

파이썬 문법 학습

https://realpython.com/python-dicts/

- 검색 realpython dict

The screenshot shows the Real Python website with the article 'Dictionaries in Python' by John Sturtz. The page features a navigation bar with links like 'Start Here', 'Learn Python', 'Store', and 'More'. A search bar and 'Join'/'Sign-In' buttons are also present. The main content area includes a large illustration of a robot and a person shaking hands, followed by the article title and author. A 'Table of Contents' section lists topics like 'Defining a Dictionary' and 'Accessing Dictionary Values'. On the right, there's a 'FREE Email Series' for 'Python Tricks' with a code snippet for merging dictionaries. Below that, 'All Tutorial Topics' are listed in a grid. At the bottom, a green banner says 'Improve Your Python'.

Real Python

Dictionaries in Python

by John Sturtz 21 Comments basics python

[Tweet](#) [Share](#) [Email](#)

Table of Contents

- Defining a Dictionary
- Accessing Dictionary Values
- Dictionary Keys vs. List Indices
- Building a Dictionary Incrementally
- Restrictions on Dictionary Keys
- Restrictions on Dictionary Values
- Operators and Built-in Functions
- Built-in Dictionary Methods

— FREE Email Series —

Python Tricks

```
1# How to merge two dicts
2# in Python 3.5+
3
4>>> x = {'a': 1, 'b': 2}
5>>> y = {'b': 3, 'c': 4}
6
7>>> z = {**x, **y}
8
9>>> z
10{'c': 4, 'a': 1, 'b': 3}
```

Email...

[Get Python Tricks »](#)

No spam. Unsubscribe any time.

All Tutorial Topics

advanced api basics best-practices
community databases data-science
devops django docker flask front-end
intermediate machine-learning python
testing tools web-dev web-scraping

Enhance Python with Redis

Improve Your Python

<https://srv.realpython.net/click/23076429117/?c=52901525492&p=58946116052&r=33409>

딕셔너리 학습

Real Python

Start Here Learn Python Store More

Stuck at home? Enjoy free courses, on us →

- If the key is already present in `d`, the corresponding value in `d` for that key is updated to the value from `<obj>`.

Here is an example showing two dictionaries merged together:

```
Python
>>> d1 = {'a': 10, 'b': 20, 'c': 30}
>>> d2 = {'b': 200, 'd': 400}

>>> d1.update(d2)
>>> d1
{'a': 10, 'b': 200, 'c': 30, 'd': 400}
```

In this example, key 'b' already exists in `d1`, so its value is updated to 200, the value for that key from `d2`. However, there is no key 'd' in `d1`, so that key-value pair is added from `d2`.

`<obj>` may also be a sequence of key-value pairs, similar to when the `dict()` function is used to define a dictionary. For example, `<obj>` can be specified as a list of tuples:

```
Python
>>> d1 = {'a': 10, 'b': 20, 'c': 30}
>>> d1.update([('b', 200), ('d', 400)])
>>> d1
{'a': 10, 'b': 200, 'c': 30, 'd': 400}
```

Or the values to merge can be specified as a list of keyword arguments:

```
Python
```

Table of Contents

- Defining a Dictionary
- Accessing Dictionary Values
- Dictionary Keys vs. List Indices
- Building a Dictionary Incrementally
- Restrictions on Dictionary Keys
- Restrictions on Dictionary Values
- Operators and Built-in Functions
- Built-in Dictionary Methods
- Conclusion

Tweet Share Email

Recommended Video Course
Dictionaries in Python

High Quality Python Video Courses
Watch Now »

Help

<https://srv.realpython.net/click/64754707389/?c=31941813441&p=58946116052&r=81003>

딕셔너리 종료

• 학습 후 퀴즈

Real Python

Start Here Learn Python Store More

Stuck at home? Enjoy free courses, on us →

would be best for a given situation.

Next you will learn about Python **sets**. The set is another composite data type, but it is quite different from either a list or dictionary.

