

Tuplos

Programação I
2017.2018

Teresa Gonçalves
tcg@uevora.pt

Departamento de Informática, ECT-UÉ

Reminder...



Como aprender?

Estudar, estudar, estudar...

Praticar, praticar, praticar...

Cometer erros, cometer erros, cometer erros...

Aprender com os erros,
aprender com os erros,
aprender com os erros ...

Sumário

Propriedades, operadores e indexação

Iteração

Funções pré-definidas

Tuplos e definição de funções

Tuplos vs. Listas

Propriedades, operadores e indexação

Tuplo

Sequência de itens

Os itens podem ser de tipos diferentes

Usualmente entre ()

Mas não é obrigatório!

Tamanho da sequência

Função len()

Exemplo

```
coordenadas=(1,7,-1)
```

```
type(coordenadas)
```

```
<class 'tuple'>
```

Propriedades

Sequencial

Relação de ordem entre elementos

Indexável

Acesso direto a cada elemento através do índice

Imutável

Não é possível alterar o valor de um item do tuplo

```
coordenadas=(4,7,-1)
```

```
coordenadas[0]=5
```

```
traceback(most recent call)
```

```
TypeError: 'tuple' object does not support item assignment
```

Geração de tuplos

A atribuição de uma sequência de valores separados por vírgula gera um tuplo

Exemplo

```
tup1 = 2011,  
type(tup1)  
<class 'tuple'>
```

```
tup2=9,8,7  
type(tup2)  
<class 'tuple'>
```

```
tup3=(1, 'dois', [3], (4), (5,))  
type(tup3[3])  
<class 'int'>
```

```
class(tup3[4])  
<class 'tuple'>
```

Operadores básicos

Concatenação

```
a=(2,1); b=('ola',5,1)
print(a+b)
(2,1,'ola',5,1)
```

Repetição

```
tuplo=b*2
print(tuplo)
('ola',5,1,'ola',5,1)
```


Tuplos vazios, índices e segmentos

Tuplo vazio

Tuplo com zero itens

Indicado por ()

Índices e segmentos

Têm a mesma semântica que os outros tipos (strings, listas)

Os segmentos de tuplos geram tuplos!

Exemplo

```
t=('a', 'b', 'c', 'd', 'e', 'f')
```

```
print(t[-1])
```

```
'f'
```

```
print(t[0::2])
```

```
('a', 'c', 'e')
```

```
print(t[-2:])
```

```
('e', 'f')
```

Operador ==

Verificação de equivalência

Operador binário booleano

Retorna True se os tuplos tiverem os mesmos elementos

A ordem tem importância

Exemplo

```
a=(1,2,3); b=(); c=(1,3,2); d=(1,2,3)
```

```
print(a==b)
```

False

```
print(a==c)
```

False

```
print(a==d)
```

True

Operador is

Verificação de identidade

Operador binário booleano

Retorna True as variaveis referem o mesmo tuplo

Exemplo

```
a=(1,2); b=a; c=(1,2)
```

```
print(a==b)
```

True

```
print(a is b)
```

True

```
print(a==c)
```

True

```
print(a is c)
```

False

Operador in

expr in tuplo

Operador binário booleano

Operação entre um item e um tuplo

Retorna True se o resultado da expressão é um item do tuplo

Exemplo

```
tuplo=('pao','queijo','fiambre','manteiga')  
print('pao' in tuplo)
```

True

```
print('azeite' in tuplo)
```

False

Operadores <, >

Comparação

Comparam o 1º item

Se maior/menor, os restantes itens não são considerados

Se igual compara-se o seguinte...

Exemplo

```
a=(1,2,3); b=(1,3,3); c=(0,3)
```

```
print(a<b)
```

True

```
print(a>b)
```

False

```
print(a>c)
```

True

Iteração

Iteração sobre tuplos

Iterar sobre os índices (for)

```
for indice in range(len(tuplo)):  
    print( tuplo[indice] )
```

Iterar sobre os índices (while)

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

Packing & Unpacking

Packing

Criar um tuplo a partir de uma sequência de valores

```
t=1,2,3
```

```
type(t)
```

```
<class 'tuple'>
```

Unpacking

Atribuir valores de um tuplo a variáveis

```
a,b,c = t
```

```
print(b)
```

```
2
```


Packing & Unpacking

Packing e unpacking

`a,b=b,a` # troca o valor das variáveis

Funcionamento

Primeiro são avaliadas as expressões do lado direito

Depois o respetivo valor é atribuído a cada variável do lado esquerdo

Nota

A maioria das linguagens de programação não permite esta construção!!!

Tuplos e funções

Número variável de argumentos

Exemplo

```
print(max(5,6))
```

6

```
print(max(5,6,1))
```

6

```
print(max(5,6,1,9))
```

9

Devolução de mais de um valor

Exemplo

```
def divide(a,b):  
    q=0; r=0  
    while a>b:  
        a=a-b  
        q=q+1  
    r=a  
    return(q,r)  
quo,resto = divide(15,4)  
print(quo)  
3  
print(resto)  
3
```

Definição de um nº variável de argumentos

Marcar o parâmetro com o *

Identifica-o como sendo um tuplo

```
def maximo(*valores):  
    if len(valores)==0:  
        return None  
    m=valores[0]  
    for i in range(1,len(valores))  
        if valores[i]>m:  
            m=valores[i]  
    return m
```

Passagem de tuplo como argumento

Marcação do argumento com *

Permite passar um tuplo a funções que não aceitam tuplos

```
range( (1,6) )
```

*Traceback (most recent call last): TypeError: range()
integer end argument expected, got tuple.*

```
print(range(*(1,6)))
```

range(1,6)

Tuplos vs. Listas

Tuplos vs. Listas

Tuplos

()

Imutavel

Convenção

Heterogéneo: sequencia de diferentes tipos de “coisas”

Constitui uma unidade coerente

Listas

[]

Mutável

Convenção

Homogéneo: sequencia do mesmo tipo de “coisas”

Cada elemento é tratado individualmente