

PYTHON

professor



Lázaro
Santos



Argnivios

Yype





FUNÇÃO TYPE

A função type **retorna** o tipo de uma variável, função ou objeto em Python.

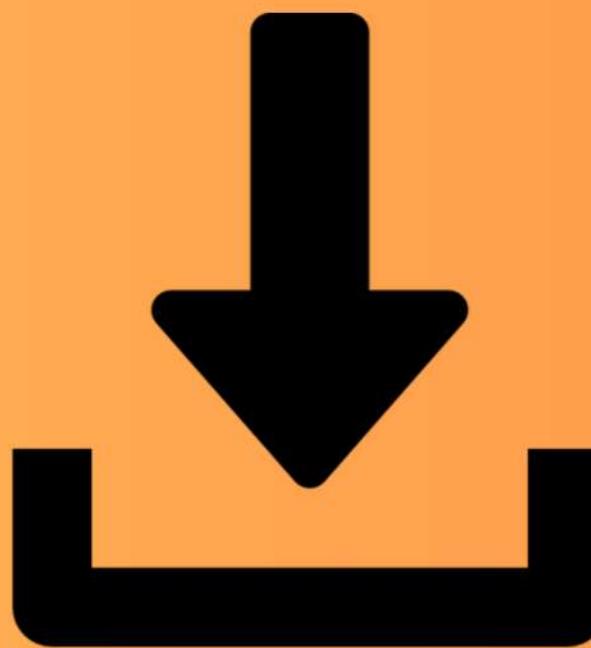




FUNÇÃO TYPE

Acesse o módulo Listagem 8.43 e faça o download.

Disponível na pasta exercícios do
repositório Alunos_Fabrica_Programadores:



<https://github.com/evitarafadiga>



EXERCÍCIO

Realize todos os testes possíveis da função `diz_o_tipo()` passando no mínimo 8 tipos diferentes como parâmetro, um de cada vez.

Ex:

```
print(diz_o_tipo("Alô"))
```

Arg nivog



O QUE É UM ARQUIVO?

Quando precisamos **armazenar** dados permanentemente, arquivos são uma **excelente** forma de entrada e saída de dados para programas.



Mas o que é um arquivo?

Arquivo é uma área em disco onde podemos ler e gravar informações. Essa área é gerenciada pelo sistema operacional do computador. Ele é acessado pelo nome e é onde podemos ler e escrever linhas de texto ou dados em geral.



ARQUIVOS

Para acessar um arquivo precisamos abri-lo. Informamos o nome do arquivo, com o nome do diretório (pasta) onde ele está (se necessário) e qual operação queremos realizar: leitura e/ou escrita.

Em Python, abrimos arquivos com a função `open()`

Essa função utiliza os parâmetros nome e modo.



ESCREVENDO ARQUIVO

Por exemplo:

```
arquivo=open("números.txt","w")
for linha in range(1,101):
    arquivo.write("%d\n" % linha)
arquivo.close()
```

Não esqueça do bloco try!



MODOS DE ABRIR ARQUIVO

Utilizamos a função **write** para escrever ou gravar dados no arquivo, **read** para ler e **close** para fechar.

Quando trabalhamos com arquivos, devemos sempre realizar:



1. abertura,
2. leitura e/ou escrita,
3. fechamento.



MODOS DE ABRIR ARQUIVO

Modo	Operações
r	leitura
w	escrita, apaga o conteúdo se já existir
a	escrita, mas preserva o conteúdo se já existir
b	modo binário
+	atualização (leitura e escrita)



LENDI ARQUIVO

Por exemplo:

```
arquivo=open("números.txt","r")
for linha in arquivo.readlines():
    print(linha)
arquivo.close()
```

Não esqueça também, do bloco try!



