REVISÃO

Karina CASOLA

SENAC - FUTURO PROGRAMADOR

O que são dicionários em Python?

Os dicionários são uma estrutura de dados muito poderosa em Python, utilizada para armazenar informações de forma organizada e acessível.

Eles funcionam como uma espécie de "tabela", onde cada item é composto por uma **chave** e um **valor**. Isso permite buscar informações de maneira eficiente sem percorrer todos os elementos, como ocorre com listas.

Características principais dos dicionários

- **Estrutura de chave e valor:** Cada item possui uma chave única que acessa um valor.
- **Acesso rápido:** Basta referenciar a chave — não é necessário percorrer a estrutura.
- **Mutáveis:** É possível adicionar, remover ou modificar itens.
- **Não ordenados:** Antes do Python 3.7, a ordem dos elementos não era garantida. A partir do 3.7+, é.

Exemplo básico de dicionário

```
usuario = {
    "nome": "Maria",
    "idade": 55,
    "cidade": "Palhoça",
    "profissão": "Engenheira"
}

print(usuario["nome"])    # Saída: Maria
print(usuario["cidade"])  # Saída: Palhoça
```

No exemplo acima, podemos acessar os dados diretamente pelas chaves.

Dicionários em sistemas de cadastro

Dicionários são extremamente úteis para armazenar dados de forma estruturada:

- **Facilidade de armazenamento:** Um dicionário pode representar cada usuário.
- **Busca eficiente:** Acesso direto pela chave (ex: nome do usuário).
- **Organização clara:** Informações bem estruturadas para manutenção e consulta.

Exemplo de sistema de cadastro com dicionário

```
usuarios = {
    "Marta": {"idade": 25, "cidade": "Palhoça", "profissão": "Engenheira"},
    "João": {"idade": 30, "cidade": "Florianópolis", "profissão": "Designer"},
    "Maria": {"idade": 28, "cidade": "São Paulo", "profissão": "Analista "}
}

nome_busca = input("Digite o nome do usuário para buscar: ")

usuario = usuarios.get(nome_busca)
if usuario:
    print(f"\nInformações de {nome_busca}:")
    for chave, valor in usuarios.items():
        print(f"{chave.capitalize()}: {valor}")
else:
    print("Usuário não encontrado.")
```

Conclusão

Os dicionários tornam sistemas de cadastro muito mais eficientes.

- Permitem organização clara e acessos rápidos.
- São fáceis de manipular e ideais para sistemas interativos.
- Fundamentais em aplicações como bancos de dados simples, logins e gerenciadores de usuários.

Exercício 1: Criando e Manipulando Dicionários

Descrição:
O usuário insere as informações e o programa as armazena em um dicionário.

Resolução - Exercício 1

```python
# Criando um dicionário para armazenar as informações do usuário
usuario = {}

# Coletando dados via entrada do usuário
usuario["nome"] = input("Digite seu nome: ")
usuario["idade"] = input("Digite sua idade: ")
usuario["cidade"] = input("Digite sua cidade: ")
usuario["profissão"] = input("Digite sua profissão: ")

# Exibindo os dados formatados
print("\nInformações do usuário:")
for chave, valor in usuario.items():
    print(f"{chave.capitalize()}: {valor}")
```

Exercício 2: Buscando Informações no Dicionário

Descrição:

O usuário adiciona vários cadastros e pode buscar um nome específico.

Resolução - Exercício 2

```python
# Criando um dicionário para armazenar múltiplos usuários
usuarios = {}

# Inserindo usuários via entrada do usuário
while True:
    nome = input("\nDigite o nome do usuário (ou 'sair' para finalizar): ")
    if nome.lower() == "sair":
        break
    idade = input("Digite a idade do usuário: ")
    cidade = input("Digite a cidade do usuário: ")
    profissao = input("Digite a profissão do usuário: ")

    usuarios[nome] = {"idade": idade, "cidade": cidade, "profissão": profissao}
    print(f"Usuário {nome} cadastrado com sucesso!")
```

Resolução - Exercício 2 (continuação)

```python
# Buscando um usuário pelo nome
nome_busca = input("\nDigite o nome do usuário para buscar: ")
usuario = usuarios.get(nome_busca)

# Exibindo informações ou mensagem de erro
if usuario:
    print(f"\nInformações de {nome_busca}:")
    for chave, valor in usuario.items():
        print(f"{chave.capitalize()}: {valor}")
else:
    print("Usuário não encontrado.")
```

Adicionar via input do usuário

```python
dados = {}  # Criando um dicionário vazio
chave = input("Digite a chave: ")
valor = input("Digite o valor: ")
dados[chave] = valor
print(f"Dicionário atualizado: {dados}")
```

