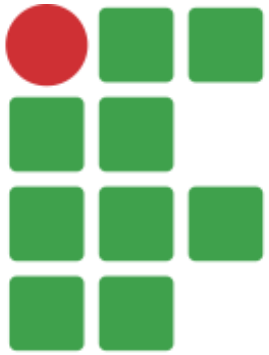

INSTITUTO FEDERAL
Paraíba

Campus Cajazeiras

ALGORITMOS E LÓGICA DE PROGRAMAÇÃO

PROFESSORES LEANDRO, MICHEL
CAJAZEIRAS / IFPB

Dicionários

OBJETIVOS

- Compreender os tipos de dados em dicionário;
- Desenvolver algoritmos utilizando Dicionários;

ROTEIRO

- Programação básica em Python (Coleções)
  - Tuplas
  - Listas
  - Dicionários
  - Conjuntos (set)

Dicionário

- Os dicionários são a implementação do Python de uma estrutura de dados, geralmente conhecida como matrizes associativas (**lista de associações compostas**).
- Podemos pensar em um dicionário como um mapeamento entre um conjunto de índices (conhecidos como **chaves**) e um conjunto de valores.
- Cada chave mapeia para um valor. A associação de uma chave e um valor é chamada de **chave: objeto** ou às vezes um **item** .

Chaves (elemento imutável)	Objetos
'nome'	"Pedro"
'idade'	25
'ocupação'	"analista"
'e-mail'	"pedro@email.com"

Dicionário

Sintaxe

dicionario = {'a': a, 'b': b, ..., 'z': z}

Você pode criar um **dicionário** colocando uma lista de **chave: objeto** separados por **vírgulas** entre chaves **{ }**. Cada chave é separada de seu valor associado por dois pontos **:**

Exemplos

main.py

```
1  #criando um dicionário vazio
2  dicionario = {}
3  print(dicionario)
4  #imprime {}
```

```
{}
>
```

main.py

```
1  #criando um dicionário
2  #registro de funcionário
3  D1 = {'nome':"pedro", 'idade':25, 'ocupacao':"analista", 'e-mail':"perdo@email.com"}
4  print(D1)
5  #imprime {'nome': 'pedro', 'idade': 25,
6  # 'ocupacao': 'analista', 'e-mail': 'perdo@email.com'}
```

```
{'nome': 'pedro', 'idade': 25, 'ocupacao': 'analista', 'e-mail': 'perdo@email.com'}
>
```

Dicionário

Exemplos

- O construtor **dict()**
- Aplicado em uma lista

Podemos converter sequências de dois valores em um dicionário com o construtor **dict()** do Python.
O primeiro item em cada sequência é usado como chave e o segundo como valor.

main.py

```
1   # Crie um dicionário com uma
2   # lista de tuplas de dois itens
3
4   L1 = [('nome', "Davi"),
5         ('idade', 8),
6         ('ocupação', "Estudante")]
7
8   D1 = dict(L1)
9   print(D1)
10  #imprime {'nome': 'Davi', 'idade': 8, 'ocupação': 'Estudante'}
11
```

```
{'nome': 'Davi', 'idade': 8, 'ocupação': 'Estudante'}
>
```

Dicionário

Exemplos

- O construtor **dict()**
- Aplicado em uma tupla

main.py

```
1   # Crie um dicionário com uma tupla
2   #  de listas de dois itens
3
4   T1 = (['nome', "Davi"],
5         ['idade', 8],
6         ['ocupação', "Estudante"])
7
8   D1 = dict(T1)
9   print(D1)
10  #imprime {'nome': 'Davi', 'idade': 8, 'ocupação': 'Estudante'}
```

```
{'nome': 'Davi', 'idade': 8, 'ocupação': 'Estudante'}
>
```

Dicionário

Exemplos

Função **zip()**

A função **zip()** combina itens de cada um dos iteráveis especificados .

O valor de retorno é uma lista de tuplas onde os itens de cada iterável passado no mesmo índice são pareados.

main.py

```
1  # Combine duas listas juntas
2
3  x = [1, 2, 3]
4  y = ["um", "dois", "três"]
5  combina = zip(x, y)
6  # usando o list para converte
7  print(list(combina))
8  # imprime [(1, 'um'), (2, 'dois'), (3, 'três')]
```

```
[(1, 'um'), (2, 'dois'), (3, 'três')]
>
```