ARQUIVOS

Gravação de
números pares e
**ímpares em arquivos
diferentes**

```
ímpares=open("ímpares.txt","w")
pares=open("pares.txt", "w")
for n in range(0,1000):
    if n % 2 == 0:
        pares.write("%d\n" % n)
    else:
        ímpares.write("%d\n" % n)
ímpares.close()
pares.close()
```



MODOS DE ABRIR ARQUIVO

Os modos podem ser combinados
("r+", "w+", "a+", " r+b", "w+b", "a+b").



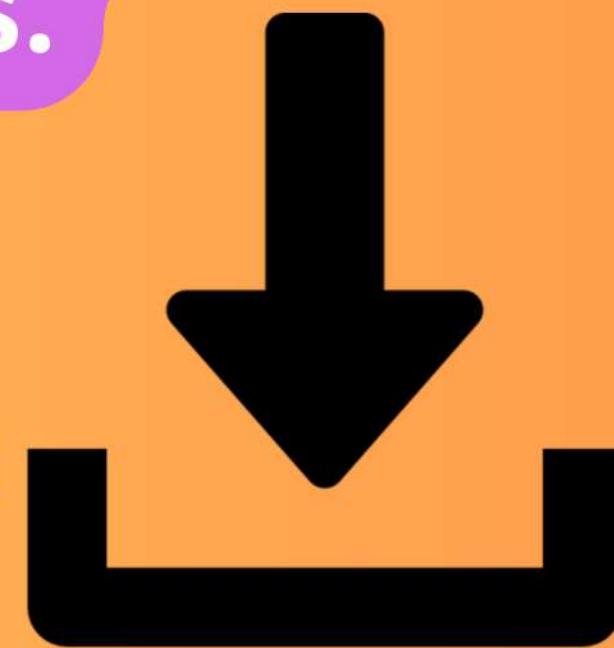
A função open retorna um objeto do tipo file (arquivo). É esse objeto que vamos utilizar para ler e escrever os dados no arquivo.





ARQUIVOS

Acesse o módulo `lê_binário.py` e imagem `binary.jpg` e faça o download de ambos.



Disponível na pasta materiais complementares do repositório `Alunos_Fabrica_Programadores`:

<https://github.com/evitarafadiga>



DESAFIO

Modifique o programa lê_binário.py de forma que ele verifique a existência de um arquivo.txt (você vai criar ele e digitar uma mensagem dentro) e se caso exista, crie mais três cópias desse arquivo com os dados exatos.

Móculos os e Sys



MÓDULO OS

Pense no módulo os (que vem de Sistema Operacional) como o controle remoto do seu computador.

Ele permite que o seu código Python converse diretamente com o Windows, o Linux ou o sistema que você estiver usando.





MÓDULO OS

Pra que serve?

- **Criar pastas:** para guardar as fotos do rolê ou os trabalhos da escola...
- **Listar arquivos:** para ver tudo que tem dentro de uma pasta sem precisar abrir o explorador de arquivos...
- **Mover e renomear arquivos:** Baixou um arquivo com nome errado? Você pode usar Python pra renomeá-lo para algo mais apropriado.





DESAFIO

Exiba o resultado em um print()

```
import os  
os.getcwd()
```



MÓDULO OS

Criando pastas

```
import os  
os.mkdir("d")  
os.mkdir("e")  
os.mkdir("f")
```

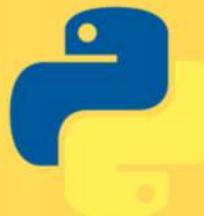


MÓDULO OS

```
import os

arquivo_path = "diretório"
try:
    os.rmdir(arquivo_path)
    print(f"\033[0;32m Pasta '{arquivo_path}'"
          'removida com sucesso! ;D""")"
except FileNotFoundError:
    print(f"\033[0;33m Pasta '{arquivo_path}'"
          'não encontrada... @_@""")"
except OSError:
    print(f"\033[0;34m '{arquivo_path}'"
          'é um arquivo, não uma pasta!!! +_""")"
```

Removendo uma pasta



DESAFIO

Remova as pastas *d*, *e* e *f* que você criou utilizando a função `rmdir()` do módulo `os`.





