



# Curso de Python

25-27/09-2019

<https://github.com/tsbressan/CursoPythonUnisinos>

# Introdução a Linguagem Python

- Criada no final dos anos 80 (1989-1990) por Guido Van Rossum, no Centro de Matemática e Tecnológica da Informática, na Holanda. (baseada em linguagem ABC, Algol, C, Java, Perl, entre outras)
- Nome da linguagem relaciona-se com o gosto do criador por humoristas britânicos (Monty Python) ou a relação com o réptil com o mesmo nome (em português-píton)
- Linguagem de programação muito utilizada para análise de dados (científico), programação web, mobile, games,...

# Introdução a Linguagem Python

- **CARACTERÍSTICAS:**
- Base no Sistema Operacional escrito em C (principal CPython)
- Linguagem interpretada e de alto nível;      Interoperabilidade;
- Orientada a objetos;      Multiplataforma;
- Tipagem dinâmica e forte;
- Filosofia de poucas linhas de código em relação a outras linguagens 'tradicionais' – programação rápida e legível
- 5ª linguagem mais popular (considerada uma linguagem simples)

# Introdução a Linguagem Python

- **Versões e software:**

- Atualmente é gerenciada pela Python Software Foundation, com licença de código aberto e livre, compatível com a GNU (General Public License).

- Site: <https://www.python.org/> ou <https://www.python.org.br>

- Versão Atual: 3.7.4

- Anaconda: <https://anaconda.org/>

Pacote completo de ferramenta da análise de dados (instala todos os softwares e bibliotecas necessárias, Editor de texto,...)

# Introdução a Linguagem Python

- **Instalação:**
- Pode-se instalar somente o Python ou o pacote completo (Anaconda) no Windows, Linux, MAC OS,..
- X86-64bits (Windows x86-64 executable installer.exe) ou para o sistema operacional específico (em formato específico)  
<https://python.org.br/instalacao-windows/>
- Anaconda3-2019.07-Windows-x86\_64.exe
- [https://repo.anaconda.com/archive/Anaconda3-2019.07-MacOSX-x86\\_64.pkg](https://repo.anaconda.com/archive/Anaconda3-2019.07-MacOSX-x86_64.pkg)

# Introdução a Linguagem Python

- **Editores:**
  - São basicamente dois editores instalados no Anaconda: Jupyter Notebook e Spyder.
  - Pode-se ser utilizado outros editores simples (como o próprio Bloco de Notas do Windows) ou instalar/utilizar outros como Pycharm, Notepad++, Gedit, Sublime,... (<https://python.org.br/ferramentas/> )
- **Execução do código:**
  - Pode-se utilizar o próprio editor *Jupyter* ou *Spyder* (já vincula com o Python e roda o código)
  - Executar via linha de comando (prompt de comando ou shell)

# Introdução a Linguagem Python

- **Tipo de arquivo da linguagem:**
- Os arquivos de código devem ser salvos com o tipo \*.py
- Nota: O editor Jupyter para executar os códigos diretamente no editor utiliza outro tipo de arquivo (\*.ipynb).

# Introdução a Linguagem Python

- **Instalação de pacotes (módulos ou bibliotecas) adicionais:**

Duas formas (linha de comando):

Utilizar o **pip** (diretamente no python, mais utilizado no Linux):

`pip install <nome_pacote>`, `pip uninstall <nome_pacote>`, `pip list`

<https://pip.pypa.io/en/stable/quickstart/>

Utilizar o **conda** (diretamente no anaconda):

`conda install -n <nome_pacote>`, `conda remove -n <nome_pacote>`,  
`conda list -n`, `conda info`

<https://docs.conda.io/projects/conda/en/latest/commands/remove.html>

Os dois formatos funcionam e são compatíveis entre si ( no mesmo computador)



