Python

O Grimório do Programador

Rafael Bezerra

Bem-vindo ao universo do Python! Neste eBook, você encontrará dicas práticas e exemplos reais para aprimorar suas habilidades de programação de forma simples e direta.

Python é uma das linguagens mais populares e versáteis, e este guia foi criado para ajudá-lo a aprender de forma eficiente e aplicada, com foco em situações do dia a dia.

Prepare-se para desbravar o mundo da programação e dominar a magia do Python com cada linha de código!

1

Trabalhando com Listas: Adicionando e Acessando Dados

As listas são uma das estruturas de dados mais importantes em Python. Elas permitem armazenar múltiplos itens em uma única variável. Aqui estão algumas dicas simples para trabalhar com listas:

Adicionar Itens à Lista

Se você quiser adicionar novos itens a uma lista, pode usar o método append() ou insert().

Exemplo:

add-lista

```
# Lista de compras
compras = ["maçã", "banana", "leite"]

# Adicionando um item ao final da lista
compras.append("arroz")

# Adicionando um item em uma posição específica
compras.insert(1, "ovos")

print(compras)  # ['maçã', 'ovos', 'banana',
#'leite', 'arroz']
```

Acessando Itens da Lista

Você pode acessar qualquer item da lista usando o índice (começando de 0).

Exemplo:

read-lista

```
# Acessando o primeiro item
print(compras[0])  # 'maçã'

# Acessando o último item
print(compras[-1])  # 'arroz'
```

2

Trabalhando com Dicionários: Acessando e Modificando Dados

Os dicionários são ideais para armazenar pares de chave-valor. Eles são úteis quando você precisa associar informações de maneira estruturada.

Criando e Modificando um Dicionário

Exemplo:

create-dic

```python
# Dicionário com informações de uma pessoa
pessoa = {
    "nome": "João",
    "idade": 30,
    "cidade": "São Paulo"
}

# Acessando o valor de uma chave
print(pessoa["nome"])  # 'João'

# Modificando um valor
pessoa["idade"] = 31

print(pessoa)  # {'nome': 'João', 'idade': 31,
#'cidade': 'São Paulo'}
```

Adicionando Novos Itens ao Dicionário

Exemplo:

add-dic

```
# Adicionando novo item
pessoa["profissao"] = "Engenheiro"

print(pessoa)  # {'nome': 'João', 'idade': 31,
#'cidade': 'São Paulo', 'profissao': 'Engenheiro'}
```

Condicionais: Tomando Decisões no Código

Condicionais ajudam a controlar o fluxo do programa, tomando decisões com base em certas condições.

Uso Básico do if

Exemplo:

if-idade

```
idade = 18

if idade ⩾ 18:
    print("Você é maior de idade!")
else:
    print("Você é menor de idade!")
```

Neste exemplo, o programa decide o que imprimir com base no valor da variável idade.

Loops: Iterando sobre Dados

Os loops são essenciais para repetir ações sem precisar escrever o mesmo código várias vezes.

Loop for: Iterando sobre uma lista

Exemplo:

loop-frutas

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

# Usando for para percorrer a lista
for fruta in frutas:
    print(f"A fruta é: {fruta}")
```

Loop while: Repetindo até uma condição ser falsa

Exemplo:

loop-while

```
# Contando até 5
contador = 1
while contador <= 5:
    print(contador)
    contador += 1
```

5

Funções: Organize seu Código

Funções são blocos de código que você pode reutilizar. Elas ajudam a tornar o código mais organizado e modular.

Definindo e Chamando uma Função

Exemplo:

funcao-soma

```python
# Função para somar dois números
def somar(a, b):
    return a + b

# Chamando a função
resultado = somar(5, 3)
print(resultado)  # 8
```

Funções com Parâmetros Opcionais

Você também pode definir valores padrão para parâmetros de função.

Exemplo:

funcao-param

```python
def saudacao(nome="Visitante"):
    print(f"Olá, {nome}!")

saudacao("Maria")  # Olá, Maria!
saudacao()  # Olá, Visitante!
```

6

Trabalhando com Arquivos: Leitura e Escrita

Você pode ler e escrever dados em arquivos com Python. Aqui está uma dica para isso:

Escrevendo e Lendo Arquivos

Exemplo:

wr-file

```python
# Abrindo um arquivo para escrita
with open("arquivo.txt", "w") as file:
    file.write("Olá, Python!")

# Lendo o conteúdo do arquivo
with open("arquivo.txt", "r") as file:
    conteudo = file.read()
    print(conteudo)  # Olá, Python!
```

O with é usado para garantir que o arquivo seja fechado automaticamente após a operação.

Agradecimentos

Obrigado por ler até aqui!

Este eBook foi gerado por IA e diagramado por um humano. O passo a passo se encontra em meu Github no link abaixo.

https://github.com/RafaSant13/prompts-recipe-to-create-a-ebook