Nesta aula…

Conteúdo

1 Listas

Listas

- sequências ordenadas de valores
- elementos identificados por índices
- podem conter valores de qualquer tipo

Exemplo: lista de alimentos

```
>>> alimentos = ["pao", "queijo", "manteiga"]
>>> alimentos[0]
'pao'
>>> alimentos[1]
'queijo'
>>> alimentos[-1]
'manteiga'
>>> len(alimentos)
3
```

Listas por extensão

- `[e1, e2, ..., en]`: lista com n elementos
- podem ocorrer elementos repetidos
- a ordem é significativa
- `[]` é a *lista vazia*

Intervalos

- `range(n)`: lista dos inteiros de 0 até $n-1$
- *índices* de uma sequência com n elementos
- variantes: `range(i,j)`, `range(i,j,k)`

```
>>> range(5)
[0, 1, 2, 3, 4]

>>> range(1, 5)
[1, 2, 3, 4]

>>> range(1,10,2)
[1, 3, 5, 7, 9]
```

Acesso aos elementos

- operador de indexação: `lista[i]`
- índices entre 0 e `len(lista)-1`
- tentativa de acesso para além do fim/princípio: erro
- fatias: `lista[i:j]`, `lista[i:]`, `lista[:j]`

1.1 Percorrer uma lista

Percorrer todos os elementos
Usando um ciclo `while`:

```
i = 0
while i < len(lista):
  valor = lista[i]
  print valor
  i = i+1
```

Percorrer todos os elementos
Usando um ciclo `for`:

```
for valor in lista:
  print valor
```

1.2 Operações com listas

Operações com listas

+ concatenação

n***** repete n vezes (número natural)

```
>>> a = [1, 2, 3]
>>> b = [4, 5, 6]
>>> a + b
[1, 2, 3, 4, 5, 6]
>>> 3*a
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

Listas são mutáveis

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

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

Remover elementos duma lista

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

Alternativa:

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

Referências e valores (1)

Dois nomes, dois objectos:

```
>>> a = [1,2,3]
>>> b = [1,2,3]
>>> a[0] = 'oops'
>>> print a, b
['oops', 2, 3] [1, 2, 3]
```

Referências e valores (2)

Dois nomes, um objecto:

```
>>> a = [1,2,3]
>>> b = a
>>> a[0] = 'oops'
>>> print a, b
['oops', 2, 3] ['oops', 2, 3]
```

Métodos sobre listas

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

>>> help(list)    # para mais informação
```

Listas embricadas

- listas podem conter outras listas...
- servem para representar tabelas ou matrizes

```
>>> matriz = [[1,2,-1],
              [3,1,0],
              [0,1,-2]]
>>> matriz[1][0]
3
>>> matriz[1][0] = -3
>>> matriz
[[1, 2, -1], [-3, 1, 0], [0, 1, -2]]
```

2 Tuplos

Tuplos

- sequências ordenadas de valores
- acesso aos valores por índices
- agrupar várias características numa só entidade

Exemplo: elemento químico = (símbolo, peso atómico)

```
>>> hidrogenio = ('H', 1)
>>> prata = ('Ag', 47)
>>> hidrogenio
('H', 1)
>>> prata[0]
'Ag'
>>> prata[1]
47
```

2.1 Operações sobre tuplos

Operações sobre tuplos

Operadores + e $*$ análogos aos de listas:

```
>>> prata = ('Ag', 47)
>>> ouro = ('Au', 79)
>>> ouro + prata
('Au', 79, 'Ag', 47)
>>> 3*ouro
('Au', 79, 'Au', 79, 'Au', 79)
```

NB: os resultado não fazem sentido como elementos químicos...

Tuplos são imutáveis

```
>>> chumbo = ('Pb', 82)
>>> chumbo[0] = 'Au'
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
TypeError: object does not support item assignment
```

Atribuição de tuplos

```
>>> (x,y) = (5,-7)
>>> x
5
>>> y
-7
```

Ou simplesmente:

```
>>> x,y = 5,-7
>>> x
5
>>> y
-7
```

Usar listas ou tuplos?

Listas	Tuplos
número *variável* de elementos	número *fixo* de elementos
elementos *mutáveis*	elementos *imutáveis*[1]

Exemplo 1: ponto no plano

Par com as coordenadas x, y:

```
>>> ponto = (2,-1)

>>> ponto[0]   # coordenada x
2
>>> ponto[1]   # coordenada y
-1
```

Exemplo 2: um polígono

Lista ordenada de pontos (vértices do polígono):

```
>>> poligono = [(0,0), (1,2), (2,-1)]

>>> poligono[0]   # primeiro vértice
(0, 0)

>>> poligono[1]   # segundo vértice
(1, 2)

>>> len(poligono) # número de vértices
3
```

2.2 Combinar listas e tuplos

Combinar listas e tuplos

Exemplo: uma parte da tabela periódica

```
>>> grupo12 = [('Zn',30),('Cd',48),
               ('Hg',80),('Uub',112)]

>>> for element in grupo12:
...   print element

('Zn', 30)
('Cd', 48)
('Hg', 80)
('Uub', 112)
```

Projecções e selecções (1)

```
>>> grupo12 = [('Zn',30),('Cd',48),
               ('Hg',80),('Uub',112)]

>>> for element in grupo12:
...   if element[1]>50:
...     print element[0],'\t',element[1]

Hg      80
Uub     112
```

Projecções e selecções (2)

```
>>> grupo12 = [('Zn',30),('Cd',48),
               ('Hg',80),('Uub',112)]

>>> for (nome,peso) in grupo12:
...   if peso>50:
...     print nome,'\t',peso

Hg      80
Uub     112
```