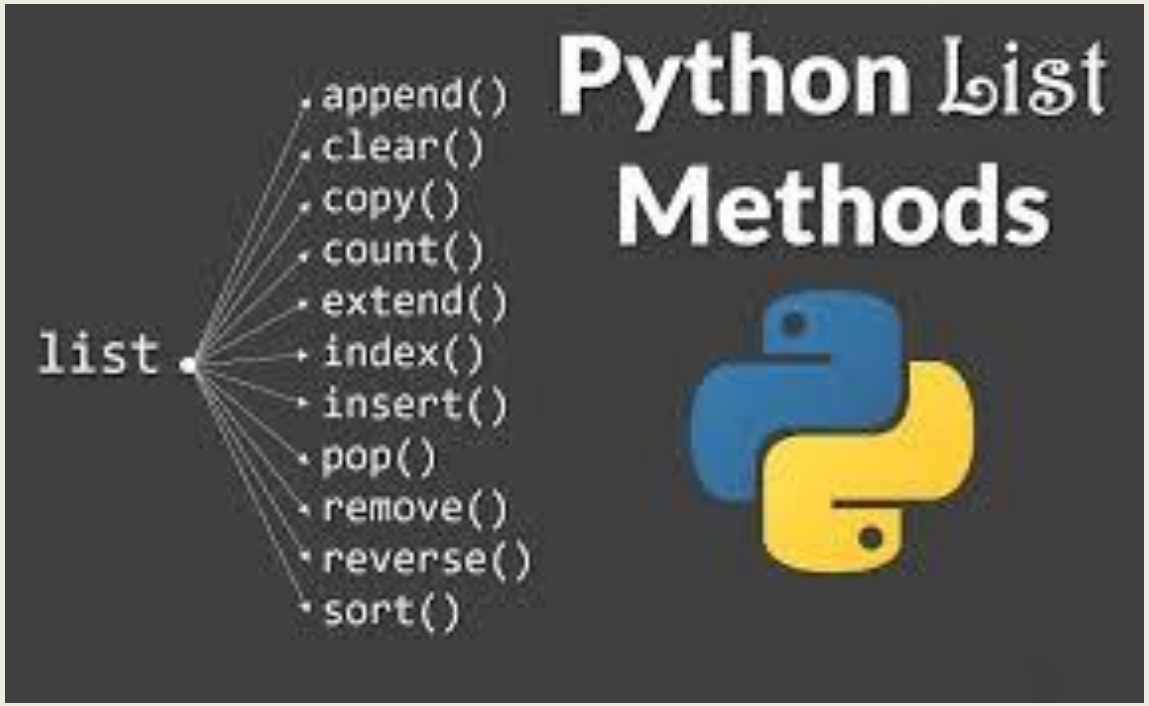
ALGORITMIA E ESTRUTURAS DE DADOS

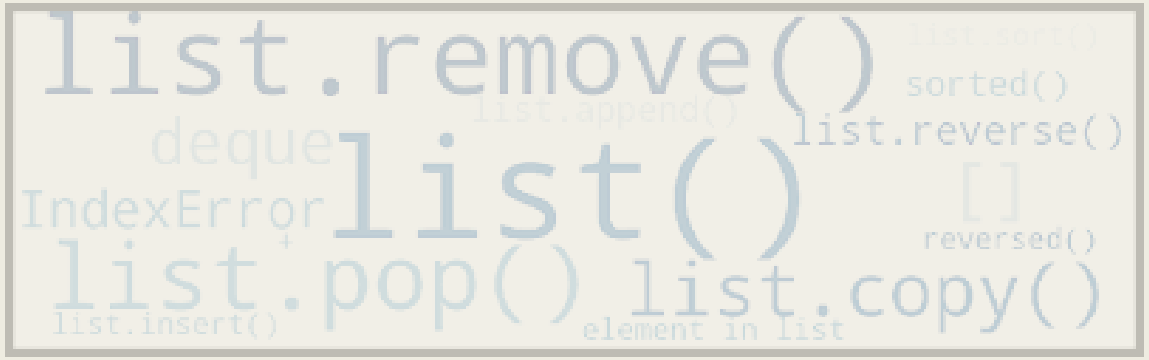
ESTRUTURAS DE DADOS NÃO PRIMITIVAS | LISTAS SIMPLES