Take the Quiz: Test your knowledge with our interactive “Python Dictionaries” quiz. Upon completion you will receive a score so you can track your learning progress over time:

[Take the Quiz »](#)

« Lists and Tuples in Python Dictionaries in Python Sets in Python »

Watch Now This tutorial has a related video course created by the Real Python team. Watch it together with the written tutorial to deepen your understanding: [Dictionaries in Python](#)

Table of Contents

- Defining a Dictionary
- Accessing Dictionary Values
- Dictionary Keys vs. List Indices
- Building a Dictionary Incrementally
- Restrictions on Dictionary Keys
- Restrictions on Dictionary Values
- Operators and Built-in Functions
- Built-in Dictionary Methods
- [Conclusion](#)

[Tweet](#) [Share](#) [Email](#)

Recommended Video Course
Dictionaries in Python

Python Tricks

Get a short & sweet **Python Trick** delivered to your inbox every couple of days. No spam ever. Unsubscribe any time. Curated by the Real Python team.

```
1 # How to merge two dicts
2 # in Python 3.5+
3
4 >>> x = {'a': 1, 'b': 2}
5 >>> y = {'b': 3, 'c': 4}
6
7 >>> z = {**x, **y}
```

[Watch Now](#)

[Help](#)

컴프리헨션:
내장, 내포, 추약, 해석

리스트, 셋, 사전

컴프리헨션(1)

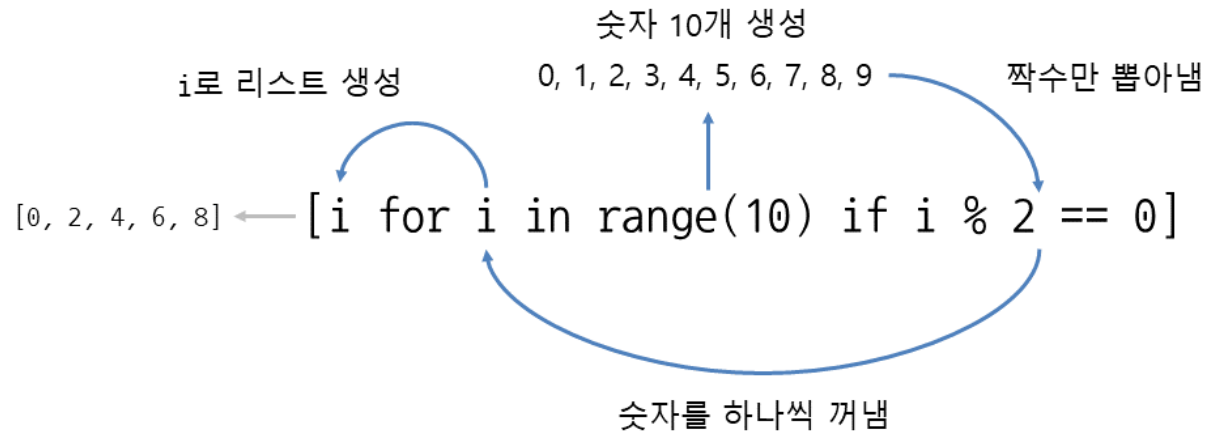
• 국내 사이트

- 컴프리헨션 으로 검색
- <https://wikidocs.net/22805>
- <https://doorbw.tistory.com/174>
- <https://ddanggle.gitbooks.io/interpy-kr/ch15-comprehension.html>
- <https://dojang.io/mod/page/view.php?id=2285>

숫자를 하나씩 꺼냄 숫자 10개 생성
0, 1, 2, 3, 4, 5, 6, 7, 8, 9
i로 리스트 생성
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9] ← [i for i in range(10)]

컴프리헨션(2)

- 조건식이 있는 내포



컴프리헨션(3)

- <https://realpython.com/list-comprehension-python/>



When to Use a List Comprehension in Python

by James Timmins · Nov 06, 2019 · 16 Comments · basics python

[Tweet](#) [Share](#) [Email](#)

Table of Contents

- [How to Create Lists in Python](#)
 - [Using for Loops](#)
 - [Using map\(\) Objects](#)
 - [Using List Comprehensions](#)
 - [Benefits of Using List Comprehensions](#)
- [How to Supercharge Your Comprehensions](#)
 - [Using Conditional Logic](#)
 - [Using Set and Dictionary Comprehensions](#)
 - [Using the Walrus Operator](#)
- [When Not to Use a List Comprehension in Python](#)
 - [Watch Out for Nested Comprehensions](#)
 - [Choose Generators for Large Datasets](#)
 - [Profile to Optimize Performance](#)
- [Conclusion](#)

