



pythonTM

Atuação, documentação, padrões e boas práticas

..

Bonito é melhor que feio

Explícito é melhor que implícito

Simple é melhor que complexo

Complexo é melhor que complicado

Plano é melhor que aglomerado

Esparsa é melhor que densa

Legibilidade faz diferença

*Casos especiais não são especiais o bastante
para quebrar as regras*

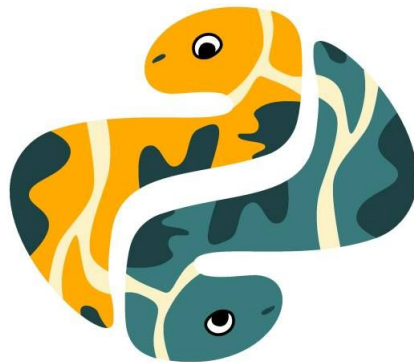
(...)

..

The Zen of Python, de Tim Peters

```
Terminal Linux:
```

```
$ python3  
>>> import this
```



Atuação

- Desenvolvimento Web
- Ciência de Dados
- Inteligência Artificial



Flask

django

 **pandas**

 **PyTorch**

Python e suas características

- Multiparadigmas:
 - Imperativa;
 - Orientada à objeto;
 - Funcional;
 - Tipagem dinâmica;
- Código Interpretado:
 - Códigos Python são executados a cada comando;





Hello World

Compilador online e Linux

Documentação Python

python.org

- <https://docs.python.org/3/whatsnew/3.10.html>

Últimas releases:

- 3.8;
- 3.9;
- 3.10;



Python Enhancement Proposal 8 (PEP8)

Funções e Métodos:

- Snake Case
- Uso de verbos



Ex: `calcular_no_python`

Pacotes e nome de módulos:

- Lower Case

Ex: `package`



Classes:

- Camel Case



Ex: `ClassePython`

Variáveis:

- Snake Case
- Evitar "l", "O" ou "I"



Ex: `variaveis_python`

Python Enhancement Proposal 8 (PEP8)

- **Indentação:**
 - Quatro espaços

- **Espaços em branco:**
 - Separar operadores matemáticos, binários e de comparação

Ex: `x = 3 ** 2`

Parâmetros:

- Snake Case
- Parâmetros referenciados para a própria classe devem se chamar "self";

Ex:

```
def calcular_no_python( self,  
    parametro_python):
```



Boas Práticas

- Python Enhancement Proposal 8 (PEP)
 - Indentação
 - Funções e Classes: duas linhas de separação
 - Dentro de Classes: separar funções com uma linha
 - Importação de Bibliotecas: Importantes → linhas específicas
 - Importação de Bibliotecas: Apenas o que será usado
 - Finalização lógica: uma linha de separação
 - Finalização de algoritmo: sempre com uma linha em branco
 - Caracteres por linha: 79 caracteres



1. Indentação

```
if "a" in "banana":  
    print("tem")
```



2. Funções e Classes: duas linhas de separação

```
class ClasseUm:  
    pass  
# quebra de linha  
# quebra de linha  
class ClasseDois:  
    pass  
# quebra de linha  
# quebra de linha
```



3. Dentro de Classes: separar métodos com uma linha


```
class ClasseUm:  
    def fazer_metodo_um(self):  
        pass  
# quebra de linha  
    def fazer_metodo_dois(self):  
        pass  
# quebra de linha
```



4. Importação de Bibliotecas

```
import sys  
import os
```

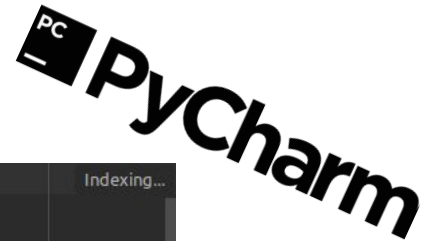
```
from Types import StringTypes, ListTypes
```



5. Finalização de código: uma linha em branco

```
...  
# fim do código  
# quebra de linha
```

6. Caracteres: 79 por linha



```
1 """
2 Read grade_one(float) and grade_two(float) - the grades must be between 0 and 10
3 Read weights(list) with 3.5 and 7.5 values
4 Average equation = (grade1 * weight1) + (grade2 * weight2) / weight1 + weight2
5 Print Average
6 """
7 grade_one = -1
8 grade_two = -1
9 weights = [3.5, 7.5]
10
11 while grade_one < 0 or grade_one > 10:
12     grade_one = float(input("GRADE 1: "))
13
14 while grade_two < 0 or grade_two > 10:
15     grade_two = float(input("GRADE 2: "))
16
17 average = ((grade_one * weights[0]) + (grade_two * weights[1])) / (weights[0] + weights[1])
18
19 print(f"GRADE 1 = {grade_one} \
20       WEIGHT 1 = {weights[0]} | \
21       GRADE 2 = {grade_two} \
22       WEIGHT 2 = {weights[1]} | \
23       AVERAGE = {average}")
```

Referências Bibliográficas

Pedro Marcolino Rampazo
(SP3100065)
Rafael Rodrigues de Sousa
(SP3100472)

- The Zen Of Python (<https://wiki.python.org.br/TheZenOfPython>)
- Framework Python Ciência de Dados
(<https://dojo.bylearn.com.br/python/ciencias-de-dados-com-python/>)
- Framework Python Inteligência Artificial
(<https://dojo.bylearn.com.br/python/6-frameworks-python-para-2020/>)
- Python Interpretado
(<https://engenhariade.software/questions/614/o-python-e-interpretado-ou-compila-do>)
- Documentação Python (<https://docs.python.org/3/whatsnew/3.10.html>)
- Nomenclatura Python
(<https://pt.stackoverflow.com/questions/8613/convenções-de-nomes-para-variáveis-e-funções-no-python>)
- Boas práticas (<https://dev.to/lucianorangelaguilar/boas-praticas-de-pep-8-5ccl>)