



Introdução à linguagem Python

Aula 4 - Tuplas, Listas e Dicionários | Parte 2

Por: Thiago Ribeiro da Silva

Estudante de Ciência da Computação (UFAL)

Sobre Dicionários

- Diferentemente das listas e tuplas, dicionários permitem que os índices (posições) dos elementos na estrutura sejam nomeados;
- Desta forma, dicionários funcionam como listas, mas com rótulos, os quais podem assumir qualquer tipo, sendo declarados entre chaves no formato *índice : valor*

```
1 #Criando dicionarios
2
3 dic1 = {"id" : 12, "nome" : "Pedro", "idade" : 29}
4 dic2 = {0 : 1, 1 : 2, 2 : 3}
5 #Dicionarios vazios
6 dic3 = {}
7 dic4 = dict()
```

Sobre Dicionários

- As maneiras de acessar informação e editar em dicionários são iguais às das outras estruturas apresentadas. O grande diferencial é que agora o id é correspondente ao que foi declarado.

```
#Acessando, editando e adicionando em dicionários

dic1 = {"id" : 12, "nome" : "Pedro", "idade" : 29}
#acessando
print(dic1["id"])
#editando
dic1["id"] = 14
print(dic1)
#adicionando
dic1["cpf"] = "123.456.789-00"
print(dic1)

⇒ 12
{'id': 14, 'nome': 'Pedro', 'idade': 29}
{'id': 14, 'nome': 'Pedro', 'idade': 29, 'cpf': '123.456.789-00'}
```



Sobre Dicionários

- Algumas das principais funções internas para manipulação de dicionários:

FUNÇÃO	DESCRIÇÃO
len(dic)	retorna a quantidade de elementos no dic
dic.items()	retorna uma lista com tuplas no formato (índice, valor)
dic.keys()	retorna uma lista com os índices do dicionário
dic.values()	retorna uma lista com os valores do dicionário

Sobre Dicionários - Exemplos

- Seguem algumas formas de percorrer um dicionário com loops

```
▶ for key, value in dic1.items():
    print(key, value)
```

```
↳ id 14
    nome Pedro
    idade 29
    cpf 123.456.789-00
    Nome None
```

```
[51] for key in dic1.keys():
    print(key, dic1[key])
```

```
id 14
nome Pedro
idade 29
cpf 123.456.789-00
Nome None
```



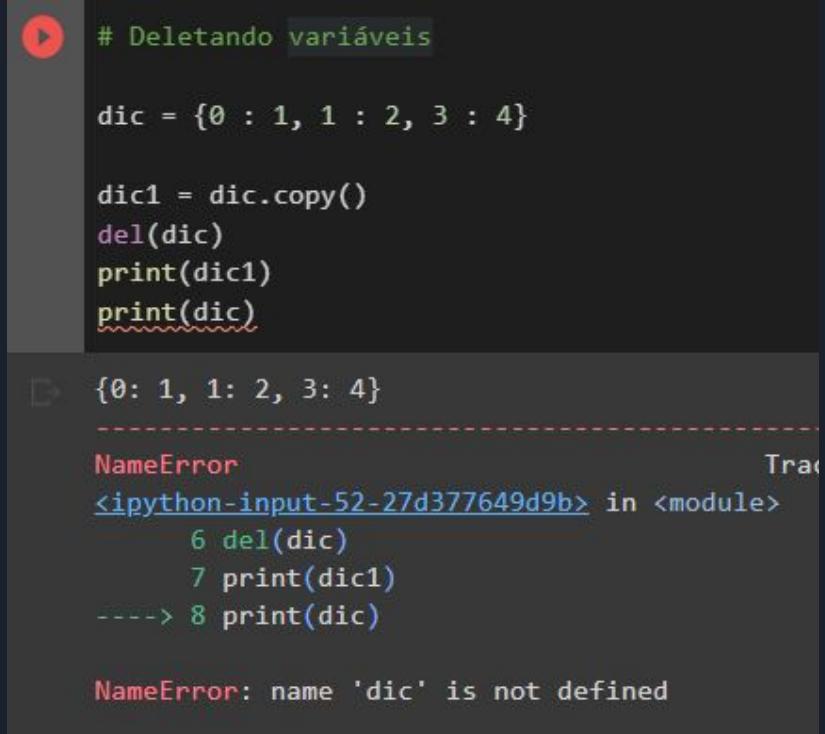
Sobre Dicionários

- Algumas das principais funções internas para manipulação de dicionários:

