Funções/Métodos das Listas

Prof. Alberto Costa Neto

Programação em Python

Métodos (funções) de Listas

```
>>> x = list()
>>> type(x)
<class 'list'>
>>> dir(x)
['append', 'clear', 'copy', 'count', 'extend',
'index', 'insert', 'pop', 'remove', 'reverse',
'sort']
>>>
```

http://docs.python.org/tutorial/datastructures.html

Contar ocorrências de um valor

```
>>> nums = [1, 3, 5, 3, 5, 5, 3, 8]
>>> print(nums.count(1))
1
>>> print(nums.count(5))
3
>>> print(nums.count(0))
0
```

Com count é possível saber quantas vezes o valor passado como parâmetro ocorre na lista

Em qual posição (índice) está um valor na lista?

- Podemos obter a posição na lista de um certo valor usando index
- Quando o valor não é encontrado, index lança uma exceção ValueError

```
>>> nums = [1, 9, 21, 10, 16]
>>> print(nums.index(9))
1
>>> print(nums.index(10))
3
>>> print(nums.index(20))
Traceback (most recent call last):
  File "<stdin>", line 1, in
<module>
ValueError: 20 is not in list
>>>
```

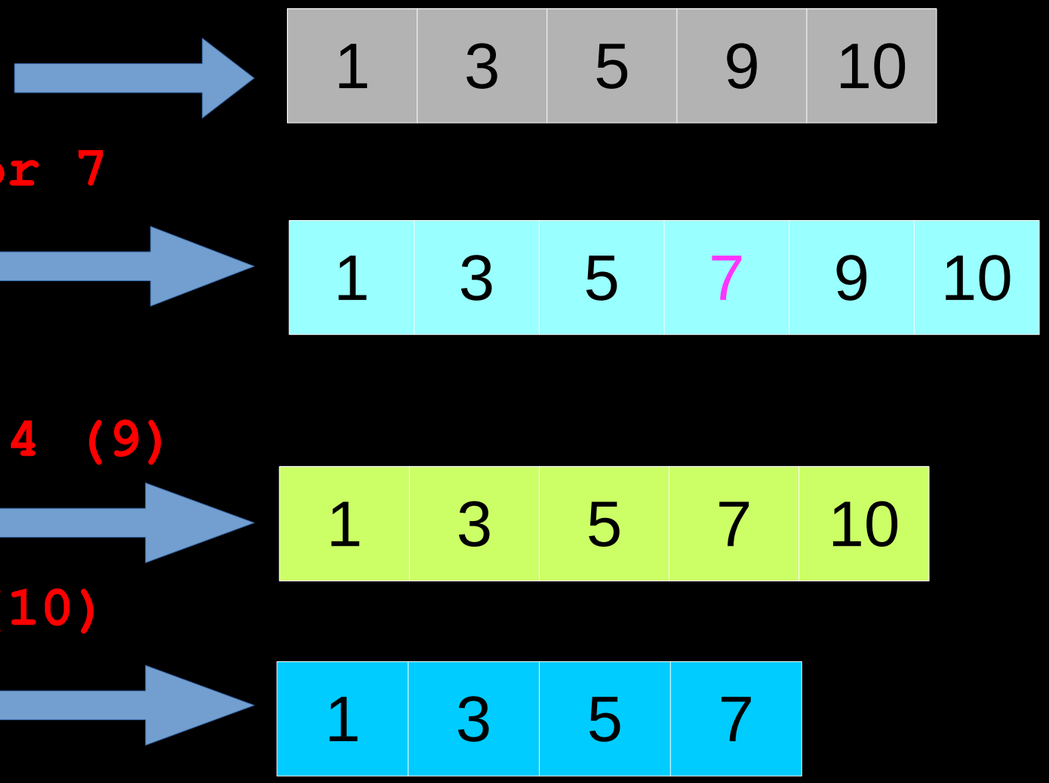
Como lidar com o ValueError?

