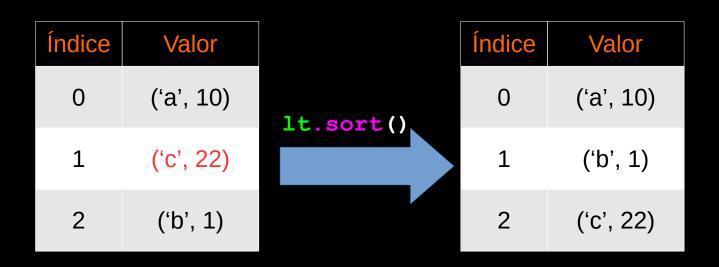
### Aplicações de Tuplas

Prof. Alberto Costa Neto Programação em Python

#### Ordenando Listas de Tuplas

- Podemos tirar vantagem da habilidade de ordenar uma lista de tuplas para obter uma versão ordenada de um dicionário
- Primeiro, ordenamos o dicionário pela chave usando a função (método) sort()

```
>>> d = {'a':10, 'b':1, 'c':22}
>>> lt = list(d.items())
>>> lt
[('a', 10), ('c', 22), ('b', 1)]
>>> lt.sort()
>>> lt
[(('a', 10), ('b', 1), ('c', 22)]
```



# Usando sorted()

Podemos fazer isto de uma forma ainda mais direta usando a função built-in sorted que recebe uma seqüência como parâmetro e retorna a seqüência ordenada

```
>>> d = \{ b':1, a':10, c':22 \}
>>> lst ord = sorted(d.items())
>>> lst ord
[('a', 10), ('b', 1), ('c', 22)]
>>> for c,v in lst ord:
... print(c, v)
a 10
b 1
c 22
```

## Ordenação pelos valores de um dicionário

- Se pudéssemos construir uma lista de tuplas da forma (valor, chave), poderíamos ordenar pelo valor
- Faremos isso com um laço for que cria uma lista de tuplas

```
>>> d = \{ 'b':1, 'a':10, 'c':22 \}
>>> temp = list()
>>> for c,v in d.items() :
temp.append( (v,c) )
>>> print(temp)
[(10, 'a'), (22, 'c'), (1, 'b')]
>>> temp.sort()
>>> print(temp)
[(1, 'b'), (10, 'a'), (22, 'c')]
```

## Ordenação decrescente pelos valores de um dicionário

 Basta usar o parâmetro reverse com valor True na chamada a sort

```
>>> d = \{ b':1, a':10, c':22 \}
>>> temp = list()
>>> for c,v in d.items() :
temp.append((v,c))
>>> print(temp)
[(10, 'a'), (22, 'c'), (1, 'b')]
>>> temp.sort(reverse=True)
>>> print(temp)
[(22, 'c'), (10, 'a'), (1, 'b')]
```

#### Uma versão ainda mais curta

```
>>> d = {'a':10, 'b':1, 'c':22}
>>> print(sorted([ (v,c) for c,v in d.items() ] ))
[(1, 'b'), (10, 'a'), (22, 'c')]
```

Compreensão de Listas cria uma lista dinâmica. Neste caso, criamos uma lista invertida das tuplas dos pares (chave, valor) de um dicionário e então a ordenamos (pelo valor e não pela chave).