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

1. Estruturas de dados não primitivas
   - ❑ List
   - ❑ Tuple
   - ❑ Set
   - ❑ Dicionary

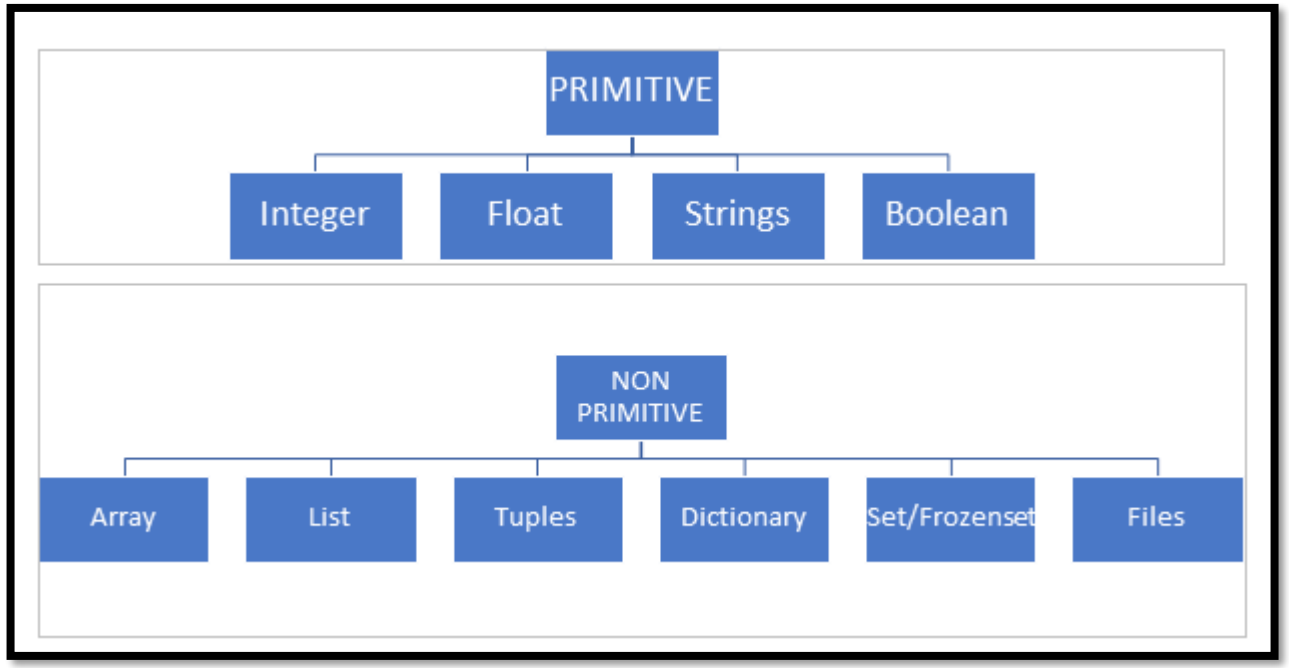
2. Arrays | Listas
   - ❑ Conceito
   - ❑ Índice de uma lista
   - ❑ Comprimento de uma lista
   - ❑ Percorrer uma lista
   - ❑ Operadores *in* e *not in*
   - ❑ Métodos *built-in Python*





❖ Coleções

- Estruturas de dados **não primitivas**, podem conter diversos dados ao mesmo tempo
- São compostas por uma conjunto de items (dados), onde cada um ocupa uma determinada posição na coleção de dados



❖ Coleções

❑ A Linguagem Python implementa 4 tipos de coleções de dados:

