

# Introdução à linguagem Python

Prof. Wanderson Rigo



## Sobre a oficina

### Nível Básico

- ✓ Para iniciantes
- ✓ Exige um pouco de conhecimento de lógica
- ✓ Exige Informática Básica
- ✓ Não focaremos em instalação



# Sobre a oficina

## Alguns códigos em:

✓ <https://github.com/wanderson-rigo/Oficina>



# Por que Aprender Python?

## 01 - Facilidade de Aprendizado

Sintaxe **simples** e **legível**

Uma das linguagens mais **fáceis** de **aprender** e **ler**.

## 02 – Versatilidade

Amplamente utilizado em diversas **áreas**, como **ciência de dados**, **desenvolvimento web** e **automação** de tarefas.

## 03 - Comunidade Ativa

Possui uma das **comunidades** mais ativas, com amplo **suporte**, **bibliotecas** disponíveis e muitos recursos **educacionais**.

# Desafios

## Se adaptar ao conciso. Ex:

```
txt = "Hello World"[:-1]  
print(txt)
```

## Indentação

## Inúmeras bibliotecas...

## Particularidades. Ex:

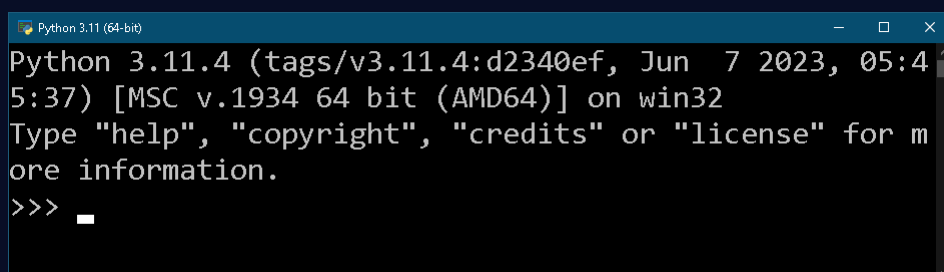
```
__init__  
__main__
```



# Ambiente

## Interpretador Python

É o programa que valida e “entende” o seu código.



# Ambiente

## Visual Studio Code (VS Code)

Editor com suporte a várias ferramentas, como o Git



# Princípios Básicos da Linguagem

## Tipagem Dinâmica

Definir variáveis **sem** especificar o tipo antecipadamente, o que oferece flexibilidade e agilidade ao escrever código.

## Indentação Significativa

No lugar de chaves ou delimitadores, Python utiliza a indentação para definir blocos de código, o que promove uma formatação organizada e legível.

# Roteiro – Atividades Práticas

**Variáveis**

**Entrada / Saída**

**Funções**

**Dicionários**

**Repetição e Seleção**

**Interfaces Gráficas**



## Atividades

- 1) Crie no **VS Code** uma pasta chamada **Oficina**
- 2) Dentro dela crie o arquivo **oficina.py**