# Introdução a Linguagem Python

- **Instalação de pacotes (módulos ou bibliotecas) adicionais**
- Pode ser utilizado pelo gerenciador (aplicativo) do Anaconda:
- Environments (ambientes): Ambiente para desenvolvimento das aplicações (códigos). Cada ambiente trabalha independente (separa os códigos e bibliotecas). base(root) é o principal
- Create, Clone, Import e Remove Environments
- Para o Clone e Import, o tipo de arquivo é \*.yml



Home

Environments

Learning

Community

Documentation

Developer Blog

Applications on

base (root)

Channels

Refresh



JupyterLab

1.0.2

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Launch



Notebook

6.0.0

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Launch



Spyder

3.3.6

Scientific Python Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Launch



Glueviz

0.13.3

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Install



Orange 3

3.19.0

Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

Install



RStudio

1.1.456

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Install



VS Code

1.37.1

Streamlined code editor with support for development operations like debugging, task running and version control.

Install



Home

Environments

Learning

Community

Documentation

Developer Blog



Search Environments

base (root)

tf-gpu

Installed

Channels

Update index...

Search Packages

Name	Description
✓ _libgcc_mutex	
✓ _tflow_select	
✓ absl-py	Abseil python common libraries, see https://github.com/abseil/abseil
✓ astor	Read, rewrite, and write python asts nicely
✓ attrs	Attrs is the python package that will bring back the joy of writing
✓ backcall	Specifications for callback functions passed in to an api
✓ blas	
✓ bleach	Easy, whitelist-based html sanitizing tool
✓ ca-certificates	Certificates for use with other packages.
✓ certifi	Python package for providing mozilla's ca bundle.
✓ cloudpickle	Extended pickling support for python objects
✓ colorama	Cross-platform colored terminal text.
✓ cudatoolkit	
✓ cudnn	Nvidia's cudnn deep neural network acceleration library
✓ cycler	Composable style cycles.
✓ cytoolz	Cython implementation of toolz. high performance functional utilities.
✓ dask-core	Parallel python with task scheduling
✓ decorator	Better living through python with decorators.
✓ defusedxml	Xml bomb protection for python stdlib modules

154 packages available

Pesquisar pacotes

Installed



Installed

Not installed

Updatable

Selected

All



Create



Clone



Import



Remove

# Introdução a Linguagem Python

- Instalação de pacotes (módulos ou bibliotecas) adicionais

- Pode ser instalado o pacote diretamente do github:

```
pip install git+<endereço_do_site>
```

```
pip install -r requirements.txt (instalar as dependências do pacote em questão) (para criar dentro do ambiente: pip freeze > requirements.txt)
```

- Exemplo:

```
pip install  
git+https://github.com/ceddlyburge/python_world#egg=python_world  
pip install git+https://github.com/fact-project/smart_fact_crawler
```

# Pacotes (Módulo ou Bibliotecas)

(**import**) Importam para o código bibliotecas ou módulos extras instalados no Python (ou Anaconda), como suporte para execução de tarefas específicas (não suportadas pela biblioteca padrão do Python).

Exemplos:

```
import numpy
```

```
import pandas
```

```
import matplotlib
```

```
import random
```

Após importar as bibliotecas é possível utilizar as funções vinculadas a cada biblioteca.

# Estrutura da Linguagem Python – Sintaxe Inicial

- **Comandos básicos:**

= -> atribuição

print () -> mostrar alguma informação na tela

# -> comentário de uma linha no código

""" .... """ ou ''' .... ''' -> comentário de várias linhas no código

## Operadores matemáticos:

+, -, \*, /, % (resto da divisão), \*\* (potência)

## Funções matemáticas:

max( ) -> máximo

min( ) -> mínimo      abs( ) -> retorna valor absoluto

# Estrutura da Linguagem Python – Sintaxe Inicial

- **Comandos básicos:**

Operações com strings

+ ->(concatenar),    \'s    ->(Escape para \')

## Funções com strings

Len( ) -> tamanho da string

Lower ( ) ->minúsculo

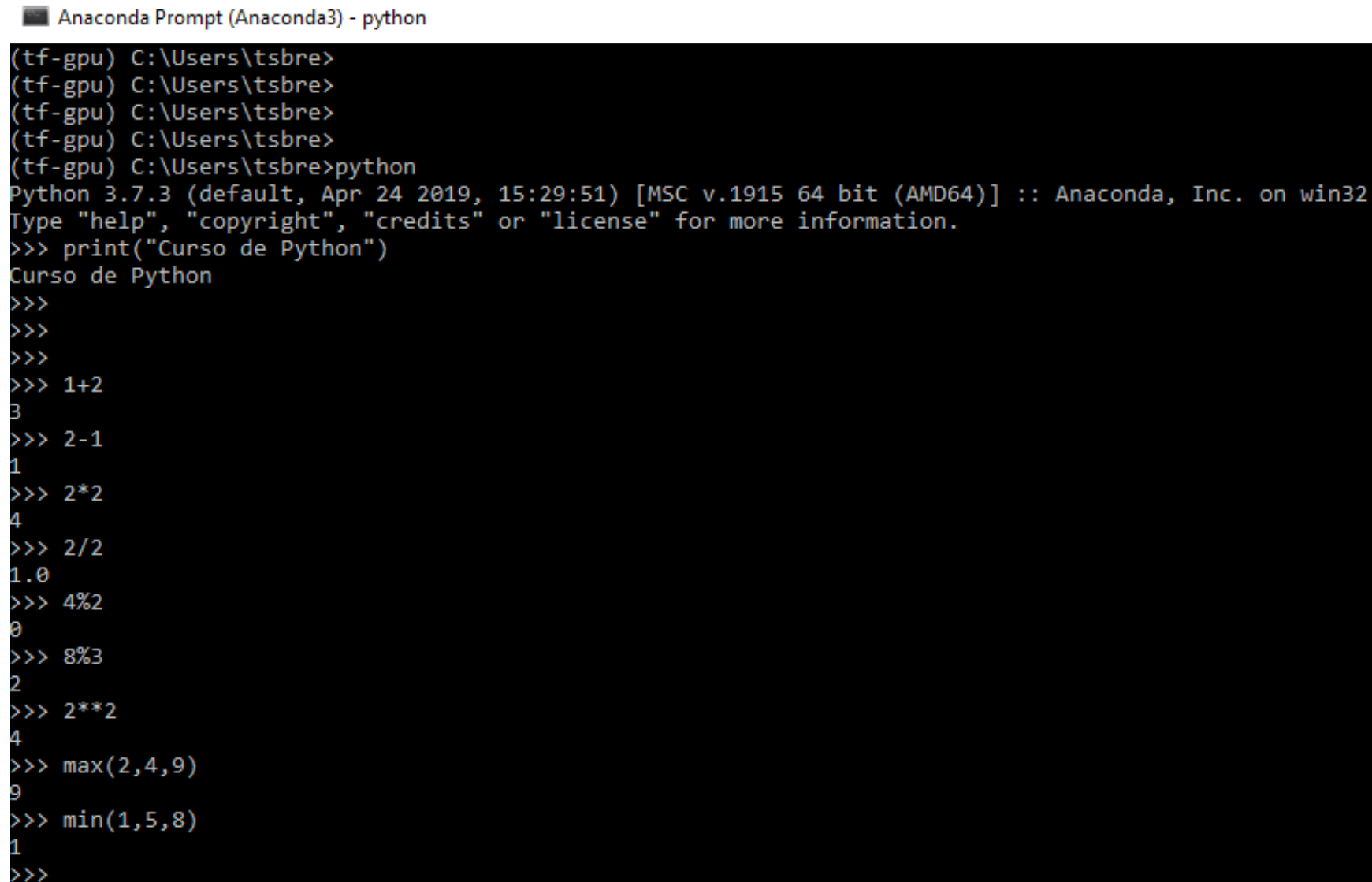
Upper ( ) -> maiúsculo

Str( ) -> converter em string

Isalpha() -> retorna *false* se a string contiver caracter que não seja letra.

# Estrutura da Linguagem Python – Sintaxe Inicial

- **EXEMPLOS:**

A screenshot of an Anaconda Prompt window titled "Anaconda Prompt (Anaconda3) - python". The window shows a series of commands and their outputs in a dark-themed terminal. The commands include directory navigation, running the Python interpreter, and various Python expressions for printing, arithmetic, and built-in functions. The outputs are displayed on the lines following the commands.

```
(tf-gpu) C:\Users\tsbre>
(tf-gpu) C:\Users\tsbre>
(tf-gpu) C:\Users\tsbre>
(tf-gpu) C:\Users\tsbre>
(tf-gpu) C:\Users\tsbre>python
Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Curso de Python")
Curso de Python
>>>
>>>
>>>
>>> 1+2
3
>>> 2-1
1
>>> 2*2
4
>>> 2/2
1.0
>>> 4%2
0
>>> 8%3
2
>>> 2**2
4
>>> max(2,4,9)
9
>>> min(1,5,8)
1
>>>
```



# Estrutura da Linguagem Python – Sintaxe Inicial

- **EXEMPLOS:**

```
>>>
>>> print ("Curso" + "de" + "Python")
CursodePython
>>> print ("Caixa d'agua")
Caixa d'agua
>>> print ('Caixa d'agua')
File "<stdin>", line 1
    print ('Caixa d'agua')
              ^
SyntaxError: invalid syntax
>>> print ('Caixa d\'agua')
Caixa d'agua
```

```
>>> var = "CURSO DE PYTHON"
>>> var.lower()
'curso de python'
>>>
>>> var = "curso de python"
>>> var.upper()
'CURSO DE PYTHON'
>>>
>>>
>>> len ("Curso de Python")
15
```

# Estrutura da Linguagem Python – Sintaxe Inicial

- **EXEMPLOS:**

```
>>> print (2,5)
2 5
>>> print (2)
2
>>> print ("Data: ", 02/04/2019)
File "<stdin>", line 1
    print ("Data: ", 02/04/2019)
                        ^
SyntaxError: invalid token
>>> print ("Data: ", "02/04/2019")
Data:  02/04/2019
>>>
>>> print ("Temperatura externa: "+ str(25))
Temperatura externa: 25
>>>
>>> var = "Curso de Python"
>>> var.isalpha()
False
>>>
>>> var = "Curso de Python 123"
>>> var.isalpha()
False
>>> var = "Curso"
>>> var.isalpha()
True
>>>
```

# Estrutura da Linguagem Python – Sintaxe Inicial

- **EXEMPLOS:**

- Digitar os códigos diretamente na tela preta??
- Podemos armazenar os códigos em um arquivo \*.py

-- abrir o arquivo de código: *codigo1.py* utilizando o Bloco de notas

-- cuidar a linha #coding: utf-8 (codificação)

Executar:

```
>> python codigo1.py
```

# Estrutura da Linguagem Python – Sintaxe Inicial

- **VARIÁVEIS:**

- Utilizada para armazenar alguma “informação” conforme um tipo de dado
- Deve ser inicializada/criada antes de ser utilizada.
- Não existe criação automática de variáveis em Python

- Por exemplo:

```
>>> soma = numero1 + numero2
```

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

NameError: name 'numero1' is not defined

# Estrutura da Linguagem Python – Sintaxe Inicial

- **VARIÁVEIS:**

## Tipo de dados em Python

- **inteiro:**

a = 123     #decimal     a = 017   #Octal inicia em zero

a = 0xAF   #Hexadecimal inicia em 0x

- **Float:**

a = 0.024

- **Long:** #números inteiros longos

a = 145897896254

# Estrutura da Linguagem Python – Sintaxe Inicial

- **VARIÁVEIS:**

## Tipo de dados em Python

- **bool:** #valores booleanos True ou False (Operadores lógicos, and-or)

a=True

b=False

- **None type:** #tipo None, ausência de valores, simular a null

a=None

- **String(str):** #entre aspas simples, duplas ou triplas.

a="Curso"

# Estrutura da Linguagem Python – Sintaxe Inicial

- **VARIÁVEIS:**

## **EXEMPLOS PRÁTICOS(1):**

a=2.25 #tipo float

b=55 #tipo inteiro

c=0740 #tipo inteiro octal

e=0xFFAB #tipo inteiro hexadecimal

f="Curso de Python" #tipo string

type(a) #mostra o tipo da variável

# Estrutura da Linguagem Python – Sintaxe Inicial

- **VARIÁVEIS:**

- **EXEMPLOS PRÁTICOS(2):**

```
meu_nome = "Carlos"
```

```
meu_sobrenome = 'Santini'
```

```
print ("Nome: %s, Sobrenome: %s" % (meu_nome.upper(), meu_sobrenome))
```

```
print (f'Nome: {meu_nome.upper()}, Sobrenome: {meu_sobrenome}') #Formatted string literals ou f-string
```

```
print("Nome: {0}, Sobrenome: {1}".format(meu_nome.upper(), meu_sobrenome)) #new-style string formatting
```

```
print ("Meu nome começa com a letra ", meu_nome[0])
```

```
print ("Meu nome começa com a letra ", meu_nome[0].lower())
```

```
print ("Meu primeiro nome é ", meu_nome[0:6])
```

```
print ("Meu sobrenome é ", meu_sobrenome[0:7])
```



# Estrutura da Linguagem Python – Sintaxe Inicial

- **VARIÁVEIS:**

## **Conversão de tipos em python**

```
a = float(21/4)
```

```
b = int(4.8)
```

```
c = int(4.9)
```

```
d = int(0xff500)
```

```
e = float(int(3.9))
```

```
f = int(float(3.9))
```

```
g = int(float(3))
```

```
h = round(3.9)
```

```
i = round(3)
```

```
j = int(round(3.9))
```

```
print (a,b,c,d,e,f,g,h,i,j)
```

# Estrutura da Linguagem Python – Sintaxe Inicial

- **VARIÁVEIS:**

Palavras reservadas – não utilizar como uma variável:

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

# Editores

- Jupyter Notebook
  - Spyder
- 
- Dois editores instalados pelo Anaconda.

# Editores

- **Jupyter Notebook**

Funcionamento:

- Clicar no ícone: Jupyter Notebook no menu do Anaconda ou via linha de comando: `>> jupyter notebook`
- É Executado por um navegador (como um servidor web local).

<http://localhost:8888>

- Sua execução é interativa, passo a passo (pode ser inserido textos, figuras entre os códigos)

Files

Running

Clusters

Select items to perform actions on them.

Upload

New ▼



0



Name ▼

Last Modified

File size


 codigo1.py

7 dias atrás

810 B


 Dia1.pptx

4 minutos atrás

7.95 MB


 {ECB14B49-2BC7-4AEA-B645-CABBEAFEB2AB}.tmp

21 horas atrás

7.94 MB

Files = Lista dos arquivos no diretório principal

New = novo arquivo Python3

Running = arquivo em aberto (em funcionamento)

# Editores

- **Spyder**

Funcionamento:

- Clicar no ícone: Spyder no menu do Anaconda
- Editor mais tradicional, interface completo com console para executar os códigos (a direita)

Spyder (Python 3.7)

Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda

Editor - C:\Users\tsbre\OneDrive - Associacao Antonio Vieira\Doutorado\Unisinos\2017-2021\eventos 2019\Curso de Python Unisinos 2019\Dia 1\codigo1.py

```
1#coding: utf-8
2
3
4#início do código
5operações matemáticas
6print (1+2)
7print (2-1)
8print (2/2)
9print (4%2)
10print (8%3)
11print (2**2)
12#funções matemáticas
13print (max(2,4,9))
14print (min(1,5,8))
15print (abs(-22))
16
17#operações com string
18
19#concatenar
20print ("Curso" + "de" + "Python")
21
22
23print ("Caixa d'agua") #correto
24
25#print ('Caixa d'agua') #errado - cuidar as aspas simples
26print ('Caixa d\'agua') #correto
27
28#funções com strings
29print (len ("Curso de Python"))
30
31var = "CURSO DE PYTHON"
32print (var.lower())
33
34var = "curso de python"
35print (var.upper())
36
37
38print (2,5)
39
40print (2)
41
42#print ("Data: ", 02/04/2019) #errado
   print ("Data: ", "02/04/2019") #certo
```

Uso

Neste painel é possível obter a ajuda de qualquer objeto ao pressionar **Ctrl+I** estando na frente do mesmo, tanto no Editor quanto no Console.

Essa ajuda também pode ser mostrada automaticamente depois de escrever um parêntese junto a um objeto. Você pode ativar este comportamento em *Preferências > Ajuda*.

Explorador de variáveis Explorador de arquivos Ajuda

Console IPython

Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)]  
Type "copyright", "credits" or "license()" for more information.

IPython 7.6.1 -- An enhanced Interactive Python.

```
In [1]: runfile('C:/Users/tsbre/OneDrive - Associacao Antonio Vieira/Doutorado/
Unisinos/2017-2021/eventos 2019/Curso de Python Unisinos 2019/Dia 1/
codigo1.py', wdir='C:/Users/tsbre/OneDrive - Associacao Antonio Vieira/
Doutorado/Unisinos/2017-2021/eventos 2019/Curso de Python Unisinos 2019/Dia 1')
3
1
1.0
0
2
4
9
1
22
CursodePython
Caixa d'agua
Caixa d'agua
15
curso de python
CURSO DE PYTHON
```

Permissões: RW Fim de linha: CRLF Codificação: LATIN-1-GUESSED Linha: 11 Coluna: 13 Memória: 44 %

09:36 10/09/2019

# Editores


- **Jupyter Notebook x Spyder**

Sugestão de uso:

- Para executar arquivos \*.py diretamente no cmd (prompt de comando) utilize o editor Spyder
- Para executar arquivos passo a passo (linha por linha) com comentários utilize o Jupyter Notebook. Arquivos criados diretamente no Jupyter possuem a extensão \*.ipynb
- O bloco de notas pode ser utilizado como editor, porém não possui as cores dos códigos e todos os recursos de um editor para Python



# Código estruturado com função

```
def main():      #função com o nome main()
    a = 3
     b = 4
    soma = a + b
    print("A soma de a + b e igual a: ", soma)
```

```
main()      # executa a função
```

- Cuidar a indentação;
- Revisar aspas, parênteses, pontos, vírgulas,..., programação é revisar o texto várias e várias vezes

# Entrada de dados – input()

- **Responsável por receber os dados que o usuário fornece via teclado.**

Formato:

```
variável = input ("prompt")
```

Exemplo:

```
nome = input ("Qual é o seu nome? ")
```

```
print ("O seu nome é:", nome)
```

# Entrada de dados – input()

```
def main():
```

```
    a = input("Digite o primeiro numero: ")
```

```
    b = input("Digite o segundo numero: ")
```

```
    soma = a + b
```

```
    print("A soma de", a, "+", b, "e igual a", soma)
```

```
main()
```

# Exemplos práticos:

- **1) Operações matemáticas com 2 variáveis**
- **2) Operações matemáticas com 3 variáveis**
- **3) Operações com String**
- **...**
- **4) Conversão de tipos**
- **5) Operações com booleanos**
- **6) Operação com String - avançado**