

Workshop Estrutura de dados em Python

Sumário

- Tupla
- Lista
- Conjunto
- Dicionário
- del e in

Tupla

<class 'tuple'>

Consiste em uma sequência de valores separados por vírgulas, envolvidas por parênteses. São imutáveis, e usualmente contém uma sequência heterogênea de elementos.

```
tupla = ("item0", "item1", "item2")
```

Informações gerais - Tupla

- Podem ser usadas para casos em que seja necessária uma sequência imutável de dados.
- Na criação, tuplas podem ser envolvidas ou não por parênteses.
- Os elementos são acessados via desempacotamento ou índice. Sendo possível fazer o fatiamento (tupla[início:fim]).
- Tuplas vazias são construídas por um par de parênteses vazios.
- Uma tupla unitária é construída por um único valor e uma vírgula entre parênteses.
- É possível criar tuplas com elementos mutáveis (lista).

Métodos - Tupla

- `tupla.count()`
Retorna o número de vezes que um valor especificado ocorre em uma tupla.

- `tupla.index()`
Procura um valor especificado em uma tupla e retorna a posição do primeiro valor achado.

Lista

<class 'list'>

As listas são coleções ordenadas de valores. Os elementos da lista são ordenados, alteráveis e permitem valores duplicados. São separados por vírgulas entre colchetes.

```
lista = ["item0", "item1", "item2"]
```

Informações gerais - Lista

- Os elementos são acessados via desempacotamento ou índice. Sendo possível fazer o fatiamento (lista[início:fim]).
- Listas aceitam concatenação.
- Listas vazias são construídas por um par de colchetes vazios.
- Listas aceitam aninhamento.

Métodos - Lista

- `list.append(x)`

Adiciona um item ao fim da lista.

- `list.extend(iterable)`

Prolonga a lista, adicionando no fim todos os elementos do argumento *iterable* passado como parâmetro.

- `list.insert(i, x)`

Adiciona um elemento a uma posição especificada

- `list.remove(x)`

Remove o primeiro item encontrado na lista cujo valor é igual a `x`. Se não existir valor igual, uma exceção `ValueError` é levantada.

- `list.pop([i])`

Remove um item em uma dada posição na lista e o retorna. Caso não seja colocado valor de `index`, será removido o último.

- `list.clear()`

Remove todos os itens de uma lista.

Métodos - Lista

- `list.index(x[, start[, end]])`

Devolve o índice base-zero do primeiro item cujo valor é igual a x, levantando `ValueError` se este valor não existe.

- `list.count(x)`

Devolve o número de vezes em que x aparece na lista.

- `list.sort(key=None, reverse=False)`

Ordena os itens na lista.

- `list.copy()`

Devolve uma cópia da lista.

- `list.reverse()`

Inverte a ordem dos elementos na lista.

Conjunto

<class 'set'>

É uma coleção desordenada de elementos, sem elementos repetidos e não indexados. Seus valores são separados por vírgula entre chaves.

```
conjunto= {"iten0", "iten1", "iten2"}
```

Informações gerais - Conjunto

- Usos comuns para conjuntos incluem a verificação eficiente da existência de objetos e a eliminação de itens duplicados.
- Suportam operações matemáticas como união, interseção, diferença e diferença simétrica.
- Chaves ou a função `set()` podem ser usados para criar conjuntos.
- Para criar um conjunto vazio é preciso usar a `set()`.
- Depois de criado, não pode é possível alterar seus itens, mas pode-se remover itens e adicionar novos itens.

Métodos - Conjunto

- `set.add()`

Adiciona um determinado valor a um conjunto.

- `set.update()`

Adiciona itens de qualquer objeto iterável(tuplas, listas, dicionários, conjuntos) ao conjunto.

- `set.copy()`

Retorna uma cópia do conjunto.

- `set.remove()`

Remove um determinado elemento do conjunto, caso não seja encontrado retornará um erro.

- `set.discard()`

Remove um determinado elemento do conjunto, caso não seja encontrado não retornará um erro.

- `set.pop()`

Remove um elemento aleatório do conjunto.

- `set.clear()`

Remove os itens do conjunto.

Métodos - Conjunto

- `set.difference()`

Retorna um conjunto contendo a diferença entre dois ou mais conjuntos.

- `set.difference_update()`

Remove os itens do conjunto que também pertence a outro especificado.

- `set.intersection()`

Retorna um conjunto com o valores da interseção de outros dois conjuntos.

- `set.intersection_update()`

Remove itens deste conjunto que não pertence a outro(s) conjunto(s) especificado(s).

- `set.isdisjoint()`

Retorna se dois conjuntos possuem interseção ou não.

- `set.issubset()`

Retorna se outro conjunto contém este conjunto.

- `set.issuperset()`

Retorna se este conjunto contém outro conjunto.

- `set.symmetric_difference()`

Retorna um conjunto com a diferença simétrica entre dois conjuntos.