내포의 특징

- 파이썬스러운(pythonic) 코딩 방식
- 효용성
 - 한 번 알아두면 쉽게 코딩
 - 속도는 반복보다 빠름
 - 내장 함수 `map()` 보다는 느림

모듈 random

모듈 random

- libWrandom.py

The screenshot shows a Jupyter Notebook window titled 'randuse - Jupyter Notebook' on a localhost. The notebook has three input cells. The first cell contains `import random`. The second cell contains `random?`. The third cell is empty. Below the input cells, the output area displays the documentation for the `random` module:

```
Type: module
String form: <module 'random' from 'D:\\Anaconda3\\lib\\random.py'>
File: d:\\Anaconda3\\lib\\random.py
Docstring:
Random variable generators.

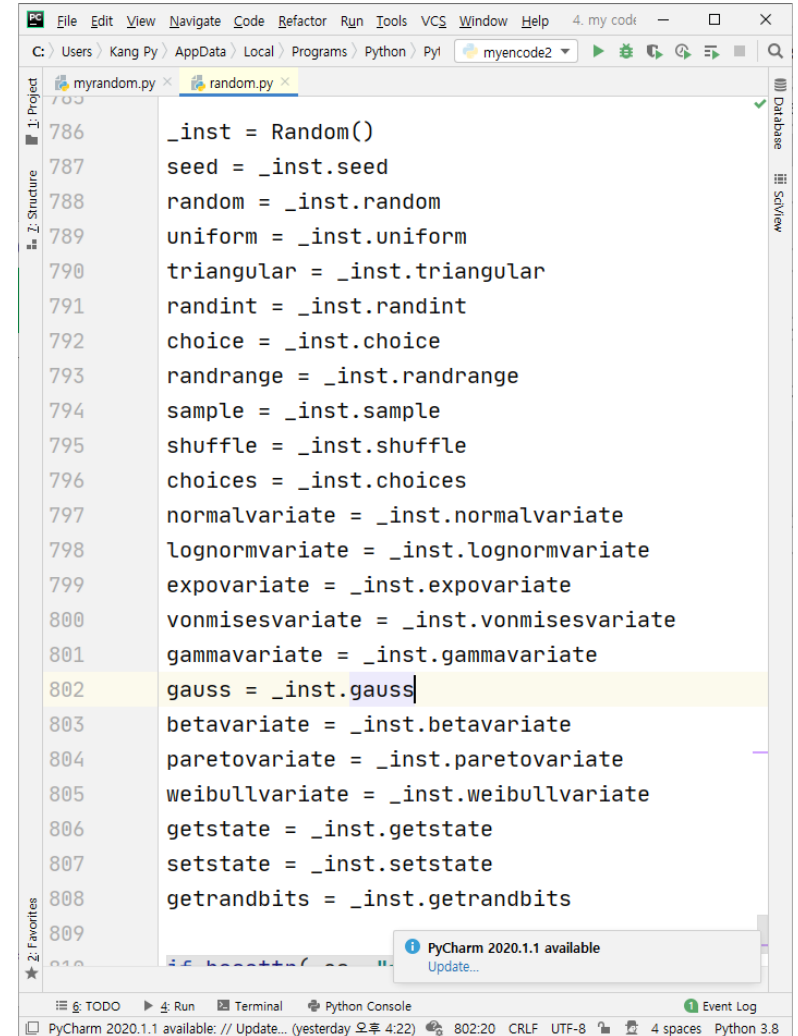
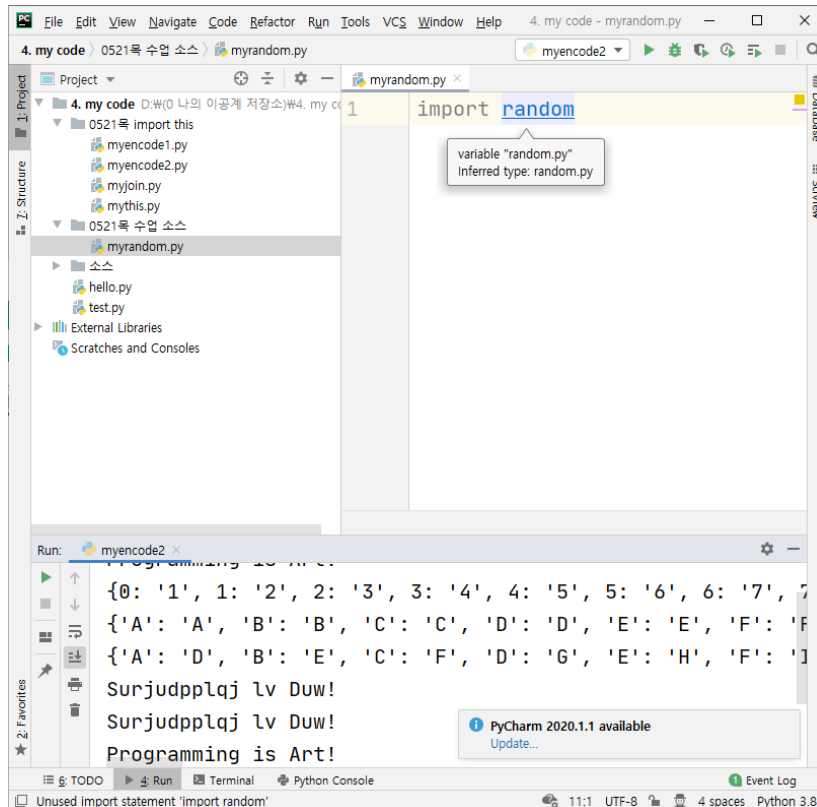
    integers
    -----
        uniform within range

    sequences
    -----
        pick random element
        pick random sample
        pick weighted random sample
        generate random permutation

    distributions on the real line:
    -----
```