# Curiosidades

## Dinamicamente tipada

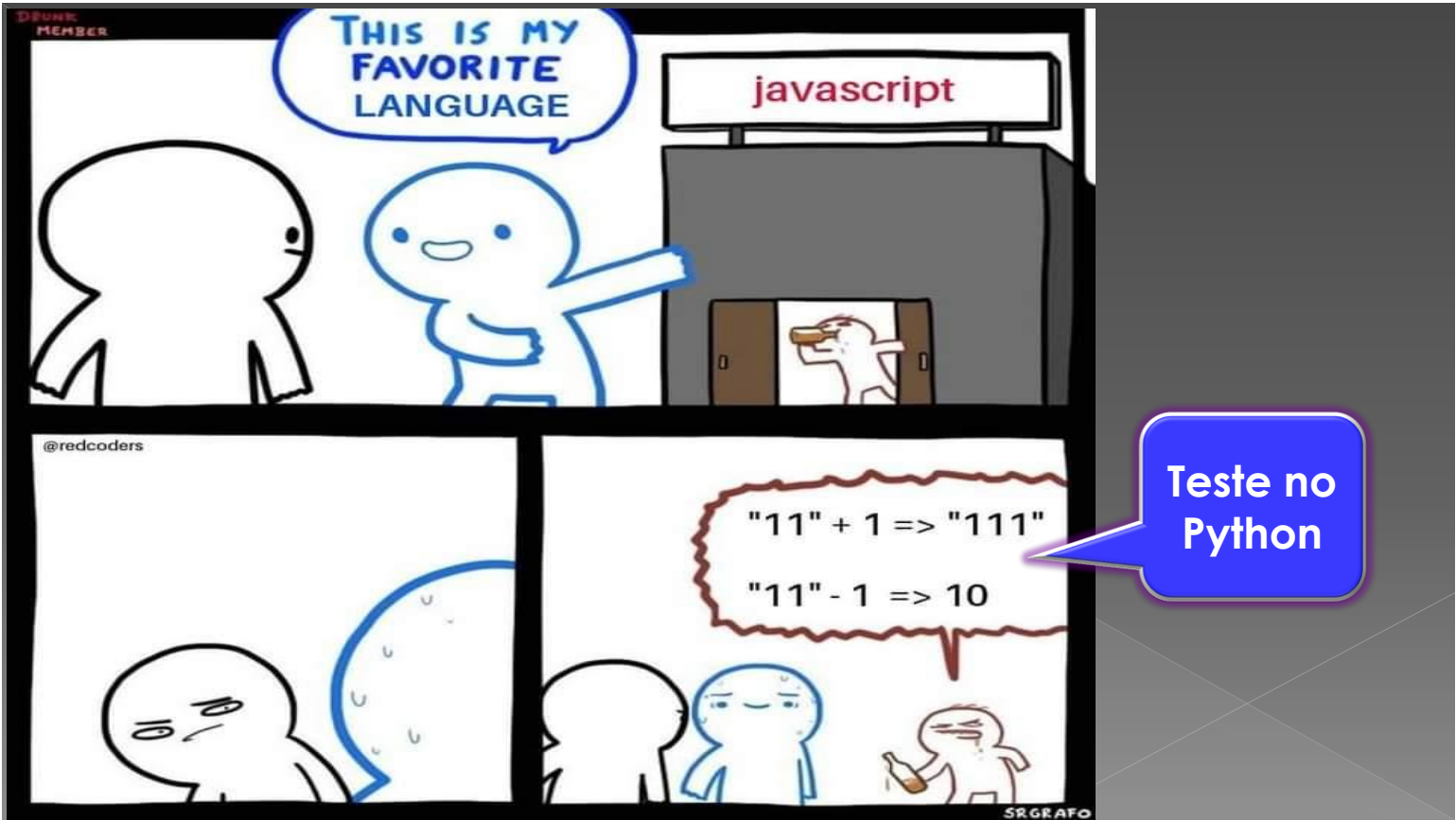
- 1) Crie uma variável **x** e atribua o número 5 a ela  
`x = 5`
- 2) Então imprima o tipo de **x** via  
`print(type(x))` # Output: <class 'int'>
- 3) Logo abaixo atribua um texto a **x**  
`x = "Hello, world!"`
- 4) E então verifique novamente o tipo de **x**  
`print(type(x))` # Output: <class 'str'>