- **LIST**
  Coleção de dados ordenada e editável.
  Permite dados duplicados

- **TUPLE**
  Coleção de dados ordenada e não editável.
  Permite dados duplicados

- **SET**
  Coleção de dados não ordenada e não indexada.
  Não permite dados duplicados

- **DICTIONARY**
  Coleção de dados não ordenada, indexada e editável.
  Não permite dados duplicados

```python
# Lista
cidadesList = ['Porto', 'Viana', 'Matosinhos', 'Maia', 'Porto']
print('\n',cidadesList)

# Tuple
cidadesTup = ('Maia', 'Matosinhos', 'Porto', 'Viana')
print('\n',cidadesTup)

# Set
cidadesSet = {'Porto', 'Viana', 'Matosinhos', 'Maia'}
print('\n',cidadesSet)

#Dicionary
cidadeObj = {
    'nome': 'Porto',
    'pais': 'Portugal',
    'continente': 'Europa'

}
print('\n',cidadeObj)
```

```
 ['Porto', 'Viana', 'Matosinhos', 'Maia', 'Porto']

 ('Maia', 'Matosinhos', 'Porto', 'Viana')

 {'Porto', 'Maia', 'Matosinhos', 'Viana'}

 {'nome': 'Porto', 'pais': 'Portugal', 'continente': 'Europa'}
Press any key to continue . . .
```

❖ Arrays| Conceito

- ❑ Um Array é uma estrutura de dados dita completa (não primitiva) que pode armazenar uma coleção de elementos do mesmo tipo.
- ❑ O Python nativo não implementa explicitamente a estrutura Array. Implementa listas que fuincionam como Arrays.
- ❑ Se quisermos manipular arrays de forma explícita devemos usar a biblioteca numPy: https://www.w3schools.com/python/numpy/default.asp

```
list = [value1, value2, value3,...valueN]
```

❖ Listas | Conceito

- ❑ Estrutura de dados **não primitiva**, composta
- ❑ Consiste numa coleção de elementos, onde cada um ocupa uma determinada posição na estrutura de dados
- ❑ Os items de uma lista são separados por vírgula e colocados entre []
- ❑ Os items de uma lista são geralmente do mesmo tipo, mas podem ser de tipos de dados diferentes

```
list = [value1, value2, value3,...valueN]
```

❖ Listas | Conceito

- ❑ Exemplos de listas

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  print(nomes)
6  print(numeros)
7  print(dados)
```

```
['António', 'Carlos', 'Fátima', 'Raquel']
[10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
['1', 'António', '1º ano', 19]
Press any key to continue . . .
```

❖ Índice de uma lista

- Posso aceder individualmente a cada uma das posições de uma lista, através de um índice, em que a primeira posição é a 0

```
1
2  nomes = ["António", "Carlos", "Fátima", "Raquel"]
3  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
4  dados = ['1', "António", "1º ano", 19]
5
6  print(nomes[0])
7  print(nomes[1])
```

```
António
Carlos
Press any key to continue . . .
```

❖ Índice de uma lista

- ❑ Posso aceder a um subconjunto de uma lista, especificando ao posição inicial e a posição final da lista

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  print(nomes[0:2])
6  print(nomes[1:3])
7  print(nomes[:3])
8  print(nomes[1:])
9
```

```
['António', 'Carlos']
['Carlos', 'Fátima']
['António', 'Carlos', 'Fátima']
['Carlos', 'Fátima', 'Raquel']
Press any key to continue . . .
```

❖ Índice de uma lista

- ❑ Posições negativas numa lista: identificam posições a partir do final da lista
- ❑ Assim, posição -1 refere-se à última posição da lista

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  print(nomes[-1])
6  print(nomes[-2])
7  print(nomes[-3])
8  print(nomes[-4])
9
10 print('\n')
11 print(nomes[-1:])
12 print(nomes[-3:-1])
13 print(nomes[::-1])
```

