

# A arte do Python



**Autor:**

**Breno de Souza S.**

O Guerreiro Silencioso nas Empresas





# INTRODUÇÃO

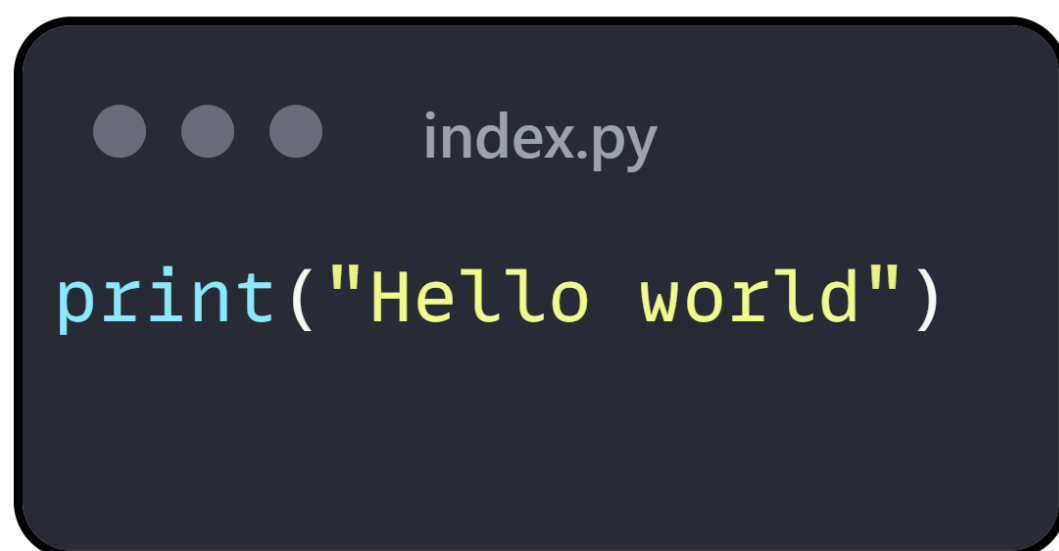
---

# Introdução

## Introdução ao Python: Por que Escolher Python?

Python é uma linguagem de programação versátil, fácil de aprender e amplamente utilizada em diversas áreas como desenvolvimento web, ciência de dados, automação, entre outras. Sua sintaxe simples e clara permite que iniciantes e profissionais desenvolvam projetos rapidamente e com eficiência.

A indentação é fator chave na linguagem python , uma vez que o código obriga a utilização da organização por espaços.





# Variáveis e tipos de dados

---

# Variáveis e tipos de dados

## Armazene Informações com Facilidade

Em Python, variáveis são usadas para armazenar dados que podem ser de diferentes tipos: inteiros, flutuantes, strings, listas, entre outros.

Sua instância deve ser feita dando um nome e um valor.

Em determinados casos onde se tem variáveis de nome composto é aconselhável utilizar algumas convenções para a instancia de um nome como por exemplo o Snake case (variavel\_nome) ou camel case(variavelNome).

```
index.py

# Inteiro
idade = 25

# Flutuante
altura = 1.75

# String
nome = "Ana"

# Lista
frutas = ["maçã", "banana", "laranja"]
```



# **ESTRUTURAS CONDICIONAIS**

---

# Estruturas condicionais

Tome decisões no seu código

As estruturas condicionais permitem que seu código execute diferentes blocos de instrução com base em condições.

Para utilizá-las é necessário utilizar : *if*, *else* ou *elif*.

```
index.py

idade = 18

if idade ≥ 18:
    print("Você é maior de idade.")
else:
    print("Você é menor de idade.")
```



# Laços de repetição

---



# Laços de repetição

Automatize tarefas repetitivas

Os laços de repetição, como *for* e *while*, são usados para executar um bloco de código várias vezes. Enquanto o laço *for* repete de acordo com uma condição quando sabemos a quantidade de vezes que será feita, o laço *while* é repetido enquanto aquela condição for verdadeira, ideal para quando não se sabe quantas vezes será necessário repetir.

```
index.py

for fruta in frutas:
    print(fruta)

# Usando while
contador = 0
while contador < 5:
    print(contador)
    contador += 1
```