Métodos - Conjunto

- `set.symmetric_difference_update()`
Insere a diferença simétrica entre este e outro conjunto neste conjunto.

- `set.union()`
Retorna um conjunto que contém a união entre conjuntos.

Dicionário

<class 'dict'>

É uma coleção ordenada, modificável e não permite chaves com o mesmo valor. São indexados por chaves (keys), que podem ser de qualquer tipo imutável (como strings e inteiros) e ficam entre chaves, separados por vírgulas.

Dicionario = {"chave0": "item0", "chave1": "item1", "chave2": "item2"}

Informações gerais - Dicionário

- As principais operações em um dicionário são armazenar e recuperar valores a partir de chaves.
- Para acessar os itens do dicionário, utiliza-se a chave entre colchetes.
- Ao armazenar um valor utilizando uma chave já presente, o antigo valor será substituído pelo novo.
- Para criar um dicionário pode se usar uma lista de pares chave: valores separados por vírgula com chaves: {'jack': 4098, 'sjoerd': 4127} ou {4098: 'jack', 4127: 'sjoerd'} ou usar o construtor de tipo: dict().

Métodos - Dicionário

- `dict.values()`

Retorna uma lista com todos os valores de um dicionário.

- `dict.copy()`

Retorna uma cópia do dicionário.

- `dict.fromkeys()`

Retorna um dicionário com as chaves e os valores especificados.

- `dict.get()`

Retorna o valor de uma chave especificada.

- `dict.items()`

Retorna uma lista contendo uma tupla para cada par chave-valor.

- `dict.keys()`

Retorna uma lista contendo as chaves de um dicionário.

- `dict.pop()`

Remove um elemento com uma chave especificada.

- `dict.popitem()`

Remove o último par chave-valor inserido.

Métodos - Dicionário

- `dict.setdefault()`

Retorna um valor de uma chave específica. Se a chave não existir.

- `dict.update()`

Atualiza o dicionário com os pares de valores-chave especificados.

- `dict.clear()`

Remove todos os elementos do dicionário.




del e in

del

- Tupla: Para apagar a tupla.
- Lista: Para remover um ou uma faixa de itens ou excluir a lista toda.
- Conjunto: Para apagar o conjunto.
- Dicionário: Para remover um par chave: valor determinado ou apagar o dicionário.



in

- A keyword “in” pode ser usada para saber se algum elemento pertence a lista ou a tupla ou ao conjunto. No dicionário pode ser buscado a chave.
- 

Exercício: Lista

- Crie uma lista contendo todos os números primos entre 1 e 100. Em seguida, faça outra lista com a média entre os pares consecutivos, dois a dois, sem levar em consideração o primeiro elemento. Finalize com uma matriz quadrada com os elementos das posições 1 a 9 da segunda lista.

Exercício: Lista

Saída:

Primeira lista - [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]

Segunda lista - [4.0, 9.0, 15.0, 21.0, 30.0, 39.0, 45.0, 56.0, 64.0, 72.0, 81.0, 93.0]

Matriz quadrada:

[9.0, 15.0, 21.0]

[30.0, 39.0, 45.0]

[56.0, 64.0, 72.0]

Exercício: Dicionário

- Mateus possui um carro e uma moto.
- Pedro possui uma bicicleta e uma moto.
- Maria possui uma bicicleta e um carro.
- José possui uma moto e um carro.
- Tiago possui um carro e uma bicicleta.
- Faça um dicionário com as informações e retorne a quantidade de carros, bicicletas e motos. Em seguida troque: carro por 50, moto por 25 e bicicleta por 10. Retorne ao dicionário modificado e a soma dos valores de cada pessoa.

Exercício: Dicionário

Saída:

Carros: 4

Bicicletas: 3

Motos: 3

{'mateus': [50, 25], 'pedro': [10, 25], 'maria': [10, 50], 'jose': [25, 50], 'tiago': [50, 10]}

A soma de Mateus é 75

A soma de Pedro é 35

A soma de Maria é 60

A soma de José é 75

A soma de Tiago é 60

Exercício: Dicionário

- Inicialmente será fornecido a quantidade n de seções do supermercado. Logo em seguida, serão apresentadas as n seções, em que, para cada seção fornecida, haverá na linha seguinte os itens presentes nela. Os itens serão todos fornecidos na mesma linha e separados por um espaço em branco. Após a apresentação das seções e de seus itens presentes no supermercado, será digitado, nas demais linhas, o conteúdo presente na lista de compras. Em cada uma das linhas conterá a seção e o item desejado, separados apenas por vírgula. A entrada termina quando for digitado FIM. Caso a seção exista no supermercado e o item desejado esteja presente nela, imprima na tela a mensagem Tem. Caso a seção não exista ou o item esteja em falta na seção, imprima na tela a mensagem Indisponível.



**Centro de
Informática**
UFPE



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