- Quando não se tem certeza se o valor está na lista, há 2 opções:
  - Usar o in ou count
  - Usar try/except

```
>>> nums = [1, 9, 21, 10, 16]
>>> if 20 in nums:
...     print(nums.index(20))
... else:
...     print('Não encontrado')
...
Não encontrado
>>> try:
...     print(nums.index(20))
... except:
...     print('Não encontrado')
...
Não encontrado
```

Inserindo e removendo valores

```
>>> nums = [1, 3, 5, 9, 10]
>>> # insere na posição 3 o valor 7
... nums.insert(3, 7)
>>> print(nums)
[1, 3, 5, 7, 9, 10]
>>> # remove o valor da posição 4 (9)
... v1 = nums.pop(4)
>>> # remove da última posição (10)
... v2 = nums.pop()
>>> print(v1, v2, nums)
9 10 [1, 3, 5, 7]
>>> nums.clear() # limpa a lista
>>> print(nums)
[]
```



Ordenar valores de uma Lista

- Uma lista guarda os elementos e mantém na mesma ordem em que foram inseridos ou atribuídos
- Uma lista pode ser ordenada (mudando sua ordem)
- O método sort (diferente das strings) significa “ordene a si mesmo”

```
>>> amigos = ['Pedro','Jose','Maria']
>>> amigos.sort()
>>> print(amigos)
['Jose', 'Maria', 'Pedro']
>>> print(amigos[1])
Maria
>>> amigos.sort(reverse=True)
>>> print(amigos)
['Pedro', 'Maria', 'Jose']
```

Cópia e inversão de uma lista

```
>>> nums = [1, 3, 5, 7, 9, 10]
>>> nums2 = nums
>>> nums3 = nums.copy()
>>> print(nums)
[1, 3, 5, 7, 9, 10]
>>> nums2.append(11)
>>> print(nums)
[1, 3, 5, 7, 9, 10, 11]
>>> nums3.reverse()
>>> print(nums3)
[10, 9, 7, 5, 3, 1]
>>> print(nums)
[1, 3, 5, 7, 9, 10, 11]
```

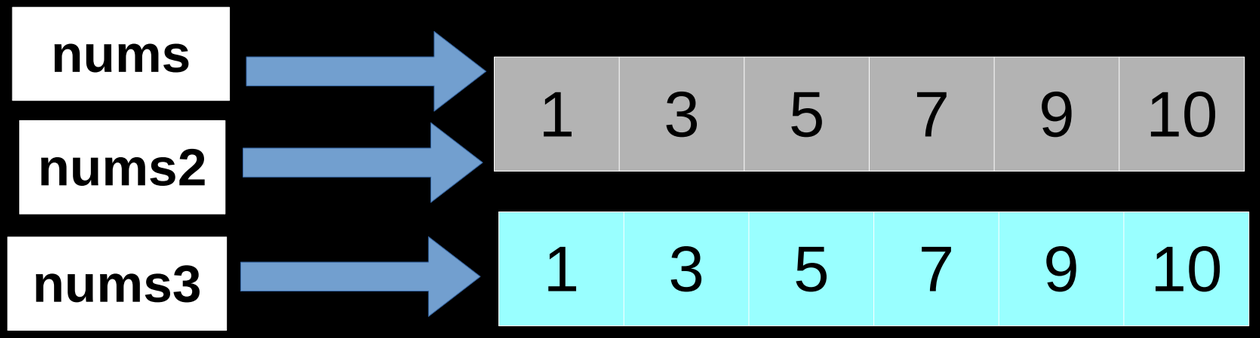
- Atribuir uma lista a outra variável é diferente de usar copy
- Ao usar reverse, a própria lista é alterada e contém os elementos na ordem reserva

Cópia e inversão de uma lista:

Explicando Melhor

```
>>> nums = [1, 3, 5, 7, 9, 10]
>>> nums2 = nums
>>> nums3 = nums.copy()
>>> print(nums)
[1, 3, 5, 7, 9, 10]
>>> nums2.append(11)
>>> print(nums)
```

Qual será o resultado?



Cópia e inversão de uma lista:

Explicando Melhor

```
>>> nums = [1, 3, 5, 7, 9, 10]
>>> nums2 = nums
>>> nums3 = nums.copy()
>>> print(nums)
[1, 3, 5, 7, 9, 10]
>>> nums2.append(11)
>>> print(nums)
[1, 3, 5, 7, 9, 10, 11]
>>> nums3.reverse()
>>> print(nums3)
[10, 9, 7, 5, 3, 1]
>>> print(nums)
[1, 3, 5, 7, 9, 10, 11]
```