

Fundamentos de Programação de Computadores

Python - listas, tuplos

Docente: Fátima Leal



Python

- Na última aulas vimos:
- **Ciclos**
- **strings**

Python - listas

- Sequências ordenadas
- Pode conter elementos repetidos
- Podem conter elementos de quaisquer tipos
- Os elementos são identificados pelos índices
- Pode ser uma lista vazia: []

[e1, e2, ..., en]

Python - listas

- Tamanho

```
>>> len([1,'dois',3])  
3
```

- Concatenação

```
>>> [1,'dois',3]+[4,5,6]  
[1,'dois',3,4,5,6]
```

- Repetição

```
>>> 2*[1,'dois',3]  
[1,'dois',3,1,'dois',3]
```

Python - listas

- Verificar se existe algum elemento

```
>>> 3 in [1,'dois',3]
True
```

- Iteração

```
>>> for x in [1,'dois',3]:
    print(x)
```

Python – Acesso a elementos de uma Lista

- Utilizar o índice: lista [i]
- Índices entre 0 e len(lista)-1
- Índices negativos: acesso a partir do fim
- Índices inválidos dão um erro de execução

```
>>> alimentos = ['pão', 'pão', 'queijo', 'queijo']
>>> alimentos[0]
'pão'
>>> alimentos[1]
'pão'
>>> alimentos[2]
'queijo'
>>> len(alimentos)
4
```

Python – Fatias de elementos de uma lista

- `lst[i:j]` elementos entre `i` e `j - 1` inclusíve
- `lst[i:]` elementos entre `i` até ao final
- `lst[:j]` elementos do primeiro até `j - 1` inclusíve
- `lst[:]` todos os elementos (cópia da lista)

```
>>> vogais = ['a','e','i','o','u']
>>> vogais[1:4]
['e', 'i', 'o']
>>> vogais[:3]
['a','e','i']
>>> vogais[3:]
['o','u']
>>> vogais[:]
['a', 'e', 'i', 'o', 'u']
```

Python – Percorrer elementos de uma lista

- Evita manipular explicitamente o índice
- Preferível quando necessitamos dos valores mas não dos índices

```
for valor in lista:  
    print(valor)
```

- Ou então usando os índices:

```
for i in range(len(lista)):  
    print(i, lista[i])
```

Python – Modificar elementos de uma lista

- **Podemos modificar** ou adicionar elementos a uma lista

```
>>> beatles = [1, 2, 3]
>>> beatles[0] = "john"
>>> beatles[2] = "ringo"
>>> beatles
['john', 2, 'ringo']

>>> beatles[1:2] = ['paul', 'george']
>>> beatles
['john', 'paul', 'george', 'ringo']
```

Python – Remover elementos de uma lista

```
>>> beatles = ['john', 'paul', 'george', 'ringo']
>>> del beatles[0]
>>> beatles
['paul', 'george', 'ringo']
```

■ ou

```
>>> beatles = ['john', 'paul', 'george', 'ringo']
>>> beatles[0:1] = []
>>> beatles
['paul', 'george', 'ringo']
```

Python – Métodos sobre listas

- `append` acrescentar um elemento ao final
- `insert` acrescentar um elemento numa posição
- `remove` remover um elemento
- `sort` ordenar os elementos por ordem crescente

`lista.método (argumentos)`

Python – Métodos sobre listas

```
>>> beatles = ['john', 'paul']
>>> beatles.append('george')
>>> beatles.append('ringo')
>>> beatles
['john', 'paul', 'george', 'ringo']
>>> beatles.insert(0, 'paul')
>>> beatles
['paul', 'john', 'paul', 'george', 'ringo']
>>> beatles.sort()
>>> beatles
['george', 'john', 'paul', 'paul', 'ringo']
```

Exercício

- Considera a seguinte lista [1, 2, 3, 4, 5]
 - Qual o tamanho da lista
 - Imprime a segunda posição da lista
 - Altera a segunda posição para o valor 4
 - Acrescenta o valor 10 na última posição
 - Soma todos os elementos da lista
 - Soma apenas os valores maiores que 3
 - Calcula a média da lista
 - Implementa a função `verificaSeExiste(lista, num)` que verifica se um número consta na lista. A função deverá retornar `True` ou `False`

Python – Tuplos

- Sequências ordenadas de elementos:

$(e1, e2, \dots, en)$

- Acesso aos elementos por índices
- Ao contrário das listas, os tuplos são imutáveis

Python – Operações com Tuplos

■ Tamanho

```
>>> len(('Pedro',12))  
2
```

■ Concatenação

```
>>> ('Pedro',12)+('João',14)  
('Pedro',12,'João',14)
```

■ Repetição

```
>>> 2*('Pedro',12)  
('Pedro',12,'Pedro',12)
```

Pertença

```
>>> 12 in ('Pedro',12)  
True
```

Iteração

```
>>> for x in ('Pedro',12):  
        print(x)
```

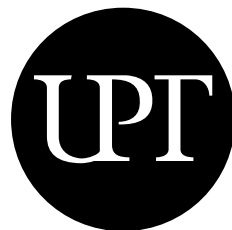
```
Pedro  
12
```

Python – Acesso a elementos de Tuplos

```
>>> nota = ('Pedro', 12)
>>> nota[0]
'Pedro'
>>> nota[1]
12
>>> nota[0] = 'Joao'
TypeError: 'tuple' object does not support item assignment
```

Python – Usar listas ou tuplos?

- Listas → sequências mutáveis
- Tuplos → sequências imutáveis
- Os tuplos são necessários em casos especiais:
 - chaves de dicionários — próximas aulas
- À semelhança dos outros tipos de dados as listas e os tuplos também permitem conversão:
 - `list(...)` converte para lista
 - `tuple(...)` converte para tuplo



UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.