



escola  
britânica de  
artes criativas  
& tecnologia

# Profissão: Analista de dados



# FUNDAMENTOS DE MATEMÁTICA



## GUIA DA AULA 2



# Conheça os arrays Numpy



Acompanhe aqui  
os temas que  
serão tratados  
na videoaula

- **Arrays vs Listas**
- **Arrays 1D: Vetores**
- **Arrays 2D: Matrizes**
- **Arrays 3D, 4D etc.**



## Arrays vs Listas

As listas Python e os arrays NumPy são similares em muitos sentidos: ambos servem para armazenar dados sequencialmente na memória, possuem sintaxe parecidas etc.

Contudo, é importante maximizar as suas diferenças:

- **Manipulação algébrica:** Arrays apresentam uma sintaxe mais simples e eficiente (velocidade e memória) por realizar operações de forma vetorial. Listas sempre trabalham de forma escalar. Exemplo:

```

In [ ]:  # escalar

l1 = [1, 2, 3]
l2 = [4, 5, 6]

l3 = [a + b for a, b in zip(l1,
l2)] print(l3)
  
```



```

In [ ]: # vetorial

a1 =
np.array(l1) a2
= np.array(l2)

a3 = a1 + a2
print(a3)

```

- **Tipo:** Arrays trabalham melhor com elementos do mesmo tipo.
- **Mutabilidade:** Arrays são menos eficientes quanto a inserção e remoção de elementos.



## Arrays 1D: Vetores

Arrays NumPy de uma dimensão (1D) são conhecidos como vetores, como listas de uma linha ou mais colunas.

- Criação:**

```
In [ ]: a1 = np.array([2, 4, 6,
                    8]) print(a1)
```

```
In [ ]: a1 = np.arange(0, 10,
                    2) print(a1)
```

```
In [ ]: a1 =
        np.zeros(10)
        print(a1)

        a1 = 10 *
        np.ones(10)
        print(a1)
```



- **Manipulação:**

```
In [ ]: a1 = np.array([2, 4, 6,
8]) print(a1)
```

```
In [ ]: a1[0]
```

```
In [ ]: a1[0:2]
```

```
In [ ]: a1[a1 > 4]
```



- **Atributos:**

```
In [ ]: a1.ndim
```

```
In [ ]: a1.shape
```

```
In [ ]: a1.size
```

```
In [ ]: a1.dtype
```





- **Métodos:**

```
In [ ]: a1.sort() # "inplace"  
        print(a1)
```

```
In [ ]: a1.tolist()
```



## Arrays 2D: Matrizes

Arrays NumPy de duas dimensões (2D) são conhecidos como matrizes, como tabelas, com linhas e colunas.

- Criação:**

```
In [ ]: m1 = np.array([[1, 2, 3], [4, 5, 6]]) # vetores como linhas
        print(m1)
```



- **Manipulação:**

```
In [ ]: m1[1,2] # linha x coluna
```

```
In [ ]: m1[1,:]
```

```
In [ ]: m1[:,1]
```



- **Atributos:**

```
In [ ]: m1.ndim
```

```
In [ ]: m1.shape
```

```
In [ ]: m1.size
```

```
In [ ]: m1.dtype
```



- **Métodos:**

```
In [ ]: m1.sort() # "inplace"  
        print(m1)
```

```
In [ ]: m1.tolist()
```



## Arrays 3D, 4D etc.

Arrays NumPy de três ou mais dimensões são apenas estruturas de dados com mais de duas dimensões.

