Computação 1 - Python
Aula 10 - Teórica: Estrutura de Dados - Dicionário

João Carlos, Carla Delgado, Ana Luisa Duboc

Dicionário

Considere que você precisa fazer uma função que guarde o nome e o telefone de seus amigos. Sua função também deve permitir a consulta aos telefones das pessoas.

Como guardar estas informações (nome e telefone)?

Como recuperar o telefone de uma dada pessoa?

Dicionário

Dicionários são estruturas para armazenar dados. Não são sequências como strings, listas e tuplas.

São **mapeamentos** formados por pares de *chave* – *valor*.

$$\textit{Chave}1 \rightarrow \text{Conteúdo1}$$
$$\textit{Chave}2 \rightarrow \text{Conteúdo2}$$
$$\textit{Chave}3 \rightarrow \text{Conteúdo3}$$
$$\ldots$$

Representam uma coleção não ordenada de **valores** onde cada valor é referenciado através de sua **chave**.

Notação: { chave1: conteúdo1, chave2: conteúdo2, ..., chaveN: conteúdoN }

Dicionário

Dicionários são **mapeamentos** formados por pares de *chave* – *valor*.

As **chaves** funcionam como os índices de uma lista

$$\textit{Chave}1 \rightarrow \text{Conteúdo1}$$
$$\textit{Chave}2 \rightarrow \text{Conteúdo2}$$
$$\textit{Chave}3 \rightarrow \text{Conteúdo3}$$
$$\ldots$$

As **chaves de dicionários** são dados de tipo imutável, geralmente strings (podem ser tuplas ou tipos numéricos).

Os valores em um dicionário são dados quaisquer.

Dicionário

Caderno de telefones

```
>>> Caderno = {"Carlos": "2222-2223", "André": "2121-9092",
"José": "9999-9291"}

>>> Caderno["André"]
'2121-9092'

>>>Caderno["Jorge"]
Traceback (most recent call last):
File "<pyshell#8>", line 1, in <module>
Caderno["Jorge"]
KeyError: 'Jorge'

>>> "Jorge" in Caderno
False

>>> len(Caderno)
3
```

Dicionário

Podemos usar a função **dict** para definir dicionários:

- **Lista de pares (chave,valor):**
  Caderno = dict([("Carlos","2222-2223"),("André","2121-9992"),("José","9999-9291")])
- **Sequência de itens no formato chave=valor:**
  Caderno = dict(Andre="2121-9992", Jose="9999-9291", Carlos="2222-2223")

Dicionário

Caderno de telefones

```
>>> Caderno = {"Carlos": "2222-2223","André": "2121-9092","José": "9999-9291"}

>>> Caderno["José"]

>>>Caderno["José"] = "8799-0405"
>>>Caderno["José"]
```

Dicionário

Caderno de telefones

```
>>> Caderno = {"Carlos": "2222-2223","André": "2121-9092","José": "9999-9291"}

>>> Caderno["José"]
'9999-9291'

>>>Caderno["José"] = "8799-0405"

>>>Caderno["José"]
'8799-0405'
```

Dicionários são mutáveis. Mesma sintaxe da inserção! Não é inserida outra chave com o mesmo nome e valor diferente, pois chaves são únicas.

Dicionário

Caderno de telefones

```
>>> Caderno = {"Carlos": "2222-2223","André": "2121-9092","José": "9999-9291"}
>>> "Jorge" in Caderno

>>>Caderno["Jorge"] = "8586-9091"
>>>Caderno["Jorge"]

>>>Caderno
```

Dicionário

Caderno de telefones

```
>>> Caderno = {"Carlos": "2222-2223","André": "2121-9092","José": "9999-9291"}
>>> "Jorge" in Caderno
False

>>>Caderno["Jorge"] = "8586-9091"

>>>Caderno["Jorge"]
'8586-9091'

>>>Caderno
{'André': '2121-9092', 'Jorge': '8586-9091', 'José': '8799-0405',
'Carlos': '2222-2223'}
```

- **Para adicionar um novo par chave:valor**
  Perceba que, diferentemente de listas, atribuir a um elemento de um dicionário não requer que uma posição exista previamente.
- O dicionário não fornece garantia de que as chaves estarão ordenadas, mas a ordem em que os elementos aparecem não é importante, pois os valores são acessados somente através de suas respectivas chaves, e não de suas posições.

Manipulação de Dicionário

Como saber quais são as chaves e os valores de um dicionário?

