O ENIGMA DOS DICIONÁRIOS

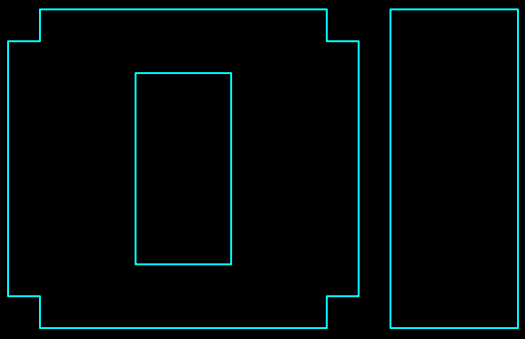
UMA AVENTURA PYTHON ÉPICA

TALITA D. SILVA

BEM-VINDO AO EBOOK

O Enigma dos Dicionários: Uma Aventura Python Épica

Bem-vindo ao nosso guia prático sobre dicionários em Python! Este ebook foi elaborado para fornecer uma compreensão clara e abrangente sobre uma das estruturas de dados mais poderosas da linguagem Python: os dicionários.



INTRODUÇÃO

AOS DICIONÁRIOS EM PYTHON

INTRODUÇÃO AOS DICIONÁRIOS EM PYTHON

Os dicionários são estruturas de dados fundamentais em Python, permitindo armazenar pares chave-valor de forma eficiente. Em um dicionário, cada chave é única e associada a um valor específico. Esta estrutura é extremamente flexível e é amplamente utilizada em Python para mapeamentos rápidos de dados.

02

CRIANDO
E ACESSANDO DICIONÁRIOS

INTRODUÇÃO AOS DICIONÁRIOS EM PYTHON

Para criar um dicionário em Python, utilizamos chaves **{}** e separamos cada par chave-valor por dois pontos **:**. Vejamos um exemplo simples:

```python
meu_dicionario = {'nome': 'João', 'idade': 30, 'cidade': 'São Paulo'}
```

Podemos acessar os valores através das chaves:

```python
print(meu_dicionario['idade'])  # Saída: 30
```

03

ADICIONANDO

E MODIFICANDO ELEMENTOS

ADICIONANDO E MODIFICANDO ELEMENTOS

Dicionários em Python são mutáveis, o que significa que podemos adicionar novos pares chave-valor ou modificar os existentes:

```python
# Adicionando um novo elemento
meu_dicionario['profissão'] = 'Engenheiro’
# Modificando um valor existente
meu_dicionario['idade'] = 31
print(meu_dicionario)
# Saída: {'nome': 'João', 'idade': 31, 'cidade': 'São Paulo’,
          'profissão': 'Engenheiro'}
```

04

MÉTODOS
ÚTEIS DE DICIONÁRIOS

MÉTODOS ÚTEIS DE DICIONÁRIOS

Python oferece diversos métodos integrados para manipular dicionários de maneira eficiente. Vejamos alguns exemplos:

keys(), values() e items()

Estes métodos permitem acessar as chaves, valores e itens (pares chave-valor) do dicionário, respectivamente:

```python
print(meu_dicionario.keys())
# Saída: dict_keys(['nome', 'idade', 'cidade’,  'profissão’])

print(meu_dicionario.values())
# Saída: dict_values(['João', 31, 'São Paulo', 'Engenheiro’])

print(meu_dicionario.items())
# Saída: dict_items([('nome', 'João'), ('idade', 31), ('cidade’,
                     'São Paulo'), ('profissão', 'Engenheiro')])
```

MÉTODOS ÚTEIS DE DICIONÁRIOS

get()

O método get() permite acessar o valor de uma chave de forma segura, retornando None se a chave não existir:

```python
profissao = meu_dicionario.get('profissão’)
print(profissao)  # Saída: 'Engenheiro’
altura = meu_dicionario.get('altura’)
print(altura)     # Saída: None
```

pop() e popitem()

Estes métodos permitem remover itens de um dicionário:

