

파이썬 리스트

제8장

Python for Everybody
www.py4e.com



프로그래밍

- 알고리즘

- 문제를 해결하는 데 사용하는 일련의 규칙 또는 단계

- 자료 구조

- 컴퓨터에서 자료를 구성하는 특별한 방법

<https://en.wikipedia.org/wiki/Algorithm>

https://en.wikipedia.org/wiki/Data_structure

“컬렉션”이 아닌 것은?

대부분의 변수는 한 값을 갖음 - 변수에 새 값을 대입하면,
이전 값에 덮어씌워 짐

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

리스트는 컬렉션의 일종



- 컬렉션은 하나의 “변수”에 많은 값을 넣을 수 있음
- 컬렉션을 쓰면 많은 변수를 전부 하나의 편리한 꾸러미에 넣어다닐 수 있어서 유용

```
friends = [ 'Joseph', 'Glenn', 'Sally' ]
```

```
carryon = [ 'socks', 'shirt', 'perfume' ]
```

리스트 상수

- 리스트 상수는 대괄호로 둘러싸여 있고 리스트의 원소는 반점으로 구분됨
- 리스트는 파이썬의 어떤 객체도 원소로 넣을 수 있음
 - 심지어 다른 리스트를 넣는 것도 가능
- 빈 리스트도 생성 가능

```
>>> print([1, 24, 76])  
[1, 24, 76]  
>>> print(['red', 'yellow',  
'blue'])  
['red', 'yellow', 'blue']  
>>> print(['red', 24, 98.6])  
['red', 24, 98.6]  
>>> print([1, [5, 6], 7])  
[1, [5, 6], 7]  
>>> print([])  
[]
```

이미 리스트를 쓰고 있었음!

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

5

4

3

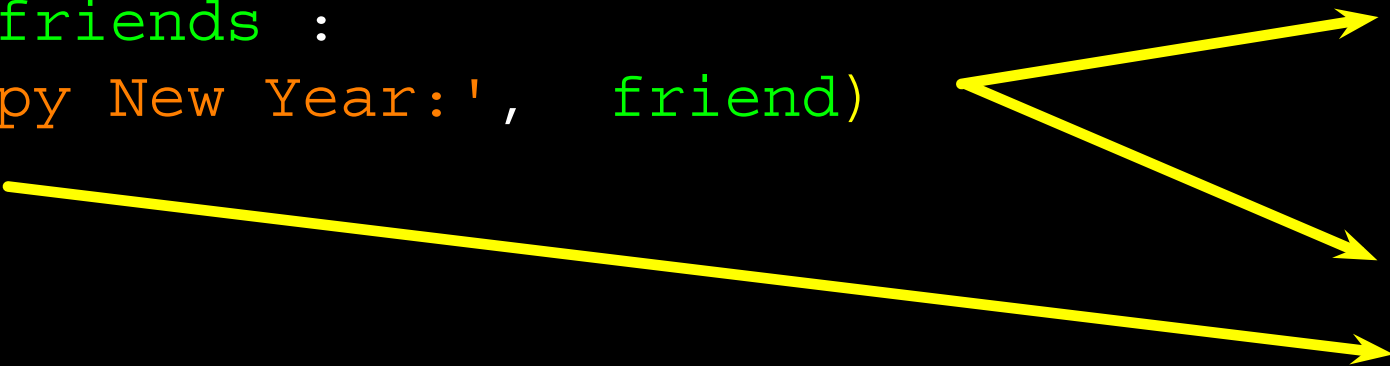
2

1

Blastoff!

리스트와 유한 루프 - 최고의 친구들

```
friends = ['Joseph', 'Glenn', 'Sally']  
for friend in friends :  
    print('Happy New Year:', friend)  
print('Done!')
```



Happy New Year: Joseph
Happy New Year: Glenn
Happy New Year: Sally
Done!

```
z = ['Joseph', 'Glenn', 'Sally']  
for x in z:  
    print('Happy New Year:', x)  
print('Done!')
```



리스트의 내부

문자열과 마찬가지로, **대괄호**를 이용한 인덱스로 리스트의 원소 하나하나를 가져올 수 있음

Joseph	Glenn	Sally
0	1	2

```
>>> friends = [ 'Joseph', 'Glenn', 'Sally' ]
>>> print(friends[1])
Glenn
>>>
```


리스트는 변경 가능

- 문자열은 “변경 불가”
 - 문자열의 내용을 변경할 수 없음
 - 변경하려면 새 문자열을 만들어야 함
- 리스트는 “변경 가능”
 - 인덱스 연산자를 사용하여 리스트의 요소를 변경 가능

```
>>> 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]
```

리스트의 길이는?

- `len()` 함수는 리스트를 매개 변수로 받고 리스트의 원소 개수를 반환함
- 사실 `len()` 함수는 아무 집합이나 시퀀스(예: 문자열)를 받아 원소의 개수를 반환함

```
>>> greet = 'Hello Bob'
>>> print(len(greet))
9
>>> x = [ 1, 2, 'joe', 99]
>>> print(len(x))
4
>>>
```

range 함수 사용하기

- **range** 함수는 0부터 매개 변수로 넣은 값보다 1 작은 범위의 수까지로 구성된 숫자 리스트를 반환
- **for** 문과 정수 반복자를 통해 인덱스 루프를 구성 가능

```
>>> print(range(4))
[0, 1, 2, 3]
>>> friends = ['Joseph', 'Glenn', 'Sally']
>>> print(len(friends))
3
>>> print(range(len(friends)))
[0, 1, 2]
>>>
```