# Curiosidades

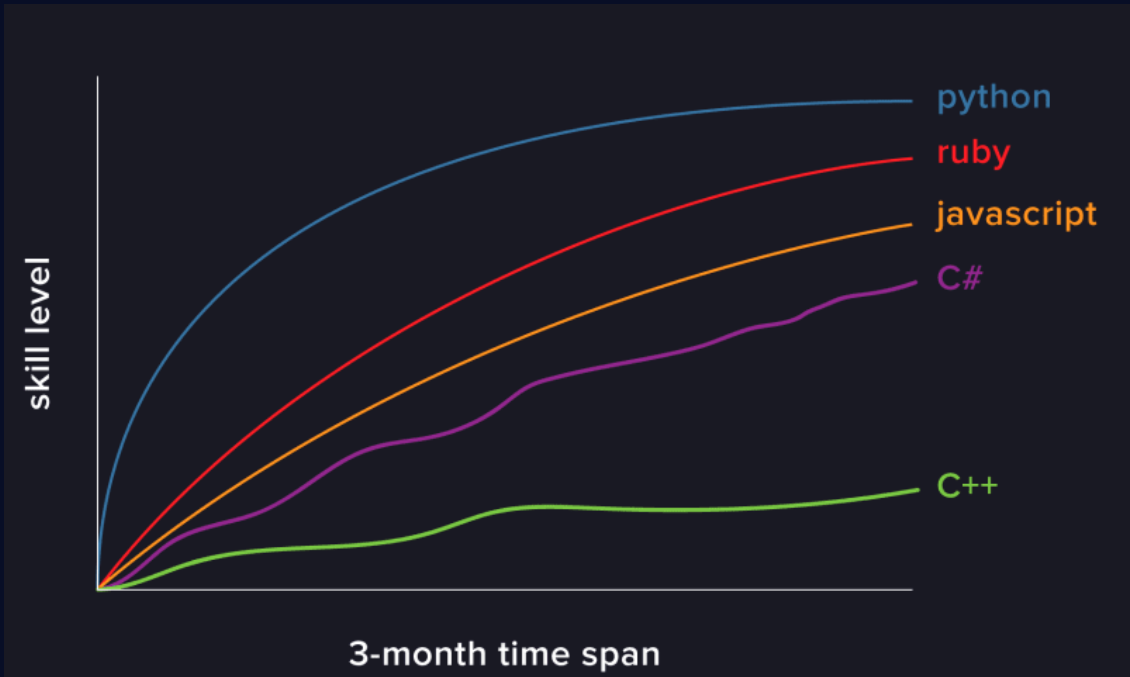
## Tipagem Forte

- 1) Crie uma variável **x** e atribua o texto "11" a ela  
`x = "11"`
  - 2) Crie uma variável **y** e atribua o número 1 a ela  
`y = 1`
  - 3) Crie uma variável **z** e tente atribuir a ela **x + y**  
`z = x + y` # Dispara um erro!
- `TypeError: can only concatenate str (not "int") to str`





# Curva de Aprendizado



# Versátil

Em todo o lugar. **Multiplataforma:**

- 1) Programação *server-side*
- 2) Aplicações GUI desktop
- 3) Programação *front-end*
- 4) Mobile Apps
- 5) Arduino

# Versátil

De tudo que é jeito. **Multiparadigma**

- 1) Orientada a Objeto (POO)
- 2) Estruturada
- 3) Imperativo
- 4) Funcional





# Nos holofotes

Todo mundo usa. **Popular**



<https://www.tiobe.com/tiobe-index>

## Atividades

1) Crie a variável **nome** com o seu nome.

**2) Imprima** o seu nome

```
nome = "Wanderson"  
print(nome)
```

3) Crie a variável **sobrenome** com o seu sobrenome.

**4) Imprima** o nome junto com o sobrenome

5) Adicione **um texto explicativo** junto ao seu nome completo. Ex: "meu nome completo é"

```
print("Meu nome é", nome, sobrenome, ",entendeu?")
```

## Atividades

- 1) Agora peça um nome e guarde numa variável **nome**. Como?

```
nome = input("Digite seu nome:")  
print(nome )
```

- 2) E com valor padrão?

```
nome = input("Digite seu nome:") or "Wanderson"  
print(nome)
```

- 3) Faça o mesmo com o **sobrenome**

## Atividades

- 1) Como contar a **quantidade** de letras do **nome** e do **sobrenome**? **Imprima** tais dados.

```
qtd_nome = len("João")  
print(qtd_nome)  
...
```

- 2) Como **somar** as quantidades?

```
total = qtd_nome + ...  
print(total)
```

# Atividades

1) Como **agrupar** tais comandos numa **função**?

```
def contar_letras():  
    qtd_nome = len("João")  
    qtd_sobrenome = len("Silva")  
    print(qtd_nome)  
    print(qtd_sobrenome)  
    total = qtd_nome + qtd_sobrenome  
    print(total)
```

# Atividades

1) Como retonar a **quantidade** de letras?

```
def contar_letras():  
    qtd_nome = len("João")  
    qtd_sobrenome = len("Silva")  
    ...  
    return qtd_nome, qtd_sobrenome
```

O que tem de interessante aqui?

Retorno de 2 valores!

```
resultado = contar_letras()
```

# Atividades

1) Como retonar a **quantidade** de letras?

```
resultado = contar_letras()  
  
print(resultado) # imprime (4,5)  
  
# fazendo a desestruturação de tupla  
qtd_nome, qtd_sobrenome = resultado
```

# Atividades

1) Como **reusar** a função para qualquer nome?  
Reescrevendo ela com **parâmetros**

```
def contar_letras(nome, sobrenome):  
    qtd_nome = len(nome)  
    qtd_sobrenome = len(sobrenome)  
    return qtd_nome, qtd_sobrenome
```

```
contar_letras("João", "Silva")
```

```
contar_letras("José", "Bueno")
```

## Explicando...

```
def contar_letras(nome, sobrenome):
    qtd_nome = len(nome)
    qtd_sobrenome = len(sobrenome)
    return qtd_nome, qtd_sobrenome
```

1) E se invertermos nome e sobrenome, funciona?

```
contar_letras("Silva", "João")
```

2) Solução via parâmetros nomeados!

```
# usando parâmetros nomeados
resultado = contar_letras(sobrenome="Silva", nome="João")

# desestruturação de tupla
qtd_nome, qtd_sobrenome = resultado
```

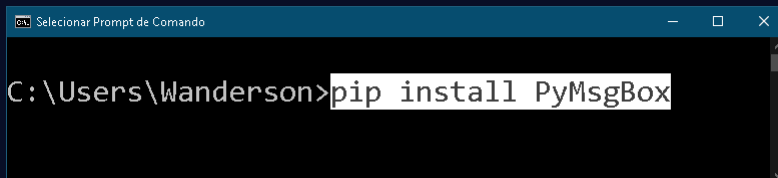
## Atividades

1) Como **pedir** um nome usando **interface gráfica**?