MÓDULO OS

Removendo um
arquivo

```
import os

arquivo_path = "texto1"
try:
    os.remove(arquivo_path)
    print(f"\033[0;32m Arquivo '{arquivo_path}'\nremovido com sucesso! ;D")
except FileNotFoundError:
    print(f"\033[0;33m Arquivo '{arquivo_path}'\nnão encontrado... @_@")
except OSError:
    print(f"\033[0;34m '{arquivo_path}'\né uma pasta, não arquivo!!! +_+")
```

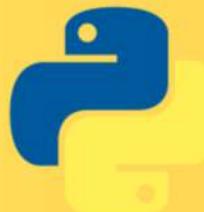


MÓDULO SYS

Ele é como um mecânico que consegue ver e mexer nas peças do motor do seu programa em tempo real.



Se o os cuida do computador, o sys (de System, ou Sistema) cuida do próprio Python enquanto ele está rodando..



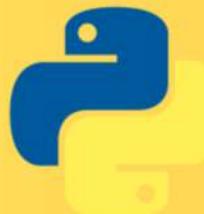
MÓDULO SYS

Podemos acessar os parâmetros passados ao programa na linha de comando utilizando o módulo sys e trabalhando com a lista *argv*.

```
import sys  
  
print("Número de parâmetros: %d" % len(sys.argv))  
  
for n,p in enumerate(sys.argv):  
    print("Parâmetro %d = %s" % (n,p))
```

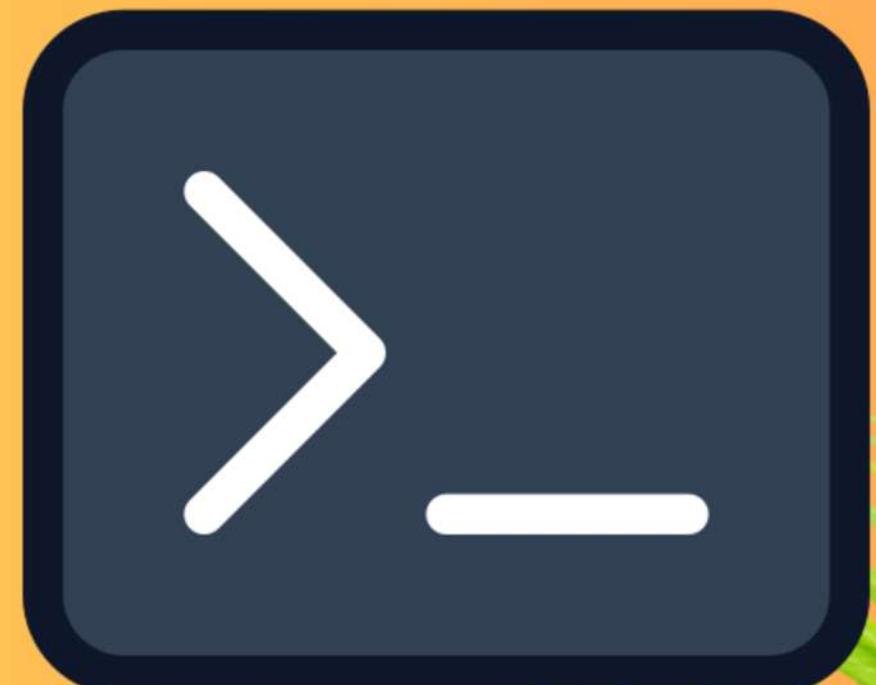
Experimente **criar** e **rodar** o script no terminal usando os seguintes parâmetros:

The screenshot shows a dark-themed terminal window with several tabs at the top: SAÍDA, CONSOLE DE DEPURAÇÃO, TERMINAL (which is underlined in green), and PORTAS. Below the tabs, there's a search bar with a magnifying glass icon and some dropdown menus. The main area of the terminal displays the command `python sys.py primeiro segundo terceiro` followed by its output: `Número de parâmetros: 4`, `Parâmetro 0 = sys.argv`, `Parâmetro 1 = primeiro`, `Parâmetro 2 = segundo`, and `Parâmetro 3 = terceiro`.



