Aula 07

# LIP

## Linguagem de Programação

Python

# Dicionários

Para as listas e tuplas é necessário um índice numérico para chegar ao item escolhido.

Já os dicionários são estruturas de chave-valor, ou seja, os valores (dados) estão sempre associados a uma chave, que pode ser números ou não.

Chaves e valores em um dicionário podem ser de diferentes tipos de dados (int, float, bool, strings, entre outros).

Python

# Dicionários

## Exemplo:

```
#criação de dicionários
dicionarioVazio={} #método explicito
print(type(dicionarioVazio))
dicionarioVazio1=dict({}) #construtor através da classe
print(type(dicionarioVazio1))
localizacao = dict({"Lat": -22.817087,"Long": -47.069750})
print(localizacao)
print(localizacao["Lat"]) #nesse caso o item é acessado pela chave "Lat"
print(localizacao["Long"]) #nesse caso o item e acessado pela "Long"
```

```
<class 'dict'>
<class 'dict'>
{'Lat': -22.817087, 'Long': -47.06975}
-22.817087
-47.06975
```

# Dicionários – lendo valores

O método get é outra forma para obter valores de um dicionário.

Ele recebe como parâmetro a chave associada ao valor desejado.

Caso a chave não seja encontrada no dicionário, será retornado None como resposta, melhor que um erro caso a chave não exista.

# Dicionários lendo valores

Python

Exemplo:

```
localizacao = {"Lat": -22.817087,"Long": -47.069750}
print(localizacao.get("Lat"))
print(localizacao.get("long")) #bem melhor procurar a chave com o método get
```

```
-22.817087
None
```

# Dicionários pertinência

Python

Exemplo:

```
localizacao = {"Lat": -22.817087,"Long": -47.069750}
if "Lat" in localizacao:
    print("V")
else:
    print("N")
if "Cidade" not in localizacao:
    print("V")
else:
    print("N")
```

V

V

# Dicionários adição de item

Python

Para adicionar uma chave e um valor:

```
localizacao = {"Lat": -22.817087,"Long": -47.069750}
localizacao["Cidade"] = "Campinas"
print(localizacao)
```

```
{'Lat': -22.817087, 'Long': -47.06975, 'Cidade': 'Campinas'}
```

# Dicionários outros métodos

Python

```
localizacao = {"Lat": -22.817087,"Long": -47.069750, "Cidade" : "Campinas", "Res" : False}
print(localizacao.items()) #retorna todos as chaves com os itens
print(localizacao.keys()) #retorna só as chaves
print(localizacao.values())#retorna os valores
print(len(localizacao))#retorna a quantidade de itens
localizacao1=localizacao.copy() #faz cópias
print(localizacao)
print(localizacao1)
print(localizacao1.pop("Res")) #remove e retorna o valor selecionado
print(localizacao)
print(localizacao1)
```

```
dict_items([('Lat', -22.817087), ('Long', -47.06975), ('Cidade', 'Campinas'), ('Res', False)])
dict_keys(['Lat', 'Long', 'Cidade', 'Res'])
dict_values([-22.817087, -47.06975, 'Campinas', False])
4
{'Lat': -22.817087, 'Long': -47.06975, 'Cidade': 'Campinas', 'Res': False}
{'Lat': -22.817087, 'Long': -47.06975, 'Cidade': 'Campinas', 'Res': False}
False
{'Lat': -22.817087, 'Long': -47.06975, 'Cidade': 'Campinas', 'Res': False}
{'Lat': -22.817087, 'Long': -47.06975, 'Cidade': 'Campinas'}
```

# Dicionários outros métodos

Python

```
localizacao = {"Lat": -22.817087,"Long": -47.069750, "Cidade" : "Campinas", "Res" : False}
print(localizacao)
del localizacao["Res"] # comando del também apaga um item do dicionário através da chave
print(localizacao)
itemApagado=(localizacao.popitem())#método popitem apaga o último item do dicionário e retorna seu valor como uma tupla
print(itemApagado)
print(localizacao)
localizacao1={"Bairro" : "Champs Elysees" , "CEP" : "04511-203"}
localizacao.update(localizacao1) #concatenar dois dicionarios
print(localizacao)
```

```
{'Lat': -22.817087, 'Long': -47.06975, 'Cidade': 'Campinas', 'Res': False}
{'Lat': -22.817087, 'Long': -47.06975, 'Cidade': 'Campinas'}
('Cidade', 'Campinas')
{'Lat': -22.817087, 'Long': -47.06975}
{'Lat': -22.817087, 'Long': -47.06975, 'Bairro': 'Champs Elysees', 'CEP': '04511-203'}
```

# Iterando sobre Dicionários

Python

```
localizacao = {"Lat": -22.817087,"Long": -47.069750, "Cidade" : "Campinas", "Res" : False}
# iterando sobre chaves
for chaves in localizacao.keys():
    print(chaves, end=" - ")
print()
# iterando sobre valores
for valor in localizacao.values():
    print(valor, end=" - ")
print()
# iterando sobre chaves e valores
for itens in localizacao.items(): #lembrando que o item retornado é um tupla
    print(itens, end=" - ")
print()
```

```
Lat - Long - Cidade - Res -
-22.817087 - -47.06975 - Campinas - False -
('Lat', -22.817087) - ('Long', -47.06975) - ('Cidade', 'Campinas') - ('Res', False) -
```

Python

# Conjuntos - set

Conjuntos são estruturas ou coleções de dados que podemos realizar operações matemáticas de união, intersecção e diferença dentre eles.

Conjuntos são mutáveis, porém, não são ordenados e também por isso não permitem dados duplicados. Dessa forma não armazenam a posição e nem a ordem de inserção.

Python

# Métodos de conjuntos

| | |
|---|---|
| add() | insere elemento no set |
| clear() | remove todos os elementos do set |
| copy() | retorna cópia do set |
| difference() | retorna um set com a diferença entre 2 ou mais sets |
| difference_update() | remove elementos incluidos no segundo set |
| discard() | remove item especificado |
| intersection() | retorna o set interseção de 2 sets |
| intersection_update() | remove items do set não presentes no segundo set especificado |
| isdisjoint() | retorna True se os 2 sets são disjuntos |
| issubset() | retorna True se o set é subconjunto do segundo set |
| issuperset() | retorna True se o set contém o segundo set |
| pop() | remove (e retorna) um elemento arbitrário do set |
| remove() | remove o elemento especificado |
| symmetric_difference() | retorna o set com a diferença simétrica de dois sets |
| symmetric_difference_update() | insere a diferença simétrica desse set em outro |
| union() | retorna um set com a união dos sets |
| update() | atualiza o primeiro set com sua união com um ou mais sets |

Python

# Quadro Resumo de estruturas de dados

| | Listas | Tuplas | Dicionários | Conjuntos |
|---|---|---|---|---|
| Ordem dos elementos | Fixa | Fixa | Mantida a partir do Python 3.7 | Indeterminada |
| Tamanho | Variável | Fixo | Variável | Variável |
| Elementos repetidos | Sim | Sim | Pode repetir valores, mas as chaves devem ser únicas | Não |
| Pesquisa | Sequencial, índice numérico | Sequencial, índice numérico | Direta por chave | Direta por valor |
| Alterações | Sim | Não | Sim | Sim |
| Uso primário | Sequências | Sequências constantes | Dados indexados por chave | Verificação de unicidade, operações com conjuntos |