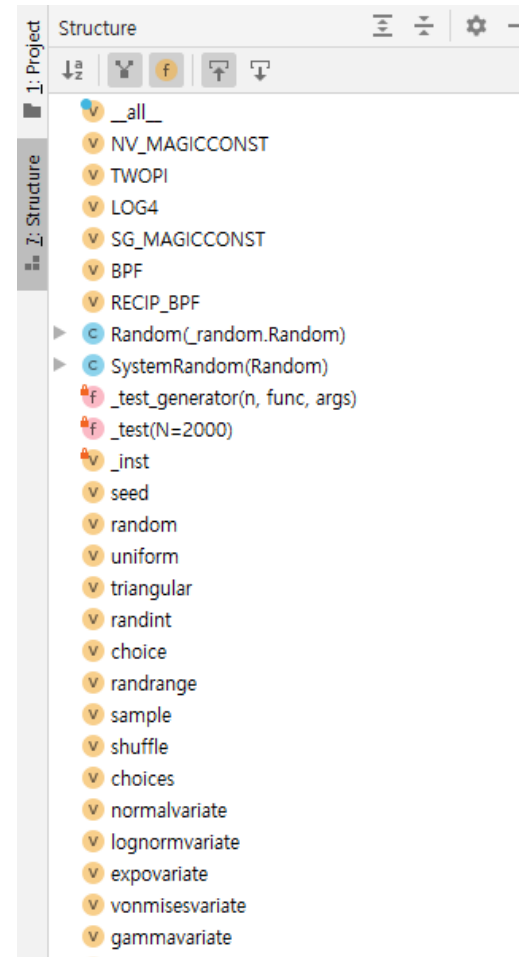
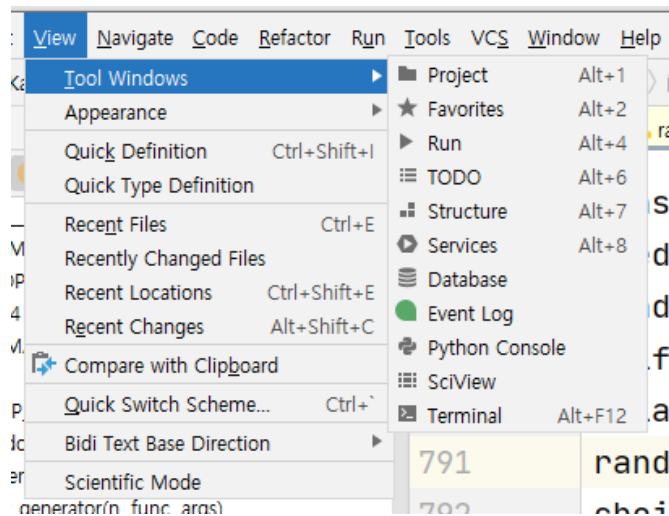
모듈 random.py 소스