EXERCÍCIO

Exercício 9.4 Crie um programa que receba o nome de dois arquivos como parâmetros da linha de comando e que gere um arquivo de saída com as linhas do primeiro e do segundo arquivo.



*Página Web
Simples*



PÁGINAS WEB

Você não precisa de um programa super complexo para criar uma página web. Na verdade, uma página web é apenas um arquivo de texto formatado com uma linguagem (de marcação de texto) chamada HTML. A parte legal é que você pode usar Python para gerar esse arquivo HTML de forma automática!





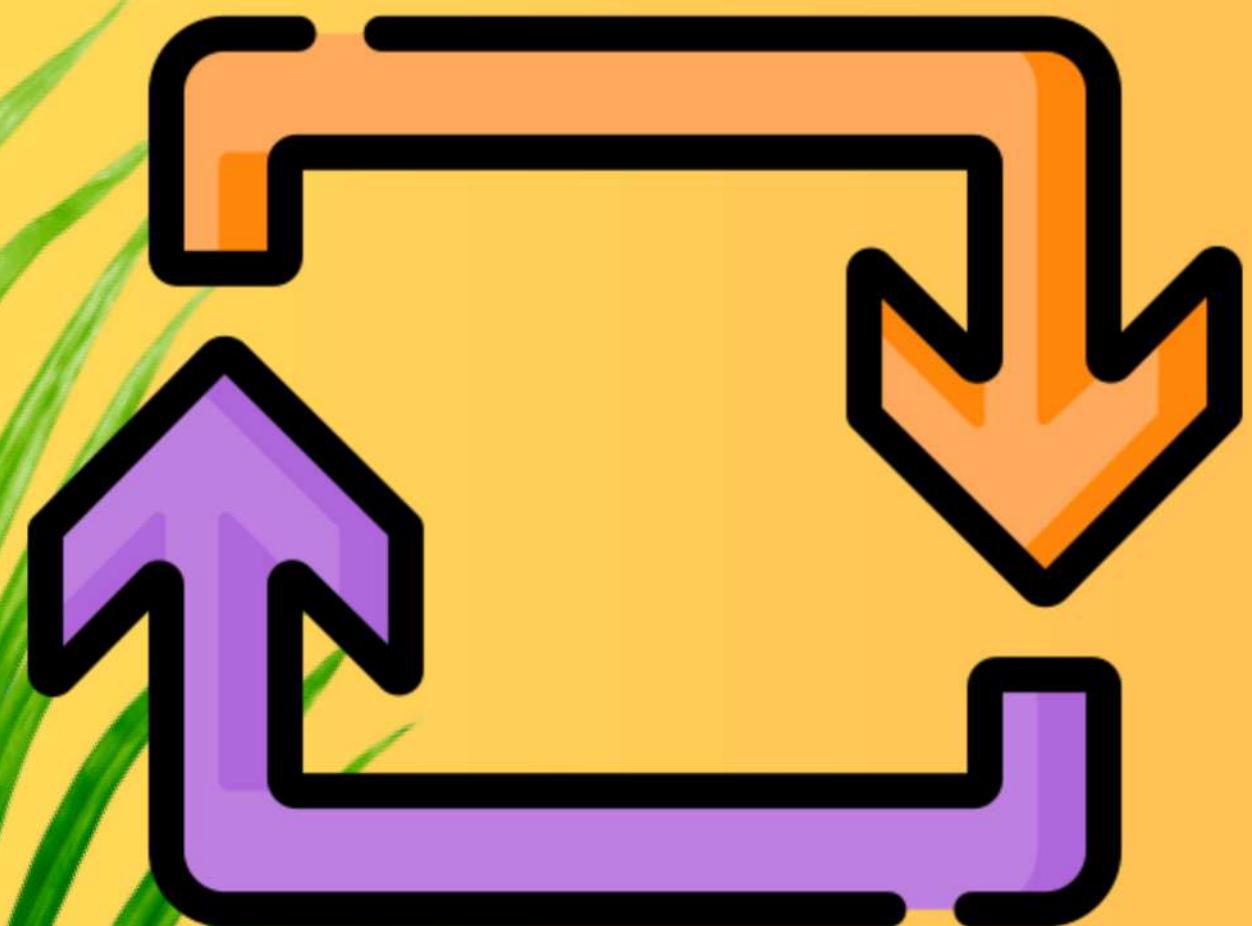
PÁGINAS WEB

Imagine que você tem essa lista no Python:
`series = ['Stranger Things', 'Dark', 'The Office']`
Seu script Python pode gerar um arquivo .html com o conteúdo dessa lista! Quando você abrir esse arquivo no navegador, verá uma página simples com um título e a lista das suas séries.





PÁGINAS WEB



A grande vantagem do Python é a automação: se sua lista tiver 100 séries, o Python escreve o HTML para todas elas com poucas linhas de código, usando um laço de repetição (for).