Alterar um valor existente

```python
dados = {"nome": "Marta", "idade": 30}
print(f"Antes da alteração: {dados}")
dados["idade"] = 31  # Modificando o valor da chave 'idade'
print(f"Depois da alteração: {dados}")
```

Excluir um item

```
dados = {"nome": "Marta", "idade": 30, "cidade": "Florianópolis"}
del dados["cidade"]  # Removendo a chave 'cidade'
print(f"Depois da exclusão: {dados}")
```

Buscar um valor

```python
dados = {"nome": "Marta", "idade": 30}
chave = input("Digite a chave que deseja buscar: ")

# Retorna um valor padrão se a chave não existir
valor = dados.get(chave, "Chave não encontrada")
print(f"Resultado da busca: {valor}")
```

Exercícios - Agora é com você

1. Criar um dicionário com 3 pares chave-valor e imprimi-lo
2. Adicionar um novo elemento ao dicionário
3. Modificar um valor existente
4. Remover um item do dicionário
5. Verificar se uma chave existe no dicionário
6. Iterar sobre as chaves do dicionário
7. Iterar sobre os valores do dicionário
8. Iterar sobre os itens do dicionário

Resolução

Karina CASOLA

SENAC - FUTURO PROGRAMADOR

1. Criar e imprimir um dicionário

```
meu_dicionario = {"chave1": "valor1", "chave2": "valor2", "chave3": "valor3"}
print("Dicionário inicial:", meu_dicionario)
```

2. Adicionar um novo elemento

```
meu_dicionario["chave4"] = "valor4"
print("Após adicionar um elemento:", meu_dicionario)
```

3. Modificar um valor existente

```
meu_dicionario["chave2"] = "novo_valor2"
print("Após modificar um valor existente:", meu_dicionario)
```

4. Remover um item

```
del meu_dicionario["chave1"]
print("Após remover um item:", meu_dicionario)
```

5. Verificar existência de uma chave

```
chave_existente = "chave3" in meu_dicionario
print("A chave 'chave3' existe no dicionário?", chave_existente)
```

6. Iterar sobre chaves

```
print("Iterando sobre as chaves:")
for chave in meu_dicionario.keys():
    print(chave)
```

7. Iterar sobre valores

```
print("Iterando sobre os valores:")
for valor in meu_dicionario.values():
    print(valor)
```

8. Iterar sobre itens (chave, valor)

```
print("Iterando sobre os itens (chave, valor):")
for chave, valor in meu_dicionario.items():
    print(chave, ":", valor)
```

Enunciado

Desenvolva um programa em Python que implemente um menu interativo para gerenciar o cadastro de usuários. O programa deve permitir que o usuário execute as seguintes ações:

- Cadastrar um novo usuário: Solicitar o nome e idade do usuário e armazená-los em um dicionário, onde o nome será a chave e a idade será o valor.
- Listar todos os usuários cadastrados: Exibir os nomes e idades dos usuários armazenados no dicionário. Caso nenhum usuário esteja cadastrado, exibir uma mensagem informativa.
- Encerrar o programa: Finalizar a execução do menu.

Garanta que o programa seja funcional, incluindo o tratamento de entradas inválidas do usuário. Além disso, o menu deve ser exibido repetidamente até que o usuário escolha a opção de sair.

Código: Parte 1

```
def menu():
    usuarios = {}

    while True:
        print("\n=== Menu ===")
        print("1. Cadastrar usuário")
        print("2. Listar usuários")
        print("3. Sair")
        opcao = input("Escolha uma opção: ")

        if opcao == "1":
            nome = input("Digite o nome do usuário: ")
            idade = input("Digite a idade do usuário: ")
            usuarios[nome] = idade
            print(f"Usuário {nome} cadastrado com sucesso!")
```

Código: Parte 2

```
        elif opcao == "2":
            print("\n=== Lista de Usuários ===")
            if usuarios:
                for nome, idade in usuarios.items():
                    print(f"Nome: {nome}, Idade: {idade}")
            else:
                print("Nenhum usuário cadastrado.")

        elif opcao == "3":
            print("Saindo do programa...")
            break

        else:
            print("Opção inválida. Tente novamente.")

# Executa o menu
menu()
```

Enunciado

Amplie o programa em Python que implementa um menu interativo para gerenciar usuários. Adicione as seguintes funcionalidades:

- **Atualizar um usuário**: Permitir que o usuário selecione um nome previamente cadastrado e atualize a idade associada a esse nome no dicionário. Caso o nome não exista, exibir uma mensagem informando que o usuário não foi encontrado.
- **Deletar um usuário**: Permitir que o usuário selecione um nome previamente cadastrado e remova-o do dicionário. Caso o nome não exista, exibir uma mensagem informando que o usuário não foi encontrado.

Garanta que o programa exiba um menu com essas opções e que o tratamento de entradas inválidas seja realizado de maneira apropriada.