두 가지 루프 이야기...

```
friends = ['Joseph', 'Glenn', 'Sally']
```

```
for friend in friends :  
    print('Happy New Year:', friend)
```

```
for i in range(len(friends)) :  
    friend = friends[i]  
    print('Happy New Year:', friend)
```

```
>>> friends = ['Joseph', 'Glenn', 'Sally']
```

```
>>> print(len(friends))
```

```
3
```

```
>>> print(range(len(friends)))
```

```
[0, 1, 2]
```

```
>>>
```

Happy New Year: Joseph

Happy New Year: Glenn

Happy New Year: Sally

+를 사용하여 리스트 연결하기

기존에 존재하는 두 리스트를
더하여 새로운 리스트 생성 가능

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

:를 사용하여 리스트 자르기

```
>>> 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]
```

주의: 문자열과 마찬가지로,
괄호 안의 두 번째 숫자
“직전”까지(미만)만 포함

리스트 메서드

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

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

리스트를 처음부터 만들기

- 빈 리스트를 만들고
append 메서드를 이용하여
원소를 추가할 수 있음
- 리스트 안은 순서가 유지되고
새 원소는 리스트 끝에 더해짐

```
>>> stuff = list()
>>> stuff.append('book')
>>> stuff.append(99)
>>> print(stuff)
['book', 99]
>>> stuff.append('cookie')
>>> print(stuff)
['book', 99, 'cookie']
```


리스트 원소 탐색

- 파이썬은 특정 원소가 리스트에 있는지 확인할 수 있는 두 가지의 연산자를 제공함
- 둘 다 참(True)과 거짓(False)을 반환하는 논리 연산자
- 리스트를 바꾸지는 않음

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

리스트에는 순서가 있다

- 리스트는 많은 아이템을 보관할 수 있고 사용자가 순서를 바꾸기 위해 별도의 행동을 하지 않는 한 아이템의 순서를 유지
- 리스트는 정렬 가능
(예: 원소 간 순서 바꾸기)
- `sort` 메서드는 “스스로를 정렬”하는 기능 (문자열과는 다름)

```
>>> friends = [ 'Joseph', 'Glenn', 'Sally' ]
>>> friends.sort()
>>> print(friends)
['Glenn', 'Joseph', 'Sally']
>>> print(friends[1])
Joseph
>>>
```

내장 함수와 리스트

- 파이썬에는 리스트를 매개 변수로 받는 내장 함수가 여러 가지 있음
- 우리가 작성했던 루프를 기억합니까? 내장 함수를 사용하는 것이 훨씬 간단함

```
>>> 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('Enter a number: ')
    if inp == 'done' : break
    value = float(inp)
    total = total + value
    count = count + 1
```

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

Enter a number: 3

Enter a number: 9

Enter a number: 5

Enter a number: done

Average: 5.6666666666667

```
numlist = list()
while True :
    inp = input('Enter a number: ')
    if inp == 'done' : break
    value = float(inp)
    numlist.append(value)

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

최고의 친구들: 문자열과 리스트

```
>>> abc = 'With three words'
>>> stuff = abc.split()
>>> print(stuff)
['With', 'three', 'words']
>>> print(len(stuff))
3
>>> print(stuff[0])
With
```

```
>>> print(stuff)
['With', 'three', 'words']
>>> for w in stuff :
...     print(w)
...
With
Three
Words
>>>
```

split 함수는 문자열을 작게 나누고 문자열로 구성된 리스트를 생성
이 문자열은 단어로 볼 수 있음
특정 단어에 접근하거나 모든 단어에 대해 루프를 실행할 수 있음

```
>>> line = 'A lot of spaces'
```

```
>>> etc = line.split()
```

```
>>> print(etc)
```

```
['A', 'lot', 'of', 'spaces']
```

```
>>>
```

```
>>> line = 'first;second;third'
```

```
>>> thing = line.split()
```

```
>>> print(thing)
```

```
['first;second;third']
```

```
>>> print(len(thing))
```

```
1
```

```
>>> thing = line.split(';')
```

```
>>> print(thing)
```

```
['first', 'second', 'third']
```

```
>>> print(len(thing))
```

```
3
```

```
>>>
```

- 구획 문자를 별도로 설정하지 않으면, 여러 칸의 공백도 하나의 구획 문자(여기서는 공백)로 여겨짐
- 문장을 나눌 때 어떤 구획 문자를 사용할지 정할 수 있음

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']
>>>
```

이중으로 나누는 패턴

문장을 하나의 구획 문자로 나누고 그 조각 중 일부를
다시 다른 구획 문자로 나누는 패턴

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

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


이중으로 나누는 패턴

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

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

stephen.marquard@uct.ac.za

이중으로 나누는 패턴

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

```
words = line.split()  
email = words[1]           stephen.marquard@uct.ac.za  
pieces = email.split('@')  ['stephen.marquard', 'uct.ac.za']
```

이중으로 나누는 패턴

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'

리스트 요약

- 컬렉션의 개념
- 리스트와 유한 루프
- 인덱스 생성과 검색
- 리스트의 가변성
- 여러 함수: `len`, `min`, `max`, `sum`
- 리스트 자르기
- 리스트 메서드: `append`, `remove`
- 리스트 정렬
- 문자열을 단어 리스트로 나누기
- `split`을 이용한 문자열 분석



Acknowledgements / Contributions



These slides are Copyright 2010- Charles R. Severance (www.dr-chuck.com) of the University of Michigan School of Information and open.umich.edu and made available under a Creative Commons Attribution 4.0 License. Please maintain this last slide in all copies of the document to comply with the attribution requirements of the license. If you make a change, feel free to add your name and organization to the list of contributors on this page as you republish the materials.

Initial Development: Charles Severance, University of Michigan School of Information

Contributor:

- 이승준
- 커넥트재단

Translation:

- Hyunjun Kim
- Jeungmin Oh (tangza@gmail.com)