# Ciência das Redes Introdução à linguagem Python

Ricardo Luders Thiago H Silva



### Motivos para Python



- Linguagem flexível (Web, GUI, Scripting, etc.)
- Foco em legibilidade e produtividade
- Bom suporte para análise de dados

#### Partes iniciais de nivelamento



• Importante para garantir que todos absorvam melhor o conteúdo do curso.



# Sintaxe

#### Hello world



```
print("Hello world!")
```

Hello world!

### Indentação



A maioria das linguagens não se importa com identação

A maioria dos humanos sim

### Indentação



```
print ("Teste")
  print("Introdução")
```

Não está indentado

### Indentação



```
if foo:
   if bar:
      baz(foo,bar)
   else:
      qux()
```

Não precisa de { } como em C ou Java

#### Comentários



```
#Uma linha de comentário
'''Um bloco
inteiro
de comentário'''
```



# Tipos

#### Strings



```
# Esta é uma string
nome = "Thiago H Silva"

# Esta também é uma string
casa = 'Curitiba, Brasil'
```

#### Números



```
# Números inteiros
ano = 2010
ano = int("2010")

# Números de ponto flutuante
pi = 3.1415
pi = float("3.1415")
```

#### Listas



```
#Declaração de uma lista
favoritos = []
#Adicionando
favoritos.append(42)
#Extendendo
favoritos.extend(["Python", True])
# Equivalente a
favoritos = [42,"Python",True]
```

#### Listas



```
numbers = [1, 2, 3, 4, 5]
len (numbers)
# 5
numbers[0]
# 1
numbers[0:2]
# [1, 2]
numbers[2:]
# [3, 4, 5]
```

#### Dicionários



```
pessoa = {}

pessoa['nome'] = 'Thiago Silva'

pessoa.update({
    'medidas':[1.80, 90],
    'sexo': 'masculino'
})
```

#### Dicionários



```
pessoa = {'nome': 'Thiago', 'sexo':'Masculino'}
print(pessoa['nome'])
print(pessoa.keys())
print(pessoa.values())
print(pessoa.items())
Thiago
dict_keys(['nome', 'sexo'])
dict_values(['Thiago', 'Masculino'])
dict_items([('nome', 'Thiago'), ('sexo', 'Masculino')])
```

#### Dicionários



```
usuario = {}
usuario.update({'login':{'thiago':{'dependente':0}}})
print(usuario)
print(usuario['login'])
print(usuario['login']['thiago'])
print(usuario['login']['thiago']['dependente'])
{'login': {'thiago': {'dependente': 0}}}
{ 'thiago': { 'dependente': 0} }
{'dependente': 0}
```



# Operadores

#### Aritmética



```
# 10
a = 10
a += 1
             # 11
             # 10
a -= 1
             # 11
b = a + 1
             # 9
c = a - 1
          # 20
d = a * 2
e = a / 2
          # 5
             # 1
f = a % 3
g = a ** 2
               100
```

### Strings

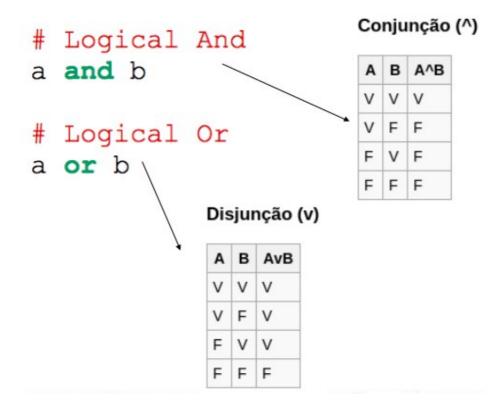


```
animais = "gato " + "cachorro "
animais += "coelho"
animais
```

'gato cachorro coelho'

## Lógicos





### Comparação aritmética



```
# Ordering
a > b
a >= b
a < b
a <= b

# Equality/Difference
a == b
a != b</pre>
```



# Controle de fluxo

#### Condicionais



```
nota = 82
if nota >=90:
    if nota ==100:
        print('A+')
    else:
        print('A')
elif nota >=80:
    print("B")
elif nota >= 70:
    print("C")
else:
    print("F")
```

#### For loops



```
for x in range(3):
    print(x)
0
```

```
frutas = ['banana', 'laranja']

for f in frutas:
    print(f)
```

banana laranja

#### For loops expandidos



```
estados = {'SP': 'Sao Paulo','MG': 'Minas Gerais'}
for chave, valor in estados.items():
    print(chave+' - '+valor)
```

SP - Sao Paulo
MG - Minas Gerais

#### While



```
c = 0
while c<4:
    print(c)
    c = c +1</pre>
```

```
0
1
2
3
```

```
c = 0
while True:
    c = c +1
    print(c)
    if c ==4:
        break
```

```
1
2
3
4
```



# Entrada e saída

#### Arquivos



```
arquivo = open('resultado.txt','w')
arquivo.write('a\n')
arquivo.write('b')
arquivo.close()
```

#### **Arquivos**



```
#with fecha o arquivo automaticamente
with open('resultado.txt', 'r') as reader:
    for line in reader.readlines():
        print(line)
```

а

b



# Funções

#### Funções



```
def imprime(t):
    print(t)

def main():
    texto = "Tenha um bom curso!"
    imprime(texto)

if __name__ == '__main__':
    main()
```

Tenha um bom curso!



```
import math
print(math.sqrt(100))
10.0
```



# Exceções

### Exceções



```
try:
    alguma coisa:
except:
    ação se erro
```



# Numpy

#### Numpy



Os arrays NumPy são mais rápidos e compactos do que as listas Python.

O NumPy usa muito menos memória para armazenar dados e fornece um mecanismo de especificação dos tipos de dados.

Isso permite que o código seja otimizado.

Um array é uma estrutura de dados central da biblioteca NumPy.

### Terminologia



Um vetor é uma matriz com uma única dimensão (não há diferença entre os vetores de linha e coluna)

Uma matriz se refere a uma matriz com duas dimensões.

Para matrizes 3-D ou dimensões superiores, o termo tensor também é comumente usado.

"ndarray" é a abreviação de "array N-dimensional". Um array N-dimensional é simplesmente um array com qualquer quantidade de dimensões.

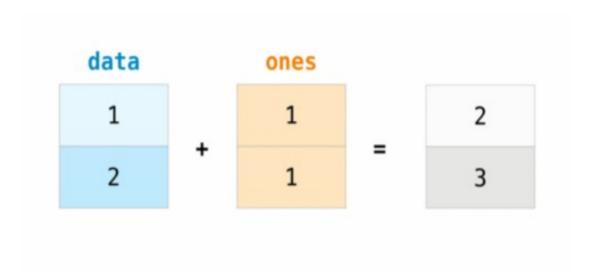
# Criando um array





# Operações básicas





# Broadcasting



1	. 1 6	_	1	*	1.6	=	1.
2	* 1.6	=	2		1.6		3.

## Mais sobre Numpy



Outros detalhes na demonstração

## Ambiente de programação



Jupyter + conda

Ou

Colab







# Ambiente de programação



Demonstração de recursos do jupyter-notebook

Mais informações nos materiais complementares

#### Tarefa



Realizar a instalação do ambiente de programação, seguindo as instruções disponíveis nos materiais complementares.

#### Referências



- https://docs.python.org/3/reference/
- Alguns dos slides foram adaptados de um material disponibilizado pelo professor Nowell Strite
- Documentação oficial da biblioteca Numpy: https://numpy.org/