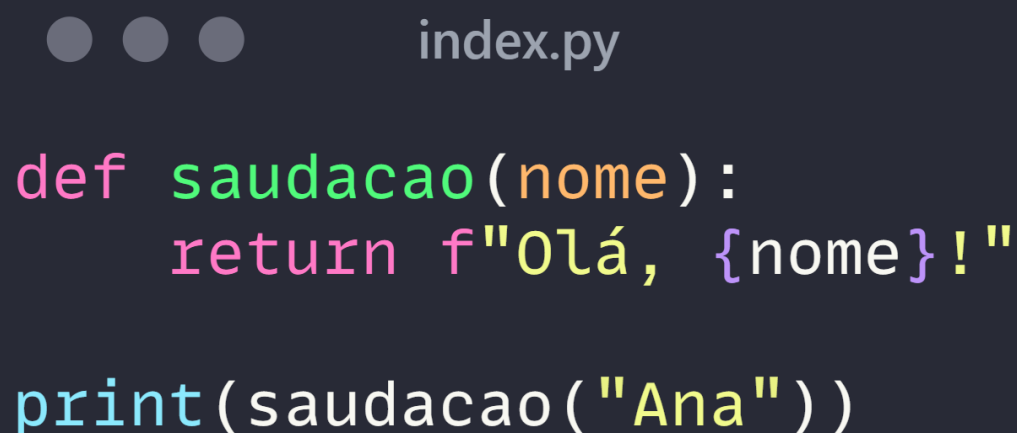
# Funções

---

# Funções

## Reutilize Código com Facilidade

Funções são blocos de código que realizam uma tarefa específica e podem ser reutilizados. A criação de funções em python são muito simples , basta utilizar o método def e utilizar o nome da função com um parênteses onde será passados parâmetros para a função caso seja necessário.



```
def saudacao(nome):  
    return f"Olá, {nome}!"  
  
print(saudacao("Ana"))
```



# Manipulação de listas

---



# Manipulação de listas

Trabalhe com Conjuntos de Dados com Facilidade

Listas ou *Arrays* são estruturas de dados que permitem armazenar múltiplos itens em uma única variável. Sua instancia pode ser feita por meio de colchetes onde cada item será separado por virgula. Existem vários métodos utilizados para a manipulação de listas , como por exemplo o *append* que adiciona um item ao final da lista e o *remove* que exclui o último item da lista.

```
index.py

frutas.append("manga")

# Removendo itens da lista
frutas.remove("banana")

# Acessando itens da lista
print(frutas[0]) # Imprime "maçã"
```



# Dicionários

---

# Dicionários

Armazene dados com chave-valor

Dicionários são estruturas de dados que armazenam pares de chave-valor, facilitando a busca e organização de dados. As estruturas de dicionário são muito importantes para a manipulação de dados no python por isso vale a pena se aprofundar nesse assunto. A instancia de um dicionário é feita por meio de chaves onde cada item deve ter 2 atributos chave e valor.

```
index.py

aluno = {
    "nome": "Carlos",
    "idade": 20,
    "curso": "Engenharia"
}

print(aluno["nome"]) # Imprime "Carlos"
```



# Manipulação de arquivos

---



# Manipulação de arquivos

Leia e escreva dados externos

Manipular arquivos é essencial para leitura e gravação de dados em Python. Para poder manipular com arquivos é necessário atribuir o caminho do arquivo , o modo de acesso , além de possuir três formas de ler *read()*, *readline()* e *readlines()*

```
index.py

with open("arquivo.txt", "w") as arquivo:
    arquivo.write("Olá, mundo!")

# Lendo de um arquivo
with open("arquivo.txt", "r") as arquivo:
    conteudo = arquivo.read()
    print(conteudo)
```



# Dicas finais

---

# Dicas finais

Pratique e aprenda constantemente

Aprender Python é um processo contínuo. Pratique sempre, explore novas bibliotecas e frameworks, e mantenha-se atualizado com as melhores práticas e novidades da linguagem.

Python é uma ferramenta poderosa que, com esses princípios básicos, permite criar soluções eficazes e inovadoras. Continue explorando e desenvolvendo suas habilidades para se tornar um expert em Python.

- [Python.org Beginner's Guide](#)
- [W3Schools Python Tutorial](#)
- [Real Python Tutorials](#)



# Obrigado por ler até aqui!



Esse Ebook foi gerado por IA, e diagramado por humano.

Esse conteúdo foi gerado com fins didáticos de construção, não foi realizado uma validação cuidadosa humana no conteúdo e pode conter erros gerados por uma IA.



<https://github.com/BrenoSouS/EbookDIO>

