Списки в Пайтон

Глава 8

Пайтон для всех
www.py4e.com

Программирование

- Алгоритм
  - Набор правил или шагов, используемых для решения задачи

- Структура данных
  - Особый способ организации данных на компьютере

https://ru.wikipedia.org/wiki/Алгоритм
https://ru.wikipedia.org/wiki/Структура_данных

Что Не является «Коллекцией»?

Большинство переменных хранят внутри одно значение. Когда мы присваиваем переменной новое значение, старое значение перезаписывается / заменяется

```
$ python
>>> x = 2
>>> x = 4
>>> print(x)
4
```

Список — это подобие Коллекции

- Коллекция позволяет нам помещать множество значений в одну переменную
- Удобство коллекции в том, что мы можем хранить много значений внутри одной удобной «упаковки».

```
friends = [ 'Катя', 'Антон', 'Павел' ]

carryon = [ 'носки', 'футболка', 'духи' ]
```

Константы списка

- Константы списка заключаются в квадратные скобки, а элементы списка разделяются запятыми
- Элементом списка в Пайтон может быть любой объект, даже другой список
- Список может быть пустым

```
>>> print([1, 24, 76])
[1, 24, 76]
>>> print(['красный', 'желтый',
'голубой'])
['красный', 'желтый', 'голубой']
>>> print(['красный', 24, 98.6])
['красный', 24, 98.6]
>>> print([ 1, [5, 6], 7])
[1, [5, 6], 7]
>>> print([])
[]
```

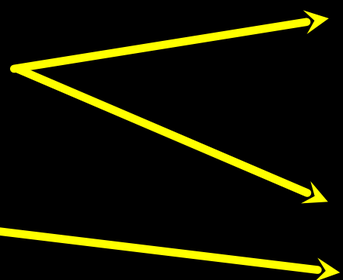
Мы уже используем списки!

```
for i in [5, 4, 3, 2, 1] :
    print(i)
print('Старт!')
```

```
5
4
3
2
1
Старт!
```

Списки и циклы со счетчиком — лучшие друзья

```
friends = ['Анатолий', 'Роман',
'Татьяна']
for friend in friends :
    print('С Новым Годом:',  friend)
print('Готово!')
```



```
С Новым Годом: Анатолий
С Новым Годом: Роман
С Новым Годом: Татьяна
Готово!
```

```
z = ['Анатолий', 'Роман', 'Татьяна']
for x in z:
    print('С Новым Годом:',  x)
print('Готово!')
```

Заглянем внутрь списка

Как и в случае со строками, мы можем получить любой отдельный элемент списка, указав его индекс в квадратных скобках

Катя	Маша	Петя
0	1	2

```
>>> friends = [ 'Катя', 'Маша', 'Петя' ]
>>> print(friends[1])
Маша
>>>
```

Списки изменяемы

- Строки неизменяемы, мы не можем изменить содержимое строки. Нам необходимо создать новую строку, чтобы внести какие-либо изменения
- Списки изменяемы, мы можем изменять элемент списка, используя взятие по индексу

```
>>> fruit = 'Banana'
>>> fruit[0] = 'b'
Traceback
TypeError: 'str' object does not
support item assignment
>>> x = fruit.lower()
>>> print(x)
banana
>>> lotto = [2, 14, 26, 41, 63]
>>> print(lotto)
[2, 14, 26, 41, 63]
>>> lotto[2] = 28
>>> print(lotto)
[2, 14, 28, 41, 63]
```

Как узнать длину списка?

- Функция len() принимает список в качестве параметра и возвращает количество элементов в списке
- На самом деле функция len() возвращает количество элементов в любом наборе или последовательности (например, в строке)

```
>>> greet = "Привет, Борис!"
>>> print(len(greet))
14
>>> x = [ 1, 2, 'солнце', 99]
>>> print(len(x))
4
>>>
```

Использование функции range

- Функция range возвращает список чисел в диапазоне от нуля до числа на единицу меньше, чем указанный параметр
- Мы можем написать цикл по индексам элементов, используя for целое число в качестве счетчика

```
>>> print(range(4))
[0, 1, 2, 3]
>>> friends = ['Петр', 'Татьяна',
'Вероника']
>>> print(len(friends))
3
>>> print(range(len(friends)))
[0, 1, 2]
>>>
```

История двух циклов...

```
friends = ['Татьяна', 'Александр',
'Мария']

for friend in friends :
    print('С Новым Годом:',  friend)

for i in range(len(friends)) :
    friend = friends[i]
    print('С Новым Годом :',  friend)
```

```
>>> friends = ['Татьяна', 'Александр',
'Мария']
>>> print(len(friends))
3
>>> print(range(len(friends)))
[0, 1, 2]
>>>
```

С Новым Годом: Татьяна

С Новым Годом: Александр

С Новым Годом: Мария

Объединение списков с помощью +

Можно создать новый список, объединив два существующих списка вместе

```
>>> a = [1, 2, 3]
>>> b = [4, 5, 6]
>>> c = a + b
>>> print(c)
[1, 2, 3, 4, 5, 6]
>>> print(a)
[1, 2, 3]
```

Срез списка с помощью :

```
>>> t = [9, 41, 12, 3, 74, 15]
>>> t[1:3]
[41,12]
>>> t[:4]
[9, 41, 12, 3]
>>> t[3:]
[3, 74, 15]
>>> t[:]
[9, 41, 12, 3, 74, 15]
```

Помните: как и в случае со строками, второе число означает «до, но не включая»

Методы списка

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

http://docs.python.org/tutorial/datastructures.html

Создаем список с нуля

- Мы можем создать пустой список, а затем добавить в него элементы, используя метод append
- Список остается упорядоченным, новые элементы добавляются в конец списка

```
>>> stuff = list()
>>> stuff.append('книга')
>>> stuff.append(99)
>>> print(stuff)
['книга', 99]
>>>
stuff.append('печенька')
>>> print(stuff)
['книга', 99, 'печенька']
```

Есть ли элемент в списке?