```
Raquel
Fátima
Carlos
António


['Raquel']
['Carlos', 'Fátima']
['Raquel', 'Fátima', 'Carlos', 'António']
Press any key to continue . . .
```

❖ Comprimento de uma lista

- ❑ Comprimento (tamanho) de uma lista: **len**
- ❑ Função len() devolve o comprimento (nº de items) de uma lista

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5
6  print(len(numeros))
7  print(len(nomes))
```

```
10
4
Press any key to continue . . .
```

❖ Percorrer uma lista

- Percorrer lista com um ciclo for

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  for i in range(len(nomes)):
6      print(nomes[i])
```

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  for nome in nomes:
6      print(nome)
```

```
António
Carlos
Fátima
Raquel
Press any key to continue . . .
```

❖ Operadores in e not in

- Operadores in e not in devolvem True ou False

```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
dados = ['1', "António", "1º ano", 19]

nomeUser = input("Indique um nome:")
# operador in
if nomeUser in nomes:
    print("O nome já existe na lista")

# operador not in
if nomeUser not in nomes:
    print("O nome não  existe na lista")
```

```
Indique um nome:Raquel
O nome já existe na lista
Press any key to continue . . .
```

❖ Listas | Métodos *built-in python*

❑ Alguns métodos que manipulam listas:

Método	Descrição
append	Acrescenta um elemento no final da lista
insert	Insere um elemento numa determinada posição (índice) na lista
remove	Remove um determinado elemento da lista (dado)
del	Remove o elemento em determinada posição (índice)
clear	Remove todos os elementos de uma lista
pop	Remove e devolve o último elemento da lista
count	Devolve o nº de elementos com um determinado valor
index	Devolve o índice do primeiro elemento com determinado valor

https://www.w3schools.com/python/python_lists.asp

https://www.tutorialsteacher.com/python/python-list

❖ Listas | Métodos *built-in python*

❑ Alguns métodos que manipulam listas:

Método	Descrição
copy	Devolve uma cópia da lista
list	Converte uma string numa lista
sort	Ordena alista por ordem crescente
reverse	Inverte a ordenação da lista
max	Devolve o maior valor contido na lista
min	Devolve o menos valor contido na lista
sum	Devolva a soma dos elementos contidos numa lista

https://www.w3schools.com/python/python_lists.asp

https://www.tutorialsteacher.com/python/python-list

❖ Listas | Métodos *built-in python*

❑ append(*value*)

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  nomeUser = input("Indique um nome: ")
6  nomes.append(nomeUser)      # Acrescenta NO FINAL da lista
7
8  numeroUser=int(input("Indique um número: "))
9  numeros.append(numeroUser)  # Acrescenta NO FINAL da lista
10
11 print(nomes)
12 print(numeros)
```

```
Indique um nome: Ernesto
Indique um número: 27
['António', 'Carlos', 'Fátima', 'Raquel', 'Ernesto']
[10, 20, 30, 5, 30, 25, 40, 30, 12, 7, 27]
Press any key to continue . . .
```

Nota: insere um elemento **no final** da lista

❖ Listas | Métodos *built-in python*

❑ insert(*pos, value*)

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  nomeUser = input("Indique um nome: ")
6  nomes.insert(0, nomeUser)      # Acrescenta na posição indicada
7
8  numeroUser=int(input("Indique um número: "))
9  numeros.insert(1,numeroUser)  # Acrescenta na posição indicada
10
11 print(nomes)
12 print(numeros)
```

```
Indique um nome: Ernesto
Indique um número: 27
['Ernesto', 'António', 'Carlos', 'Fátima', 'Raquel']
[10, 27, 20, 30, 5, 30, 25, 40, 30, 12, 7]
Press any key to continue . . .
```

Nota: insere um elemento na lista, **na posição indicada**

❖ Listas | Métodos *built-in python*

❑ remove(*value*)

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  nomeUser = input("Indique um nome: ")
6  nomes.remove(nomeUser)      # remove um elemento da lista (pelo conteúdo)
7
8  numeroUser=int(input("Indique um número: "))
9  numeros.remove(numeroUser)  # remove um elemento da lista
10
11 print(nomes)
12 print(numeros)
```