- 소스 바로 가기
 - Ctrl + 마우스 클릭

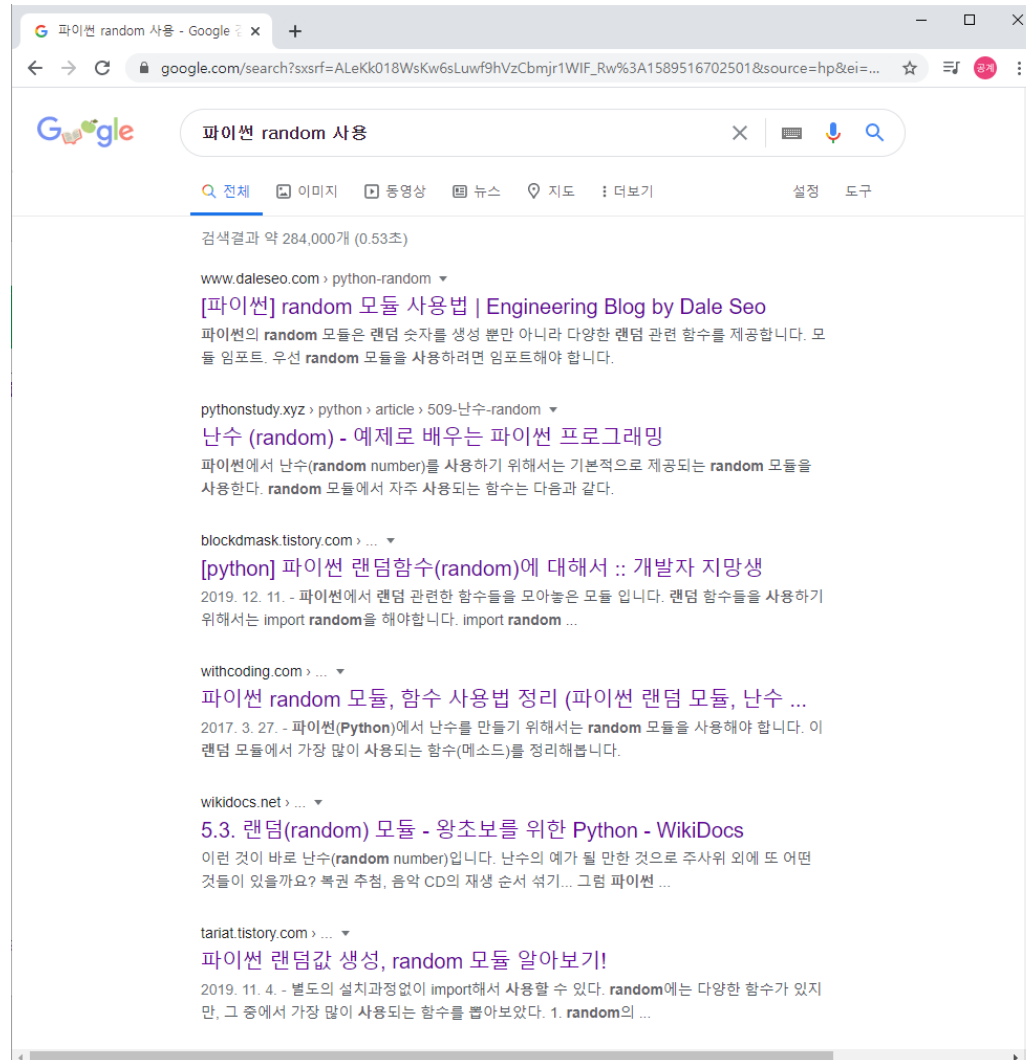


소스의 구조 보기

• Alt + 7



검색



다양한 함수

- **random() 함수**
 - 0부터 1사이의 랜덤 실수를 리턴
- **uniform() 함수**
 - 2개의 숫자 사이의 랜덤 실수를 리턴
- **randint(a, b) 함수**
 - 2개의 숫자 사이의 랜덤 정수를 리턴
- **randrange(a, b) 함수**
 - range(start, stop, step) 함수로 만들어지는 정수 중에 하나를 랜덤하게 리턴
- **choice(리스트) 함수**
 - 랜덤하게 하나의 원소를 선택
- **sample(리스트, 갯수) 함수**
 - 랜덤하게 갯수 원소를 유니크하게 선택
- **shuffle(리스트) 함수**
 - 원소의 순서를 랜덤하게 바꿈
- **seed(정수) 함수**
 - 시드 값 지정, 난수가 일정하게 생성