# Como usar **interface gráfica?**

1) Instalar pacote via **pip install PyMsgBox**



```
Selecionar Prompt de Comando
C:\Users\Wanderson>pip install PyMsgBox
```

2) Importar o pacote **via import pymsgbox**

3) Usar **pymsgbox.alert('Mensagem', 'Título')**

```
import pymsgbox
pymsgbox.alert('Este é um alerta!', 'Título do Alerta')
```

# Como usar **interface gráfica?**

1) **Confirmar** algo

```
confirmarResposta =
    pymsgbox.confirm('Está certo disso?',
                    'Confirme',
                    ['Sim', 'Não', 'Talvez'])
```



# Como usar **interface gráfica?**

## 1) **Confirmar** algo com **tempo**

```
confirmarResposta =  
    pymsgbox.confirm('Está certo disso?',  
        'Confirme',  
        ['Sim', 'Não', 'Talvez'],  
        timeout=2000)
```

# Como usar **interface gráfica?**

## 1) **Perguntar** nome

```
nomeResposta = pymsgbox.prompt('Qual o seu nome?')
```

## 2) **Perguntar** nome com **resposta padrão**

```
nomeResposta = pymsgbox.prompt('Qual o seu nome?',  
                                default='Fulano')
```

# Como guardar **dados**?

1) **Dicionário** guardar dados via **chave** → **valor**

**Fulano**

- 123
- fulano@nada.com

**Beltrano**

- 456
- beltrano@nada.com

**Sicrano**

- 789
- sicrano@nada.com

# Como guardar **dados**?

```
# Usuários tem senha e email
usuarios = {
  "Fulano": {
    "senha": "123",
    "email": "fulano@nada.com"
  },
  "Beltrano": {
    "senha": "456",
    "email": "beltrano@nada.com"
  },
  "Sicrano": {
    "senha": "789",
    "email": "sicrano@nada.com"
  }
}
```

# Como guardar **dados**?

## 1) **Iterando** sobre os dados:

```
for usuario in usuarios:  
    print(usuario)
```

**Fulano**

- 123
- fulano@nada.com

**Beltrano**

- 456
- beltrano@nada.com

**Sicrano**

- 789
- sicrano@nada.com

```
for usuario, dados in usuarios.items():  
    email = dados["email"]  
    senha = dados["senha"]  
    print(f "User: {usuario}, Email: {email}, Senha: {senha}")
```

**f-strings**

# Como guardar **dados**?

## 1) **Selecionando** dados:

**Fulano**

- 123
- fulano@nada.com

**Beltrano**

- 456
- beltrano@nada.com

**Sicrano**

- 789
- sicrano@nada.com

```
#imprima se a senha for igual a 123  
if senha == "123":  
    print(f"Usuário: {usuario} autorizado")  
else  
    print("Senha errada")
```

## Atividade Final 01

- 1) Pergunte pelo nome do usuário
- 2) Então pergunte pela senha usando o nome fornecido
- 3) Se a senha for igual 1234 alerte que “Acertou!”.
- 4) Senão, diga “Usuário ou senha inválidos” e depois ofereça as opções:
  - a) **Cadastrar Novo usuário**
    - 1) Perguntar nome e senha
    - 2) Alertar “Cadastrado com sucesso”
  - b) **Recuperar a senha**
    - 1) Verificar se o usuário é igual a Fulano
    - 2) Se for, mostre a senha dele. Senão alerte “usuário não existe”.



## Atividade Final 02 (para casa)

- 1) Crie um programa com **interface gráfica**.
- 2) Sugestão 01: usar a biblioteca **tkinter**
- 3) Sugestão 02: Ver tutorial “Tkinter: Interfaces gráficas em Python”  
<https://www.devmedia.com.br/tkinter-interfaces-graficas-em-python/33956>
- 4) Sugestão 03: ferramenta gráfica em <https://visualtk.com>





## Concluindo

Python é uma linguagem versátil, com uma sintaxe simples e clara. Aprender Python abrirá portas para um mundo de possibilidades e oportunidades.

**Bons programas!!!**