- **pop()** remove um item com a chave especificada e retorna o valor associado:

```python
cidade = meu_dicionario.pop('cidade’)
print(cidade)     # Saída: 'São Paulo’
print(meu_dicionario)  # Saída: {'nome': 'João', 'idade': 31,
                                'profissão': 'Engenheiro'}
```

MÉTODOS ÚTEIS DE DICIONÁRIOS

- **popitem()** remove e retorna o último par chave-valor inserido no dicionário como uma tupla:

```python
item_removido = meu_dicionario.popitem()
print(item_removido)   # Saída: ('profissão', 'Engenheiro')
print(meu_dicionario)  # Saída: {'nome': 'João', 'idade': 31}
```

update()

O método update() permite mesclar dois dicionários, atualizando o dicionário atual com os pares chave-valor do dicionário fornecido como argumento:

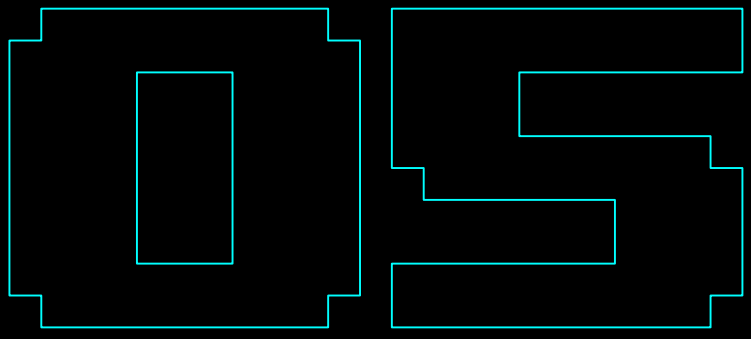
```python
outro_dicionario = {'profissão': 'Analista', 'cidade': 'Rio de Janeiro'}
meu_dicionario.update(outro_dicionario)
print(meu_dicionario)
# Saída: {'nome': 'João', 'idade': 31, 'profissão': 'Analista',
          'cidade': 'Rio de Janeiro'}
```

MÉTODOS ÚTEIS DE DICIONÁRIOS

clear()

O método clear() remove todos os itens do dicionário:

```python
meu_dicionario.clear()
print(meu_dicionario)  # Saída: {}
```



ITERAÇÃO

EM DICIONÁRIOS

MÉTODOS ÚTEIS DE DICIONÁRIOS

Podemos iterar sobre dicionários utilizando loops for, o que nos permite acessar tanto as chaves quanto os valores:

```python
meu_dicionario = {'nome': 'João', 'idade': 31, 'profissão': 'Engenheiro’}

# Iterando sobre as chaves
for chave in meu_dicionario:
    print(chave)
# Saída: nome idade profissão

# Iterando sobre os valores
for valor in meu_dicionario.values():
    print(valor)
# Saída: João 31 Engenheiro

# Iterando sobre os itens
for chave, valor in meu_dicionario.items():
    print(f'{chave}: {valor}’)
# Saída: nome: João idade: 31 profissão: Engenheiro
```

06

COMPREENSÃO

DE DICIONÁRIOS

COMPREENSÃO DE DICIONÁRIOS

Assim como listas e conjuntos, é possível criar dicionários usando compreensão de dicionários:

```python
# Exemplo de compreensão de dicionário
numeros = [1, 2, 3, 4, 5]
quadrados = {num: num**2 for num in numeros}
print(quadrados)
# Saída: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

07

CONSIDERAÇÕES FINAIS

CONSIDERAÇÕES FINAIS

Os dicionários são uma ferramenta poderosa em Python para manipulação eficiente de dados estruturados. Com sua flexibilidade e variedade de métodos integrados, são ideais para muitas aplicações, desde armazenamento de configurações até mapeamento de dados complexos. Dominar o uso de dicionários ampliará suas habilidades de programação em Python de maneira significativa.

AGRADECIMENTOS

OBRIGADO POR LER ATÉ AQUI

Esse Ebook foi gerado por IA, e diagramado por humano.
O passo a passo se encontra no meu Github.

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.

ACESSE O REPOSITÓRIO GITHUB

👩‍💻 Autor

Talita Dias

GitHub | LinkedIn