- **dict.keys(<dicionário>)**: retorna a lista com todas as chaves do dicionário passado como parâmetro.
- **dict.values(<dicionário>)**: retorna a lista com todos os valores do dicionário passado como parâmetro.

```
>>> meses = {"janeiro":31, "fevereiro":28, "marco":31, "abril":30, "maio":31,
"junho":30, "julho":31, "agosto":31, "setembro":30,"outubro":31, "novembro":31,
"dezembro":31 }

>>> dict.keys(meses)
['novembro', 'marco', 'julho', 'agosto', 'fevereiro', 'junho', 'dezembro',
'janeiro', 'abril', 'maio', 'outubro', 'setembro']

>>> dict.values(meses)
[31, 31, 31, 31, 28, 30, 31, 31, 30, 31, 31, 30]
```

Manipulação de Dicionário

- **dict.items(<dicionário>)**: retorna uma lista com todos os pares (chave, conteúdo) do dicionário passado como parâmetro.

```
>>> meses = {"janeiro":31, "fevereiro":28, "marco":31,
"abril":30, "maio":31, "junho":30, "julho":31, "agosto":31,
"setembro":30, "outubro":31, "novembro":31, "dezembro":31 }

>>> dict.items(meses)
[('novembro', 31), ('marco', 31), ('julho', 31), ('agosto',
31), ('fevereiro', 28), ('junho', 30), ('dezembro', 31),
('janeiro', 31), ('abril', 30), ('maio', 31), ('outubro', 31),
('setembro', 30)]
```

Manipulação de Dicionário

- **dict.get(<dicionário>,chave,[valor de retorno])**: Retorna o valor associado com a chave.
  Se *chave* não está no dicionário e se o *valor de retorno* é especificado, retorna o valor especificado.
  Se o *valor de retorno* não é especificado, retorna **None**.

```
>>> meses = {"janeiro":31, "fevereiro":28, "marco":31,
"abril":30, "maio":31, "junho":30, "julho":31, "agosto":31,
"setembro":30, "outubro":31, "novembro":31, "dezembro":31 }

>>> dict.get(meses,"marco")
31
>>> dict.get(meses,"marco","Valor nao encontrado")
31

>>> dict.get(meses,"março")
None
>>> dict.get(meses,"março","Valor nao encontrado")
"Valor nao encontrado"
```

Manipulação de Dicionário

- **dict.clear(<dicionário>)**: apaga todos os itens do dicionário.
- **dict.copy(<dicionário>)**: cria e retorna uma cópia do dicionário. Os elementos do novo dicionário são apenas referências aos elementos do dicionário original.

```
>>> meses = {"janeiro":31, "fevereiro":28}
>>> novo=dict.copy(meses)
>>> novo
{'fevereiro': 28, 'janeiro': 31}
>>> novo["maio"]=31
>>> novo
{'maio': 31, 'fevereiro': 28, 'janeiro': 31}
>>> meses
{'fevereiro': 28, 'janeiro': 31}
>>> dict.clear(meses)
>>> meses
{}
```

Manipulação de Dicionário

Escreva uma função que receba uma frase como parâmetro e retorne um dicionário, onde cada chave seja um caracter e seu valor seja o número de vezes que o caracter aparece na frase lida.
Exemplo: "Os ratos" $\rightarrow$ {"O":1, " ":1, "s":2, "r":1, "a":1, "t":1,"o":1}
Faça uma versão usando while e outra usando for.

Manipulação de Dicionário

Escreva uma função que receba uma frase como parâmetro e retorne um dicionário, onde cada chave seja um caracter e seu valor seja o número de vezes que o caracter aparece na frase lida.
Exemplo: "Os ratos" $\rightarrow$ {"O":1, " ":1, "s":2, "r":1, "a":1, "t":1,"o":1}
Faça uma versão usando while e outra usando for.

```
# Solução com while
# str → dic
def cria_dicionario(frase):
    dicionario = { }
    i = 0
    while i < len(frase):
        if frase[i] not in dicionario:
            dicionario[frase[i]] = 1
        else:
            dicionario[frase[i]] += 1
        i += 1
    return dicionario
```

```
# Solução com for
# str → dic
def cria_dicionario(frase):
    dicionario = { }
    for c in frase:
        if c not in dicionario:
            dicionario[c] = 1
        else:
            dicionario[c] += 1
    return dicionario
```

Manipulação de Dicionário

1. Escreva uma função que transforma uma lista de tuplas em um dicionário que associa o primeiro componente de cada tupla ao segundo.

2. Escreva uma função que transforma um dicionário em uma lista de tuplas, onde as tuplas estão ordenadas pelo primeiro componente. Lembre-se de **list.sort**.

Autores

- João C. P. da Silva ▸ Lattes
- Carla Delgado ▸ Lattes
- Ana Luisa Duboc ▸ Lattes

Colaboradores

- Fabio Mascarenhas ▸ Lattes
- Anamaria Martins Moreira ▸ Lattes
- Leonardo de Oliveira Carvalho ▸ Lattes
- Charles Figueiredo de Barros ▸ Lattes
- Fabrício Firmino de Faria ▸ Lattes

Computação 1 - Python
Aula 10 - Teórica: Estrutura de Dados - Dicionário