

Кортежи

Глава 10



Пайтон для всех
www.py4e.com



Кортежи похожи на списки

Кортежи — еще один вид последовательности, которая очень похожа на список. Элементы кортежа индексируются, начиная с 0

```
>>> x = ('Гленн', 'Салли', 'Джозеф')
>>> print(x[2])
Джозеф
>>> y = ( 1, 9, 2 )
>>> print(y)
(1, 9, 2)
>>> print(max(y))
9
>>> for iter in y:
...     print(iter)
...
1
9
2
>>>
```

но... Кортежи неизменяемы

```
>>> 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
>>>
```

Ни строка, ни кортеж не поддерживают присваивание элементов

Что нельзя сделать с кортежем

```
>>> x = (3, 2, 1)
```

```
>>> x.sort()
```

```
Traceback:
```

```
AttributeError: 'tuple' object has no attribute 'sort'
```

Кортеж не сортируется

```
>>> x.append(5)
```

```
Traceback:
```

```
AttributeError: 'tuple' object has no attribute 'append'
```

В кортеж нельзя добавить элемент

```
>>> x.reverse()
```

```
Traceback:
```

```
AttributeError: 'tuple' object has no attribute 'reverse'
```

Нельзя изменить порядок элементов

```
>>>
```

История двух последовательностей

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

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

Кортежи более эффективны

- Поскольку в Пайтон кортежи являются неизменяемыми, они гораздо проще и эффективнее с точки зрения использования памяти и производительности, чем списки
- Поэтому в нашей программе при создании «временных переменных» мы отдаем предпочтение кортежам, а не спискам

Кортежи и присваивание

- Мы можем поместить **кортеж** в **левую часть** выражения присваивания
- Мы даже можем опустить круглые скобки:

```
>>> (x, y) = (4, 'Фред')
```

```
>>> print(y)
```

```
Фред
```

```
>>> (a, b) = (99, 98)
```

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

```
99
```

Кортежи и Словари

Метод `items()`,
примененный к
словарю, 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)
dict_items([('csev', 2), ('cwen',
4)])
```

Кортежи сопоставимы

Операторы сравнения работают с **кортежами** и другими последовательностями. Если первые элементы равны, Пайтон переходит к следующему элементу, и так далее, пока не найдет элементы, которые отличаются.

```
>>> (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

- Подвергнув сортировке список **кортежей**, мы можем получить отсортированную версию словаря
- Сначала сортируем словарь по ключу, используя метод **items()**, а затем применяем функцию **sorted()**

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

Использование sorted()

Можно сделать еще проще:
используем встроенную
функцию `sorted()`, которая
принимает последовательность
в качестве параметра и
возвращает отсортированную
последовательность

```
>>> d = {'a':10, 'b':1, 'c':22}
>>> 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, а не по Ключам

- Если бы мы могли получить список **кортежей** в виде - **(значение, ключ)**, то смогли бы **отсортировать** содержимое по значению
- Мы можем сделать это при помощи цикла с **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 = sorted(tmp,
reverse=True)
>>> print(tmp)
[(22, 'c'), (10, 'a'), (1, 'b')]
```

10 самых распространенных слов

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

lst = []
for key, val in counts.items():
    newtup = (val, key)
    lst.append(newtup)

lst = sorted(lst, reverse=True)

for val, key in lst[:10]:
    print(key, val)
```

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')]
```

Генератор списка создает динамический список.
В данном случае мы создаем список инвертированных кортежей (значение, ключ), а затем сортируем его.

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

Резюме

- Синтаксис кортежей
- Неизменяемость
- Сопоставимость
- Сортировка
- Кортежи в операторах присваивания
- Сортировка словарей по ключу или по значению



Авторы / Благодарности



... Insert new Contributors and Translations here

Авторские права на эти слайды принадлежат Чарльзу Р. Северансу (www.dr-chuck.com), 2010 г., Школе Информации Мичиганского Университета и open.umich.edu, и доступны по лицензии Creative Commons Attribution 4.0 License.

Пожалуйста, сохраняйте этот слайд во всех копиях этого документа, в соответствии с требованиями Лицензии. Если вы внесли изменения, добавьте свое имя или организацию в список участников на этой странице.

Исходная разработка: Чарльз Северанс, Школа Информации Мичиганского Университета.

Перевод выполнила Фомкина Виолетта.

... Добавьте сюда новых авторов и переводчиков