- В Пайтон имеется два оператора, позволяющих проверить, находится ли элемент в списке
- Это логические операторы, возвращающие True (Правда) или False (Ложь)
- Они не изменяют список

```
>>> some = [1, 9, 21, 10, 16]
>>> 9 in some
True
>>> 15 in some
False
>>> 20 not in some
True
>>>
```

Списки являются упорядоченными

- Список может содержать множество элементов и хранить их в определенном порядке, пока мы не изменим порядок
- Список можно сортировать (т.е., изменять его порядок)
- Метод sort (в отличие от строк) означает «сортировать себя»

```
>>> friends = [ 'Михаил', 'Глеб',
'Ирина' ]
>>> friends.sort()
>>> print(friends)
['Глеб', 'Михаил', 'Ирина']
>>> print(friends[1])
Михаил
>>>
```

Встроенные функции и списки

- В Python имеется ряд встроенных функций, которые принимают списки в качестве параметров
- Помните циклы, которые мы написали? Это намного проще

```
>>> nums = [3, 41, 12, 9, 74, 15]
>>> print(len(nums))
6
>>> print(max(nums))
74
>>> print(min(nums))
3
>>> print(sum(nums))
154
>>> print(sum(nums)/len(nums))
25.6
```

```
total = 0
count = 0
while True :
    inp = input('Введите число: ')
    if inp == 'Готово' : break
    value = float(inp)
    total = total + value
    count = count + 1

average = total / count
print('Среднее арифметическое:',
average)
```

```
Введите число: 3
Введите число : 9
Введите число : 5
Введите число : done
Среднее арифметическое:
5.66666666667
```

```
numlist = list()
while True :
    inp = input('Введите число: ')
    if inp == 'Готово' : break
    value = float(inp)
    numlist.append(value)

average = sum(numlist) / len(numlist)
print('Среднее арифметическое:',
average)
```

Лучшие друзья: строки и списки

```
>>> abc = 'Эти три слова'
>>> stuff = abc.split()
>>> print(stuff)
['Эти', 'три', 'слова']
>>> print(len(stuff))
3
>>> print(stuff[0])
Эти
```

```
>>> print(stuff)
['Эти', 'три', 'слова']
>>> for w in stuff :
...     print(w)
...
Эти
три
слова
>>>
```

Split разбивает строку на части и создает из них список, состоящий из строк. Мы представляем их как отдельные слова. Можно получить доступ к определенному слову или с помощью цикла пройти по всем словам.

```
>>> line = 'Тут очень
много пробелов'
>>> etc = line.split()
>>> print(etc)
['Тут', 'очень', 'много',
'пробелов']
>>>
>>> line = 'первый;второй;третий'
>>> thing = line.split()
>>> print(thing)
['первый;второй;третий']
>>> print(len(thing))
1
>>> thing = line.split(';')
>>> print(thing)
['первый', 'второй', 'третий']
>>> print(len(thing))
3
>>>
```

- Если вы не укажете разделитель, при разделении несколько пробелов будут считаться как один
- Вы можете указать какой символ-разделитель использовать при разделении строки

From stephen.marquard@uct.ac.za Sat Jan 5 09:14:16 2008

```
fhand = open('mbox-short.txt')
for line in fhand:
    line = line.rstrip()
    if not line.startswith('From ') : continue
    words = line.split()
    print(words[2])
```

Sat
Fri
Fri
Fri
...

```
>>> line = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
>>> words = line.split()
>>> print(words)
['From', 'stephen.marquard@uct.ac.za', 'Sat', 'Jan', '5', '09:14:16', '2008']
>>>
```

Пример двойного разделения

Иногда бывает необходимо сначала разделить строку одним образом, а затем взять один из получившихся кусков и разделить его ещё раз

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
words = line.split()
email = words[1]
```

Пример двойного разделения

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
words = line.split()
email = words[1]
```

stephen.marquard@uct.ac.za

Пример двойного разделения

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
words = line.split()
email = words[1]
pieces = email.split('@')
```

```
stephen.marquard@uct.ac.za
['stephen.marquard', 'uct.ac.za']
```

Пример двойного разделения

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
words = line.split()
email = words[1]
pieces = email.split('@')
print(pieces[1])
```

```
stephen.marquard@uct.ac.za
['stephen.marquard', 'uct.ac.za']
'uct.ac.za'
```

Резюме

- Концепция коллекции
- Списки и циклы со счётчиком
- Индексация и поиск
- Изменяемость списков
- Функции: len, min, max, sum
- Срезы в списках
- Методы списка: append, remove
- Сортировка списка
- Разделение строк на списки слов
- Использование split для разделения строк

Авторы / Благодарности

Исходная разработка: Чарльз Северанс, Школа Информации Мичиганского Университета.

Перевод выполнила Фомкина Виолетта.

… Добавьте сюда новых авторов и переводчиков

...