1. O que é o Numpy

Link do site do Numpy: https://numpy.org/.

O Numpy é uma biblioteca construída para trabalhar com o processamento de *arrays n*-dimensionais e processamento numérico. Além de facilitar indexação de *arrays*, o Numpy oferece funcionalidades como: funções matemáticas, geradores de números aleatórios, rotinas de álgebra linear, e mais.

2. Instalando o Numpy

Link para instalação: https://numpy.org/install/

Para instalar o Numpy, basta ativar o ambiente python do projeto e digitar no terminal:

```
- python -m pip install numpy
```

3. Utilizando o Numpy

In [1]:
```
import numpy as np
```

In [92]:
```
x = np.array([1, 10, 2, 3, 4])
print(f"x: {x}, tipo: {type(x)}, tipo dos dados no array: {x.dtype}")
print(f"tamanho do array: {x.shape}")
print(f"acessando o terceiro elemento do array: {x[2]}")
print(f"acessando a partir do terceiro elemento do array: {x[2:]}")
print(f"acessando os valores anteriores ao terceiro elemento do array: {x[:2]
print(f"somando 1 a cada elemento do array: {x + 1}")
print(f"dividindo cada elemento do array por 2: {x / 2}")

y = np.array([2, 3, 4, 5, 1])
print(f"somando dois arrays de mesma ordem: {x + y}")
print(f"multiplicando dois arrays de mesma ordem: {x * y}")
```

```
x: [ 1 10  2  3  4], tipo: <class 'numpy.ndarray'>, tipo dos dados no array:
int64
tamanho do array: (5,)
acessando o terceiro elemento do array: 2
acessando a partir do terceiro elemento do array: [2 3 4]
acessando os valores anteriores ao terceiro elemento do array: [ 1 10]
somando 1 a cada elemento do array: [ 2 11  3  4  5]
dividindo cada elemento do array por 2: [0.5 5.  1.  1.5 2. ]
somando dois arrays de mesma ordem: [ 3 13  6  8  5]
multiplicando dois arrays de mesma ordem: [ 2 30  8 15  4]
```

In [91]:
```
# Exemplos de funções úteis do Numpy
print(f"soma de x: {x.sum()}")
print(f"média de x: {x.mean()}")
print(f"produtório de x: {x.prod()}")
print(f"maior valor numérico de x: {x.max()}")
```

```
print(f"menor valor numérico de x: {x.min()}")
print(f"índice do maior valor numérico de x: {x.argmax()}")
print(f"índice do menor valor numérico de x: {x.argmin()}")
```

```
soma de x: 20
média de x: 4.0
produtório de x: 240
maior valor numérico de x: 10
menor valor numérico de x: 1
índice do maior valor numérico de x: 1
índice do menor valor numérico de x: 0
```

In [96]:

```
# gerando uma sequência numérica
sequencia_1 = np.arange(10)
sequencia_2 = np.arange(0, 5)
sequencia_3 = np.arange(0, 10, 2)
sequencia_4 = np.linspace(0, 10, num=5)

print(f"sequencia_1: {sequencia_1}")
print(f"sequencia_2: {sequencia_2}")
print(f"sequencia_3: {sequencia_3}")
print(f"sequencia_4: {sequencia_4}")
```

```
sequencia_1: [0 1 2 3 4 5 6 7 8 9]
sequencia_2: [0 1 2 3 4]
sequencia_3: [0 2 4 6 8]
sequencia_4: [ 0.   2.5  5.   7.5 10. ]
```

In [93]:

```
# arrays com duas dimensões
X_zeros = np.zeros(shape=(5, 5)) # criando um array de zeros
X_ones = np.ones(shape=(3, 5)) # criando um array de uns
X_random = np.random.random(size=(3, 3)) # criando um array com número aleato
I = np.eye(5) # criando uma matriz identidade

print(f"X_zeros: \n{X_zeros}, shape: {X_zeros.shape}\n")
print(f"X_ones: \n{X_ones}, shape: {X_ones.shape}\n")
print(f"X_random: \n{X_random}, shape: {X_random.shape}\n")
print(f"Identity Matrix: \n{I}, shape: {I.shape}\n")
```

```
X_zeros:
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]], shape: (5, 5)

X_ones:
[[1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]], shape: (3, 5)

X_random:
[[0.56603301 0.88416046 0.11244685]
 [0.72533344 0.12086722 0.65483365]
 [0.0038181  0.88108137 0.87553503]], shape: (3, 3)

Identity Matrix:
[[1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0.]
```

```
[0. 0. 0. 0. 1.]], shape: (5, 5)
```

In [95]:

```python
X = np.array(
    [[1, 4, 3],
     [1, 2, 3],
     [1, 5, 4]]
)

print(f"X: \n{X}\n")

# acessando uma linha de um array 2d
print(f"acessando os valores da primeira linha: {X[0, :]}")

# acessando uma coluna de uma array 2d
print(f"acessando os valores da segunda coluna: {X[:, 1]}")

# retornando o valor médio das linhas de um array 2d
# o eixo (ou dimensão) deve ser especificado
print(f"média das linhas: {X.mean(axis=0)}")

# retornando o valor médio das colunas de um array 2d
print(f"média das colunas: {X.mean(axis=1)}")

# a mesma dinâmica de acesso da dimensão funciona para as outras funções do r
print(f"soma dos valores das colunas: {X.sum(axis=1)}")
```

```
X:
[[1 4 3]
 [1 2 3]
 [1 5 4]]

acessando os valores da primeira linha: [1 4 3]
acessando os valores da segunda coluna: [4 2 5]
média das linhas: [1.         3.66666667 3.33333333]
média das colunas: [2.66666667 2.         3.33333333]
soma dos valores das colunas: [ 8  6 10]
```

4. Link para Revisar

Link: https://numpy.org/doc/stable/user/absolute_beginners.html (olhar até a parte ***How to convert a 1D array into a 2D array (how to add a new axis to an array)***).