shuffle과 컴프리헨션의 조합

```
In [28]: items = [1, 2, 3, 4, 5, 6, 7]
         random.shuffle(items)
         items
```

```
Out[28]: [5, 4, 6, 3, 1, 7, 2]
```

```
In [30]: items = [1, 2, 3, 4, 5, 6, 7]
         [random.shuffle(items) for _ in range(5)]
```

```
Out[30]: [None, None, None, None, None]
```

```
In [106]: items = [1, 2, 3, 4, 5, 6, 7]

         def myshuffle(lst):
             random.shuffle(lst)
             return lst
```

```
In [120]: myshuffle(items)
         items
```

```
Out[120]: [7, 6, 3, 1, 2, 5, 4]
```

```
In [125]: [mysuffle(items) for _ in range(5)]
```

```
Out[125]: [[2, 4, 5, 6, 1, 3, 7],
            [2, 4, 5, 6, 1, 3, 7],
            [2, 4, 5, 6, 1, 3, 7],
            [2, 4, 5, 6, 1, 3, 7],
            [2, 4, 5, 6, 1, 3, 7]]
```


다음 코드의 결과 예상은?

• 목적

- 1에서 7까지의 정수를 shuffle를 5번하여 매번 결과를 리스트로 생성

Write code in Python 3.6

```

1 import random
2
3 # 내부적으로 변화더라도 전역 변수인 마지막 값으로
4 items = [1, 2, 3, 4, 5, 6, 7]
5 def myshuffle2(lst):
6     random.shuffle(lst)
7     print(lst)
8     return lst
9
10 my = [myshuffle2(items) for _ in range(5)]
11 print(my)

```

• 예상과는 다른 결과

[[2, 4, 5, 6, 1, 3, 7], [2, 4, 5, 6, 1, 3, 7], [2, 4, 5, 6, 1, 3, 7], [2, 4, 5, 6, 1, 3, 7], [2, 4, 5, 6, 1, 3, 7]]

메모리 내부 모습 참조

• Pythontutor.com 에서 보기

Visualize Python, Java, JavaScript, C, C++, Ruby code execution

북마크바로 지정된 플러그인 북마크가 없습니다. 북마크를 추가해 보세요. 북마크 관리자로 가기

Get live help! These Python Tutor users are asking for help right now. Please volunteer to help!

- user_6eb from Palo Alto, California, US needs help with Python3 - [click to help](#) (active a few seconds ago, requested 7 minutes ago)
- user_134 from Northwood, North Dakota, US needs help with Python3 - [click to help](#) (active a few seconds ago, requested 3 minutes ago)
- user_7f8 from Adelaide, Australia needs help with Python3 - 2 people chatting - [click to help](#) (active a few seconds ago, requested 2 minutes ago)
- user_cf1 from Lake Forest, California, US needs help with Python3 - [click to help](#) (IDLE: last active an hour ago, requested 8 hours ago)
- user_a33 from Toronto, Canada needs help with Python3 - [click to help](#) (IDLE: last active 20 minutes ago, requested 20 minutes ago)

Start private chat

Python 3.6

```

1 import random
2
3 # 내부적으로 변화더라도 전역 변수인 마지막 값으로
4 items = [1, 2, 3, 4, 5, 6, 7]
5 def myshuffle2(lst):
6     random.shuffle(lst)
7     print(lst)
8     return lst
9
10 my = [mysuffle2(items) for _ in range(5)]
11 print(my)

```

[Edit this code](#)

→ line that just executed
→ next line to execute

Done running (38 steps)

[Customize visualization \(NEW!\)](#)

Print output (drag lower right corner to resize)

```

[5, 3, 2, 1, 6, 4, 7]
[3, 5, 6, 4, 7, 2, 1]
[1, 4, 3, 2, 6, 5, 7]
[3, 1, 2, 4, 7, 5, 6]
[1, 6, 7, 4, 2, 3, 5]
[[1, 6, 7, 4, 2, 3, 5], [1, 6, 7, 4, 2, 3, 5], [1, 6, 7, 4, 2, 3, 5], [1, 6, 7, 4, 2, 3, 5], [1, 6, 7, 4, 2, 3, 5]]