FUNÇÃO	DESCRIÇÃO
dic.get(índice)	retorna o valor de um determinado índice
dic.clear()	limpa todo o dicionário
dic.pop(key)	remove um item no dicionário por um índice
dic.popitem()	remove o último item do dicionário

Sobre Dicionários

- Assim como listas, dicionários são passados pelo seu endereço na memória, sendo necessário usar a função `.copy()`, para gerar um cópia e salvá-la em outra variável;
- Todas as estruturas aqui apresentadas podem ser, também, deletadas com o comando `del(variavel)`.



```
# Deletando variáveis

dic = {0 : 1, 1 : 2, 3 : 4}

dic1 = dic.copy()
del(dic)
print(dic1)
print(dic)

{0: 1, 1: 2, 3: 4}
-----
NameError: name 'dic' is not defined
```

The screenshot shows a Jupyter Notebook cell. The code defines a dictionary `dic` with keys 0, 1, and 3 mapping to values 1, 2, and 4 respectively. It then creates a copy of `dic` named `dic1`, deletes the original `dic` using the `del` keyword, and prints both `dic1` and the deleted `dic`. The output shows the copied dictionary `{0: 1, 1: 2, 3: 4}` and an error message indicating that the variable `dic` is not defined because it was deleted.

Estruturas Compostas - Exemplos



```
#Sistema de cadastro

"""

Criando um sistema de cadastro que pode cadastrar, atualizar e deletar usuários.

"""

def cadastrar(usuarios):
    pass

def cadastrar(usuarios):
    pass

def cadastrar(usuarios):
    pass
```



Estruturas Compostas - Exemplos

```
#Sistema de cadastro

#Lista de usuários
usuarios = []

#Função para cadastrar usuário
def cadastrar(usuarios):
    nome = input("Digite o nome do usuário: ")
    email = input("Digite o email do usuário (exemplo@email.com): ")
    cadastro = int(input("Digite o valor do cadastro do usuário (9999):"))
    usuarios.append({'nome': nome, 'email' : email, 'cadastro' : cadastro })
```



Estruturas Compostas - Exemplos

```
#Função para atualizar usuário
def atualizar(usuarios):
    cadastro = int(input("Informe o cadastro para atualizar dados: "))
    for i in range(usuarios):
        if usuarios[i]['cadastro'] == cadastro:
            nome = input("Digite o novo nome: ")
            email = input("Digite o novo email: ")
            usuarios[i]['nome'] = nome
            usuarios[i]['email'] = email
    return
print("Usuário não encontrado!")
return
```



Estruturas Compostas - Exemplos

```
#Função para deletar usuário
def deletar(usuarios):
    cadastro = int(input("Informe o cadastro para atualizar dados: "))
    for i in range(usuarios):
        if usuarios[i]['cadastro'] == cadastro:
            usuarios.pop(i)
            return
    print("Usuário não encontrado!")
return
```

Estruturas Compostas - Exemplos

```
# Main
print("===== SISTEMA DE CADASTRO =====")

while True:
    print("\nO que deseja fazer? ")
    print("1 - Cadastrar usuário\n2 - Atualizar usuário\n3 - Deletar usuário\n0 - Sair")
    op = int(input("Digite: "))
    if op < 0 or op > 3: continue
    if op == 1:
        cadastrar(usuarios)
    elif op == 2:
        atualizar(usuarios)
    elif op == 3:
        deletar(usuarios)
    elif op == 0:
        break

print("Até a próxima!")
```



Links para a documentação de outras estruturas compostas

- Set: <https://python-reference.readthedocs.io/en/latest/docs/sets/>
- Frozenset: <https://python-reference.readthedocs.io/en/latest/docs/functions/frozenset.html>
- Collections: <https://docs.python.org/3/library/collections.html>



Obrigado!