```
Indique um nome: Carlos
Indique um número: 20
['António', 'Fátima', 'Raquel']
[10, 30, 5, 30, 25, 40, 30, 12, 7]
Press any key to continue . . .
```

Nota: remove **a primeira ocorrência de determinado valor**, no caso de existirem dados repetidos

❖ Listas | Métodos *built-in python*

❑ del(*pos*)

```
1  nomes = ["António", "Carlos", "Fátima", "Raquel"]
2  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
3  dados = ['1', "António", "1º ano", 19]
4
5  del nomes[0]
6  del numeros[9]
7
8  print(nomes)
9  print(numeros)
```

```
['Carlos', 'Fátima', 'Raquel']
[10, 20, 30, 5, 30, 25, 40, 30, 12]
Press any key to continue . . .
```

Nota: remove **determinada posição** na lista

❖ Listas | Métodos *built-in python*

❑ clear()

```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
dados = ['1', "António", "1º ano", 19]

#  Remove TODOS os elementos da lista
nomes.clear()
numeros.clear()

print(nomes)
print(numeros)
```

```
[]
[]
Press any key to continue . . .
```

Nota: remove **todos** os elementos da lista

❖ Listas | Métodos *built-in python*

❑ pop()

```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
dados = ['1', "António", "1º ano", 19]

#  Remove e devolve o último elemento da lista
nomeUser= nomes.pop()
numerosUser= numeros.pop()

print(nomeUser)
print(numerosUser)
```

```
Raquel
7
Press any key to continue . . .
```

Nota: remove e devolve o **último** elemento da lista

❖ Listas | Métodos *built-in python*

- ❑ count(*value*)

```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
dados = ['1', "António", "1º ano", 19]

nomeUser = input("Indique um nome: ")
numeroUser=int(input("Indique um número: "))

# devolve o número de ocorrências de determinado valor
print(nomes.count(nomeUser))
print(numeros.count(numeroUser))
```

```
Indique um nome: Raquel
Indique um número: 30
1
3
Press any key to continue . . .
```

Nota: devolve a o nº de ocorrências do objeto de pesquisa.
Não é possível indicar posição a partir da qual faço a contagem

❖ Listas | Métodos *built-in python*

❑ index (*value, start, end*)

```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
dados = ['1', "António", "1º ano", 19]

nomeUser = input("Indique um nome: ")
numeroUser=int(input("Indique um número: "))

# devolve a posição da primeira ocorrência
pos1 = nomes.index(nomeUser)
pos2 = numeros.index(numeroUser)

print(f'O nome {nomeUser} ocorre na posição {pos1}')
print(f'O número {numeroUser} ocorre na posição {pos2}')
```

```
Indique um nome: Carlos
Indique um número: 30
O nome Carlos ocorre na posição 1
O número 30 ocorre na posição 2
Press any key to continue . . .
```

Nota: devolve a **posição da primeira ocorrência** do objeto de pesquisa

❖ Listas | Métodos *built-in python*

❑ index (*value, start, end*)

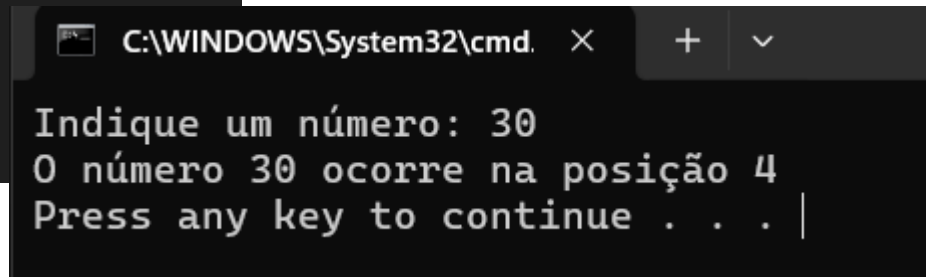
```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
dados = ['1', "António", "1º ano", 19]


numeroUser=int(input("Indique um número: "))