```

Frames

Global frame

- random
- items
- mysuffle2
- my

Objects

- module instance
- list [5, 3, 2, 1, 6, 4, 7]
- list [3, 5, 6, 4, 7, 2, 1]
- list [1, 4, 3, 2, 6, 5, 7]
- list [3, 1, 2, 4, 7, 5, 6]
- list [1, 6, 7, 4, 2, 3, 5]
- function myshuffle2(lst)
- list [1, 6, 7, 4, 2, 3, 5]

unsupported features | [setting breakpoints](#) | [hiding variables](#) | [live programming](#)

다음으로 수정해 출력

- 리스트를 깊은 복사로 새로운 리스트 반환

```
import random
items = [1, 2, 3, 4, 5, 6, 7]

# 내부적으로 새로운 값으로 복사해 반환
def myshuffle3(lst):
    random.shuffle(lst)
    print(lst)
    return lst.copy()

my = [mysuffle3(items) for _ in range(5)]
print(my)
```

```
[1, 4, 6, 7, 3, 2, 5]
[7, 4, 2, 5, 6, 3, 1]
[7, 5, 1, 3, 2, 4, 6]
[3, 1, 4, 5, 6, 2, 7]
[1, 7, 6, 2, 5, 3, 4]
[[1, 4, 6, 7, 3, 2, 5], [7, 4, 2, 5, 6, 3, 1], [7, 5, 1, 3, 2, 4, 6], [3, 1, 4, 5, 6, 2, 7], [1, 7, 6, 2, 5, 3, 4]]
```

메모리 내부 모습 확인

Visualize Python, Java, JavaScript, C, C++, Ruby code execution

북마크바로 지정된 폴더에 북마크가 없습니다. 북마크를 추가해 보세요. [북마크 관리자 가기](#)

Get live help! These Python Tutor users are asking for help right now. Please volunteer to help!

- user_ea3 from Toronto, Canada needs help with Python3 - [click to help](#) (active a few seconds ago, requested a few seconds ago)
- user_cf1 from Lake Forest, California, US needs help with Python3 - [click to help](#) (IDLE: last active an hour ago, requested 9 hours ago)
- user_134 from Northwood, North Dakota, US needs help with Python3 - [click to help](#) (IDLE: last active 22 minutes ago, requested 26 minutes ago)

Start private chat

Python 3.6

```

1 import random
2 items = [1, 2, 3, 4, 5, 6, 7]
3
4 # 내부적으로 새로운 값으로 복사해 반환
5 def myshuffle3(lst):
6     random.shuffle(lst)
7     print(lst)
8     return lst.copy()
9
10 my = [mysuffle3(items) for _ in range(5)]
11 print(my)

```

[Edit this code](#)

→ line that just executed
→ next line to execute

Done running (38 steps)

[Customize visualization \(NEW!\)](#)

Print output (drag lower right corner to resize)

```

[5, 3, 2, 1, 6, 4, 7]
[3, 5, 6, 4, 7, 2, 1]
[1, 4, 3, 2, 6, 5, 7]
[3, 1, 2, 4, 7, 5, 6]
[1, 6, 7, 4, 2, 3, 5]
[[5, 3, 2, 1, 6, 4, 7], [3, 5, 6, 4, 7, 2, 1], [1, 4, 3, 2, 6, 5, 7], [3, 1, 2, 4, 7, 5, 6], [1, 6, 7, 4, 2, 3, 5]]

```

Frames

Global frame

- random
- items
- mysuffle3
- my

Objects

module instance

list

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 6 | 7 | 4 | 2 | 3 | 5 |

function myshuffle3(lst)

list

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 5 | 3 | 2 | 1 | 6 | 4 | 7 |

list

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 3 | 5 | 6 | 4 | 7 | 2 | 1 |

list

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 4 | 3 | 2 | 6 | 5 | 7 |

list

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 3 | 1 | 2 | 4 | 7 | 5 | 6 |

list

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 6 | 7 | 4 | 2 | 3 | 5 |

list

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |