



Curso de Introdução à Programação em Python

Autor: Alexandre Cardoso Garcia Leite

email: alexandrecgleite@gmail.com

Repositório GitHub: github.com/aleitebr



Introdução

Python é simples de aprender e é elegante, contudo, é uma linguagem poderosa, que roda na maioria das plataformas, com poderosas estruturas de dados já prontas para uso. É aberta e possui uma comunidade empolgante de desenvolvedores espalhados por todo o mundo.

É uma linguagem totalmente orientada a objetos, mas que permite ao programador iniciante construir *softwares* sem a necessidade de ter um profundo entendimento de POO.

Você pode tirar vantagens da POO, mas ao mesmo tempo você pode usá-la para implementar código sem a necessidade de criar classes e objetos.

Você pode até usar a linguagem *Python* sem precisar escrever um código completo.



Objetivos I

1. Mostrar todas as palavras chaves ou reservadas do Python
2. Aprender sobre a linguagem Python usando o tutorial oficial da comunidade de desenvolvedores
3. Demonstrar o uso de todas as estruturas de dados do Python
4. Demonstrar resumidamente o uso da biblioteca NumPy
5. Demonstrar como usar o Microsoft Copilot para aumentar a produtividade de um desenvolvedor de *software*
 - a. Conceituar e entender o que é criptografia
 - b. Demonstrar outros algoritmos que a IA pode criar automaticamente e como podemos aprender sobre os algoritmos nos dado de presente



Objetivos II

1. Uma breve introdução a biblioteca *pygame* para criação de jogos
2. Introdução a POO
 - a. Criar classes e objetos para entender como eles podem aumentar a clareza, a produtividade e aumentar a facilidade de manutenção de um *software*, através da abstração dos elementos que compõem um *software*
 - b. Implementar um jogo usando biblioteca *pygame* e POO

Objetivos III

1. Mostrar algumas palavras chaves em inglês que auxiliam na busca de soluções de problemas no desenvolvimento de *software*.



Objetivos IV

1. Mostrar a importância de aprender inglês para o futuro profissional
2. Demonstração do Software Google Praktika para aumentar a velocidade do aprendizado de inglês.



“Não precisamos reinventar a roda, é mais vantajoso montar em ombros de gigantes”. (autor desconhecido)





Python IDLE X Python Spyder IDE

Python 3.11.9 (tags/v3.11.9:de54cf5, Apr 2 2024, 10:12:12) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright()", "credits()" and "quit()" for more

```
>>> = RESTART: C:\Users\aleitebr\Scripts\poligonos_regulares_v1.5.py - C:\Users\aleitebr\OneDrive\Matemática\0.1 Meu Curso de Matemática\...
>>> 2 + 2
4
>>> 2**8
256
>>>
```

poligonos_regulares_v1.5.py - C:\Users\aleitebr\OneDrive\Matemática\0.1 Meu Curso de Matemática\...

```
File Edit Format Run Options Window Help
# -*- coding: utf-8 -*-
"""
Created on Tue Jul 16 04:27:06 2024
Última Modificação: 22 de Julho de 2024

@author: aleitebr


Script Name: poligonos_regulares_v1.5.py
"""
import turtle
import math

t = turtle.Pen()
t.pencolor('blue')
turtle.bgcolor('black')

n_arestas_poligono = 2
while (n_arestas_poligono < 3):
    n_arestas_poligono = int(input('Digite o número de aresta: '))
    if n_arestas_poligono < 3:
        print('Erro: Digite num número maior que 2.')

# calculamos os valores do ângulo interno e central do polígono
```

Ln: 15 Col: 0



plugin.py

Plots

get_name

get_description

get_icon

register

unregister

current_widget

add_shellwidget

remove_shellwidget

set_shellwidget

chart_plot_example.py

Plot final terrain model

generate_polar_plot

generate_dem_plot

main

plugin.py

IPythonConsole

init

-- SpyderPluginMixin API

update_font

_apply_gui_plugin_settings

_apply_mpl_plugin_settings

_apply_advanced_plugin_s

_apply_pdb_plugin_setting

apply_plugin_settings_to_c

apply_plugin_settings

toggle_view

-- SpyderPluginWidget AP

get_plugin_title

get_plugin_icon

get_focus_widget

closing_plugin

refresh_plugin

get_plugin_actions

register_plugin

plugin.py - plots

chart_plot_example.py

plugin.py - ipythonconsole

```
1  -*- coding: utf-8 -*-
2  #
3  # Copyright © Spyder Project Contributors
4  # Licensed under the terms of the MIT License
5  # (see spyder/_init_.py for details)
6
7  """
8  Plots Plugin.
9  """
10
11 # Third party imports
12 from qtpy.QtCore import Signal
13
14 # Local imports
15 from spyder.api.plugins import Plugins, SpyderDockablePlugin
16 from spyder.api.translations import get_translation
17 from spyder.plugins.plots.widgets.main_widget import PlotsWidget
18
19
20 # Localization
21 _ = get_translation('spyder')
22
23
24 class Plots(SpyderDockablePlugin):
25     """
26     Plots plugin.
27     """
28     NAME = 'plots'
29     REQUIRES = [Plugins.IPythonConsole]
30     TABIFY = [Plugins.VariableExplorer, Plugins.Help]
31     WIDGET_CLASS = PlotsWidget
32     CONF_SECTION = NAME
33     CONF_FILE = False
34     DISABLE_ACTIONS_WHEN_HIDDEN = False
35
36     # --- SpyderDockablePlugin API
37     # ---
38     def get_name(self):
39         return _('Plots')
40
41     def get_description(self):
42         return _('Display, explore and save console generated plots.')
43
44     def get_icon(self):
45         return self.create_icon('hist')
46
47     def register(self):
48         # Plugins
49         ipyconsole = self.get_plugin(Plugins.IPythonConsole)
50
51         # Signals
52         ipyconsole.sig_shellwidget_changed.connect(self.set_shellwidget)
53         ipyconsole.sig_shellwidget_process_started.connect(
54             self.add_shellwidget)
55         ipyconsole.sig_shellwidget_process_finished.connect(
56             self.remove_shellwidget)
```

Name

Type

Size

Value

bool

bool

1

True

data

Array of str128

(3, 3)

ndarray object of numpy module

datetime_object

datetime

1

2021-04-14 17:35:14.687085

df

DataFrame

(2, 2)

Column names: Col1, Col2

filename

str

53

/Users/Documents/spyder/spyder/tests, test_dont_use.py

li

list

5

['abcd', 745, 2.23, 'efgh', 70.2]

myset

set

3

{'2', '1', '3'}

r

float

1

6.46567886443

t

tuple

5

('abcd', 745, 2.23, 'efgh', 70.2)

tinylis

list

2

[123, 'efgh']

x

float64

1

1.1235123099439

Help

Variable Explorer

Files

Code Analysis

0 %

Plots

IPython console

History

conda: spyder-dev (Python 3.8.5)

LSP Python: ready

master

Line 10, Col 1

UTF-8

LF

RW

Mem 64%



**“Lembrem-se *the best way* para
aprender uma linguagem é praticá-la”.
(autor desconhecido)**



Vantagens da linguagem de programação Python

Em Python é possível escrever programas fáceis de serem lidos por outros programadores e muito mais compactos do que em linguagens como C/C++/Java, pelos motivos abaixo:

1. O alto nível das estruturas de dados permite você fazer operações complexas em uma simples linha de comando
2. Não é necessário fazer declarações prévias de variáveis ou argumentos de funções, o Python possui um sistema de gerenciamento de dados mais dinâmico

Referência: <https://docs.python.org/3/tutorial/appetite.html>



Exemplo Usando Lista

```
>>> lista = ['Bruno', 'Jorge', 'Clayton', 'Maurício']
>>> for nome in lista:
...     print(nome)
...
...
Bruno
Jorge
Clayton
Maurício
>>> |
```



Exemplo Usando Tuplas



Exemplo Usando Dicionários

```
>>> contato1 = {'NOME': 'Bruno', 'EMAIL': 'bruno@gmail.com', 'TELEFONE': '555-1234'}
>>> contato2 = {'NOME': 'Jorge', 'EMAIL': 'jorge@hotmail.com', 'TELEFONE': '233-1234'}
>>> contato3 = {'NOME': 'Clayton', 'EMAIL': 'clayton@outlook.com', 'TELEFONE': '322-1234'}
>>> lista_contatos = [contato1, contato2, contato3]
>>> for contato in lista_contatos:
...     print(f"Nome: {contato['NOME']}")
...     print(f"E-mail: {contato['EMAIL']}")
...     print(f"Telefone: {contato['TELEFONE']}")
...     print()
...
...
Nome: Bruno
E-mail: bruno@gmail.com
Telefone: 555-1234

Nome: Jorge
E-mail: jorge@hotmail.com
Telefone: 233-1234

Nome: Clayton
E-mail: clayton@outlook.com
Telefone: 322-1234
```