# devolve a posição da primeira ocorrência a partir da posição indicada
pos2 = numeros.index(numeroUser,3)

print(f'O número {numeroUser} ocorre na posição {pos2}')

```



No caso do objeto de pesquisa não existir, **devolve um erro**!
o método find() não existe nas listas!

❖ Listas | Métodos *built-in python*

- ❑ index (*value, start, end*)

```python

nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
dados = ['1', "António", "1º ano", 19]

nomeUser = input("Indique um nome: ")
numeroUser=int(input("Indique um número: "))

# devolve a posição da primeira ocorrência a partir da
pos1 = nomes.index(nomeUser,3)
pos2 = numeros.index(numeroUser,3)

print(f'O nome {nomeUser} ocorre na posição {pos1}')
print(f'O número {numeroUser} ocorre na posição {pos2}
```

```
11  # devolve a posição da primeira ocorrência a partir da posição indicada
12  pos1 = nomes.index(nomeUser,3)

Exception has occurred: ValueError ×
'Carlos' is not in list
  File "C:\Users\mario\OneDrive\AED\2023-24\4 - Exercicios\Ficha 06\Exemplo_listas.py", li
    pos1 = nomes.index(nomeUser,3)
           ^^^^^^^^^^^^^^^^^^^^^^^
ValueError: 'Carlos' is not in list

13  pos2 = numeros.index(numeroUser,3)
14
15  print(f'O nome {nomeUser} ocorre n
```

```
C:\WINDOWS\System32\cmd.
Indique um nome: Carlos
Indique um número: 30
```

No caso do objeto de pesquisa não existir, **devolve um erro**!
o método find() não existe nas listas!

❖ Listas | Métodos *built-in python*

❑ copy()

```
1
2  nomes = ["António", "Carlos", "Fátima", "Raquel"]
3  numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]
4
5  nomesBackup= nomes.copy()
6  print(nomesBackup)
7
```

```
['António', 'Carlos', 'Fátima', 'Raquel']
Press any key to continue . . .
```

Nota: método copy devolve uma lista idêntica à inicial. Não tem argumentos.

❖ Listas | Métodos *built-in python*

- ❑ list(*string*)

```
1  nome = input("Indique um nome:")
2  # Converte uma string para uma lista
3  caracteres = list(nome)
4  print(caracteres)
5
```

```
Indique um nome:Carla
['C', 'a', 'r', 'l', 'a']
Press any key to continue . . .
```

Nota: converte uma string numa lista

❖ Listas | Métodos *built-in python*

❑ sort | reverse

```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]

# Ordena a lista por ordem ascendente
nomes.sort()
numeros.sort()

print(nomes)
print(numeros)

print("\n")
# Inverte ordem da lista
numeros.reverse()
print(numeros)

# Fazer o sort e o reverse na mesma linha
nomes.sort(reverse=True)
print(nomes)

```

```
['António', 'Carlos', 'Fátima', 'Raquel']
[5, 7, 10, 12, 20, 25, 30, 30, 30, 40]


[40, 30, 30, 30, 25, 20, 12, 10, 7, 5]
['Raquel', 'Fátima', 'Carlos', 'António']
Press any key to continue . . .
```

Tip !

lista.sort(reverse=True) => ordena por ordem descendente

❖ Listas | Métodos *built-in python*

- ❑ max(*lista*) | min(*lista*) | sum(lista)

```python
nomes = ["António", "Carlos", "Fátima", "Raquel"]
numeros = [10, 20, 30, 5, 30, 25, 40, 30, 12, 7]

# função max
print(f'O maior valor é: {max(numeros)}')

# função min
print(f'O menor valor é: {min(numeros)}')

# função sum
print(f'O soma da lista  é: {sum(numeros)}')

```

```
O maior valor é: 40
O menor valor é: 5
O soma da lista  é: 209
Press any key to continue . . .
```