Кортежи

Глава 10

Python for Informatics: Exploring Information
www.pythonlearn.com

Кортежи подобны спискам

- Кортеж - это еще один вид последовательности по своим функциям похожей на списки. В кортежах имеются элементы, индексация которых начинается с 0.

```
>>> x = ('Glenn', 'Sally', 'Joseph')
>>> print x[2]Joseph
>>> y = ( 1, 9, 2 )
>>> print y
(1, 9, 2)
>>> print max(y)
9
```

```
>>> for iter in y:
...     print iter
...
1
9
2
>>>
```

Но... кортежи являются неизменяемыми

- В отличие от содержимого списка, содержимое кортежа нельзя изменить (как и в строках)

```
>>> x = [9, 8, 7]
>>> x[2] = 6
>>> print x
>>>[9, 8, 6]
>>>
```

```
>>> y = 'ABC'
>>> y[2] = 'D'
Traceback:'str'
object does
not support item
Assignment
>>>
```

```
>>> z = (5, 4, 3)
>>> z[2] = 0
Traceback:'tuple'
object does
not support item
Assignment
>>>
```

Что нельзя делать с кортежами

```
>>> x = (3, 2, 1)
>>> x.sort()
Traceback:
AttributeError: 'tuple' object has no attribute 'sort'
>>> x.append(5)
Traceback:
AttributeError: 'tuple' object has no attribute 'append'
>>> x.reverse()
Traceback:
AttributeError: 'tuple' object has no attribute 'reverse'
>>>
```

История о двух последовательностях

```
>>> l = list()
>>> dir(l)
['append', 'count', 'extend', 'index', 'insert', 'pop',
'remove', 'reverse', 'sort']

>>> t = tuple()
>>> dir(t)
['count', 'index']
```

Кортежи являются более эффективными

- Поскольку структура кортежа в языке Python является неизменяемой, с точки зрения производительности и использования памяти кортежи являются более простыми и эффективными, чем списки
- Следовательно, при создании “временных переменных” в наших программах мы отдаем предпочтение кортежам

Кортежи и присваивание

- Кортеж можно также поставить в левой стороне инструкции присваивания
- Можно даже опустить скобки

```
>>> (x, y) = (4, 'fred')
>>> print y
Fred
>>> (a, b) = (99, 98)
>>> print a
99
```

Кортежи и словари

- Метод items() в словарях возвращает список кортежей (ключ, значение)

```
>>> d = dict()
>>> d['csev'] = 2
>>> d['cwen'] = 4
>>> for (k,v) in d.items():
...     print k, v
...
csev 2
cwen 4
>>> tups = d.items()
>>> print tups
[('csev', 2), ('cwen', 4)]
```

Кортежи можно сравнить

- Операторы сравнения работают с кортежами и другими последовательностями. Если первый элемент одной последовательности равен первому элементу другой, Python переходит к следующему элементу, и так до тех пор, пока не найдет отличающиеся элементы.

```
>>> (0, 1, 2) < (5, 1, 2)
True
>>> (0, 1, 2000000) < (0, 3, 4)
True
>>> ( 'Jones', 'Sally' ) < ('Jones', 'Sam')
True
>>> ( 'Jones', 'Sally') > ('Adams', 'Sam')
True
```

Сортировка списков кортежей

- Мы можем воспользоваться возможностью сортировки списка кортежей, чтобы получить отсортированный словарь
- Для этого мы сортируем словарь по ключам с помощью метода items()

```
>>> d = {'a':10, 'b':1, 'c':22}
>>> t = d.items()
>>> t
[('a', 10), ('c', 22), ('b', 1)]
>>> t.sort()
>>> t
[('a', 10), ('b', 1), ('c', 22)]
```

Метод sorted()

Мы также можем сделать это напрямую с помощью встроенной функции sorted, которая принимает последовательность в качестве параметра и возвращает отсортированную последовательность

```
>>> d = {'a':10, 'b':1, 'c':22}
>>> d.items()
[('a', 10), ('c', 22), ('b', 1)]
>>> t = sorted(d.items())
>>> t
[('a', 10), ('b', 1), ('c', 22)]

>>> for k, v in sorted(d.items()):
...     print k, v
...
a 10
b 1
c 22
```

Сортировка по значениям вместо ключей

- Если создать список кортежей в виде (значение, ключ), то мы сможем отсортировать словарь по значениям
- Это делается с помощью цикла for, который создает список кортежей

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

```
fhand = open('romeo.txt')
counts = dict()
for line in fhand:
    words = line.split()
    for word in words:
        counts[word] = counts.get(word, 0 ) + 1

lst = list()
for key, val in counts.items():
    lst.append( (val, key) )
lst.sort(reverse=True)
for val, key in lst[:10] :
    print key, val
```

10 наиболее
употребляемых слов

Короткая версия

```
>>> c = {'a':10, 'b':1, 'c':22}

>>> print sorted( [ (v,k) for k,v in c.items() ] )

[(1, 'b'), (10, 'a'), (22, 'c')]
```

Списковое включение создает динамический список. В данном случае мы создаем список обратных кортежей и сортируем его.

http://wiki.python.org/moin/HowTo/Sorting

Обзор

- Синтаксис кортежей
- Неизменяемость
- Сравнение
- Сортировка

- Кортежи в инструкциях присваивания
- Сортировка словарей по ключам или значениям

Благодарность / Содействие

Первоначальная разработка: Чарльз Северанс, Школа информации Мичиганского университета

Здесь впишите дополнительных авторов и переводчиков...

...