

Introdução ao Python

Marco Antônio A. Fernandes
Vinicius S. Silva




O que é Python?

- Criada em 1991 por **Guido van Rossum**.
- Está atualmente na **versão 3.7**, lançada em 27 de junho de 2018.
- Foi criada com foco na **legibilidade do código**, se assemelhando ao Inglês.
- É uma linguagem de **alto nível, orientada a objetos, interpretada e dinamicamente tipada**.



Diferenças comparadas a outras linguagens

- Python utiliza “**nova linha**” para identificar o final de um comando, ao contrário de “**ponto e vírgula**” como na maioria das linguagens.
- Python utiliza **indentação** para definição escopos de programação, enquanto outras linguagens utilizam “**chaves**”.
- Python não possui os comandos “**do ... while**” e “**switch ... case**”



```
x = True
if x is True:
    print('Olá mundo!')
```

Anaconda



- É uma distribuição *open-source* das linguagens Python e R para programação científica.
- Possui ferramentas de desenvolvimento/análise inclusas, como: **Jupyter, Spyder, IPython, Orange, Glueviz, RStudio**, etc.
- Possui o gerenciador de pacotes **conda**, enquanto Python puro utiliza o gerenciador **pip**.
- Vem com uma gama de bibliotecas/pacotes já instalados e outros mais disponíveis para instalação.



O que o Python te possibilita

- **GUI:** Tkinter, PyQt (Qt Design), Kivy
- **Manipulação/Análise de Dados:** Numpy, Pandas, Scipy
- **Visualização de Dados:** Matplotlib, Plotly, ggplot, Seaborn, Dash, Bokeh
- **Visão Computacional/Processamento de Imagens:** OpenCV, SimpleCV, scikit-image, Pillow, Luminoth
- **Desenvolvimento Web:** Django, Flask
- **Inteligência Artificial:** TensorFlow, scikit-learn, Keras, PyTorch, Theano, PyBrain
- **Jogos:** PyGame, Pyglet, Pyxel



Diretrizes de Estilo da PEP 8

- **PEPs (Propostas de Aprimoramento do Python)** são documentos com diretrizes de boas práticas de programação em Python.
- **PEP 8:** documento que fornece convenções de estilo de código em Python.



Diretrizes de Estilo da PEP 8

Convenção de nomes

- **Função/Método:** palavra(s) minúscula(s) separada(s) por *underline*.
 - Ex.: `funcao`, `minha_funcao`
- **Variável:** letra/palavra(s) minúscula(s) separada(s) por *underline*.
 - Ex.: `x`, `y`, `var`, `minha_variavel`
- **Classe:** começar cada palavra com letra maiúscula. Não separar palavras por *underline*. Conhecido como ***camel case***.
 - Ex.: `Modelo`, `Pessoa`, `MinhaClasse`
- **Constante:** letra/palavra(s) maiúscula(s) separada(s) por *underline*.
 - Ex.: `X`, `Y`, `CONST`, `MINHA_CONST`



The Zen of Python

- Os “mandamentos” do Python.
- Definido na **PEP 20**



```
import this
```


Operadores Aritméticos

Operador	Exemplo
+ Adição	$a + b = 30$
- Subtração	$a - b = -10$
* Multiplicação	$a * b = 200$
/ Divisão	$b / a = 2$
% Módulo	$b \% a = 0$
** Exponenciação	$a^{**}b = 10^{20}$
// Divisão inteira (floor)	$9 // 2 = 4$

Considerando **a = 10** e **b = 20**

Operadores Comparativos

Operador	Descrição
==	igual
!=	diferente
>	maior
<	menor
>=	maior ou igual
<=	menor ou igual

Operadores Lógicos

Operador	Descrição
and	e
or	ou
not	negação

Operadores Afiliativos

Operador	Descrição
in	x in y , verifica se x está em y
not in	x not in y , verifica se x não está em y

Considerando **y** uma sequência.
Ex.: string, lista, tupla, dicionário,
etc.

Operadores de Identidade

Operador	Descrição
is	x is y , verifica se x também é y
is not	x is not y , verifica se x não é y

Compara dois objetos pelo endereço de memória.

Tipos de dados

- Variáveis são criadas por **atribuição** e não por **declaração**.



Tipo inteiro (int)

x = 10 # decimal

x = 0b1010 # binário

x = 0o12 # octal

x = 0xA # hexadecimal

Tipos de dados



Tipo real (float)

x = 15.3

x = 9. # 9.0

x = .5 # 0.5

x = 6.4e2 # 640.0



Tipo complexo (complex)

x = 2+3j

x = -5+1j

Tipos de dados



Tipo string (str)

```
x = 'Olá mundo!'
```

```
x = "Olá mundo!"
```



Concatenando/formatando string

```
x = 'Python é ' + str(10)
```

```
x = 'Python é {}'.format(10)
```

```
x = 'Python é %i' % (10)
```


Tipos de dados



Tipo lista (list)

```
x = list()
```

```
x = []
```

```
x.append(1)
```

```
x.append(2)
```

```
x.append(3)
```

```
x = [1, 2, 3]
```

```
x = [5, 'sete', 9]
```

```
x = [10.5, [8,9]]
```

Tipos de dados



Tipo tupla (tuple)

```
x = tuple()
```

```
x = ()
```

```
x = (1, 2, 3)
```

```
x = ('uva', [1,2,3], (5, 4))
```

```
x = (1, 2) + (3, 4)
```

Tipos de dados



Tipo set (set)

```
x = set([1, 2, 3])
```

```
x = {1, 2, 3}
```

```
x = {1, 2, 3, 1, 2}
```

```
x = {'laranja', 'uva', 'pera', 'uva', 10}
```

operações com sets

```
x = {1, 2} | {2, 3, 4} # união
```

```
x = {1, 2} & {2, 3, 4} # interseção
```

```
x = {1, 2} - {2, 3, 4} # diferença
```

```
x = {1, 2} ^ {2, 3, 4} # diferença simétrica
```

Tipos de dados



Tipo dicionário (dict)

```
x = dict( )
```

```
x = {}
```

```
x = {'chave': 'valor'}
```

```
x = dict(chave='valor')
```

```
x = {'a': 1, 'b': 2, 'c': 3}
```

```
x['d'] = 4
```

```
x.update(d=4)
```

```
x.update({'d':4})
```

Entrada e Saída de dados



```
x = input()

x = input('X= ')

x = int(input('X= '))

x, y, z = input().split()

x, y, z = map(int, input().split())

nomes = list(input.split())
```



```
x = 10
print('Python é {:d}'.format(x))
```

Condicional SE SENÃO



```
x = 10; y = 20
```

```
if x < y:  
    print("'x' é menor que 'y'")  
else:  
    print("'x' não é menor que 'y'")
```

Condicional SENÃO SE



```
x = 10; y = 20
```

```
if x < y:  
    print("'x' é menor que 'y'")  
elif x > y:  
    print("'x' não é menor que 'y'")
```

Operador ternário



```
a = 10
```

```
b = 0
```

```
x = a / b if b != 0 else 'Divisão por zero!'
```

```
# b != 0 ? a / b : 'Divisão por zero!'
```


Laço ENQUANTO



```
i = 0
while i < 10:
    print(i)
    i += 1
```



```
i = 0
while True:
    print(i)
    i += 1
    if i >= 10:
        break
```

Laço PARA



```
for i in range(10):  
    print(i)  
  
for i in range(10, 21):  
    print(i)  
  
for i in range(0, 36, 5):  
    print(i)  
  
for i in range(-1, -11, -1):  
    print(i)
```



```
cods = (5, 3, 10)  
for cod in cods:  
    print(cod)  
  
nomes = ['Maria', 'João', 'Ana']  
for nome in nomes:  
    print(nome)  
  
dicio = dict(zip(cods, nomes))  
for cod in dicio:  
    print(cod)  
  
for cod, nome in dicio.items():  
    print(cod, nome)
```

Função



```
def minha_funcao():  
    print('Olá mundo!')  
  
def soma(x, y):  
    return x + y  
  
def soma(x=0, y=0):  
    return x + y  
  
def soma_subtracao(x=0, y=0):  
    return x+y, x-y
```



```
soma_subtracao(15, 5)      # (20, 10)  
soma_subtracao(y=15, x=5) # (20, -10)
```



```
def soma(x=0, y=0, *args):  
    total = x + y  
    for n in args:  
        total += int(n)  
    return total  
  
def preco(valor, **kwargs):  
    if 'desconto' in kwargs:  
        valor -= kwargs['desconto']  
    return valor
```

Função Lambda (Anônima)



```
quadrado = lambda a : a**2
```

```
soma = lambda a, b : a + b
```

```
quadrado(4)
```

```
soma(2, 3)
```

Importação



```
import math
import math as m
from math import *

from math import cos, sin
from math import pi as PI
```



```
import numpy as np
import matplotlib.pyplot as plt
# %matplotlib inline

x = np.linspace(0, 2*np.pi, 100)
y = np.sin(x)
plt.plot(x, y)
plt.show()
```

Orientação a objetos



```
class Carro:
    def __init__(self, modelo, ano, cor):
        self.modelo = modelo
        self.ano = ano
        self.cor = cor
        self.ligado = False

    def ligar(self):
        self.ligado = True

    def desligar(self):
        self.ligado = False
```



```
carro = Carro('Uno', 2001, 'preto')
carro = Carro(modelo='Uno',
               ano=2001, cor='preto')

carro.ligar()
print(carro.ligado)
carro.desligar()

del carro
```

Agora vamos praticar!

- <http://www.urionlinejudge.com.br>
- 1005 - Média 1
- 1045 - Tipos de Triângulos
- 1069 - Diamantes e Areia
- 1164 - Número Perfeito
- 1168 - LED
- 1169 - Trigo no Tabuleiro
- 1828 - Bazinga!