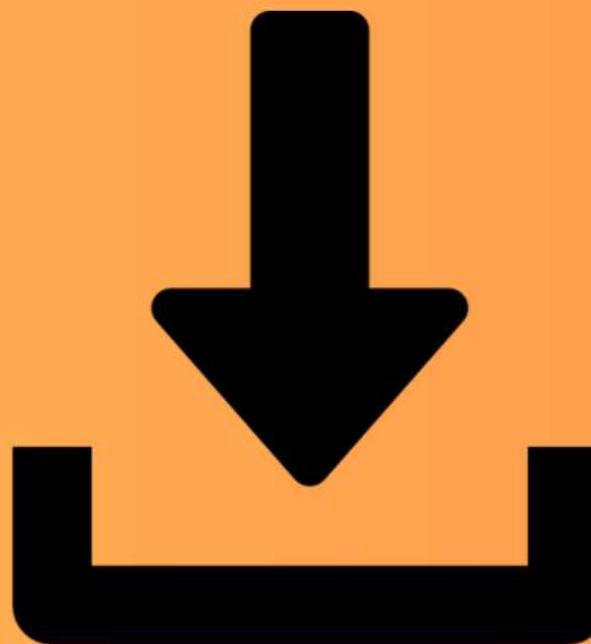
É uma ótima maneira de começar a entender como a web funciona!



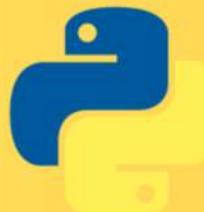
ARQUIVOS

Acesse o módulo Listagem 9.10 e faça o download.

Disponível na pasta exercícios do
repositório Alunos_Fabrica_Programadores:



<https://github.com/evitarafadiga>



EXERCÍCIOS

Exercício 9.29 Modifique o programa da listagem 9.10 para utilizar o elemento p em vez de h2 nos filmes.

Exercício 9.30 Modifique o programa da listagem 9.10 para gerar uma lista html, usando os elementos ul e li. Todos os elementos da lista devem estar dentro do elemento ul, e cada item dentro de um elemento li. Exemplo:

```
<ul><li>Item1</li><li>Item2</li><li>Item3</li></ul>
```

REFERÊNCIA BIBLIOGRÁFICA:



MENEZES, Nilo Ney Coutinho.;
Introdução à Programação com
Python, ed. 2. Novatec, 2014.

INTRODUÇÃO À PROGRAMAÇÃO COM
PYTHON

Algoritmos
e lógica de programação
para iniciantes

novatec

Nilo Ney Coutinho Menezes

2ª EDIÇÃO
Revisada e ampliada