Dicionário

Exemplos

Função **zip()**

Descompacte / descompacte itens compactados

zip() em conjunto com o operador ***** pode ser usado para descompactar uma lista:

main.py

```
1   # compactar duas listas
2   x = [1, 2, 3]
3   y = ["um", "dois", "três"]
4   combina = zip(x, y)
5
6   # descompactar
7   a, b = zip(*combina)
8
9   print(a)
10  # imprime (1, 2, 3)
11
12  print(b)
13  # imprime ('um', 'dois', 'três')
```

```
(1, 2, 3)
('um', 'dois', 'três')
>
```

Dicionário

Exemplos

- O construtor **dict()**

main.py

```python
# Crie um dicionário com uma lista de
# chaves / valores compactados
chaves = ['nome', 'idade', 'ocupação']
valores = ["Davi", "8", "Estudante"]

D1 = dict(zip(chaves, valores))

print(D1)
# imprime {'nome': 'Davi', 'idade': '8', 'ocupação': 'Estudante'}

```

```
{'nome': 'Davi', 'idade': '8', 'ocupação': 'Estudante'}
>
```

Dicionário

Exemplos

Propriedades importantes de um dicionário

As chaves devem ser exclusivas:

Uma chave pode aparecer em um dicionário apenas uma vez.

Mesmo que seja especificado uma chave mais de uma vez durante a criação de um dicionário, o último valor dessa chave se torna o valor associado.

main.py

```
1  # chaves devem ser exclusivas
2  D1 = {'nome':"pedro",
3        'idade':25,
4        'nome':"Davi"}
5
6  print(D1)
7  # imprime {'nome': 'Davi', 'idade': 25}
```

```
{'nome': 'Davi', 'idade': 25}
>
```

A chave deve ser do tipo imutável:

```
D = {(2,2): 32,
     True: 'a',
     'name': "Davi"}
```

O valor pode ser de qualquer tipo:

```
D = {'a':[1,2,3],
     'b':{1,2,3}}
```

Dicionário

Exemplos

Adicionar ou atualizar itens do dicionário

Basta referir-se ao item por sua chave e atribuir um valor.
Se a chave já estiver presente no dicionário, seu valor será substituído pelo novo.

main.py

```python
# add ou atualizar itens
D1 = {'nome':"Davi",
      'idade':8}
D1['nome'] = "Pedro" # chave já existe
D1['ocupação'] = "Estudante" # chave nova

print(D1)
#imprime {'nome': 'Pedro', 'idade': 8, 'ocupação': 'Estudante'}
```

```
{'nome': 'Pedro', 'idade': 8, 'ocupação': 'Estudante'}
```

Dicionário

Exemplos

Métodos de Dicionário

A tabela abaixo mostra um resumo dos principais métodos para manipulação dos Dicionário em Python.

D = {'nome': "Davi", 'idade': 8}

método	Descrição	Exemplo
clear()	remover todos os itens do dicionário	D.**clear()** print(D) # imprime { }
copy()	retorna a cópia do dicionário	X = D.copy() print(X) # imprime {'nome': "Davi", 'idade': 8 }
pop() **popitem()**	remover uma **chave** do dicionário. O popitem() remover o último item.	D.pop('idade') ou D. popitem() print(D) # imprime {'nome': "Davi"}

setdefault()	retorna o valor para a chave se a chave estiver no dicionário . Caso contrário, ele insere a chave com um valor padrão e retorna o padrão	v = D.setdefault('ocupação', "Estudante")
		print(D)
		# imprime {'nome': "Davi", 'idade': 8, 'ocupação': 'Estudante'}
		print(v)
		# imprime Estudante

Exercícios A

1 - Verificar se um determinado valor existe em um dicionário.
Obs: podemos usar o método values(), que retorna os valores como uma lista e, em seguida, usar o operador in.

2- Obtenha a chave de um valor mínimo do seguinte dicionário

```
d1 = { 'física': 82, 'Matemática': 65, 'historia': 75 }
```

3 - Fazer um dicionário e adicionar 3 disciplinas com sua respectiva nota (ex: {'algoritmos': 90, 'física': 80, 'matemática': 70}), depois criar uma estrutura que encontre dentro do dicionário qual foi a menor nota.

Exercícios B

4 - Crie uma estrutura de repetição para fazer a leitura de vários números inteiros e ímpares e os armazene dentro de uma lista, quando essa lista chegar a 5 elementos pare com a leitura. Em seguida, converta essa lista em uma tupla e crie outra estrutura de repetição para calcular a média entre todos os valores dentro da tupla.

5 - Crie um dicionário para armazenar o nome do aluno (chave) e 4 notas (elementos) para 3 alunos, fazendo a leitura dos valores por meio de uma estrutura de repetição. Depois, crie uma nova estrutura de repetição para somar todas as notas e retornar a média.