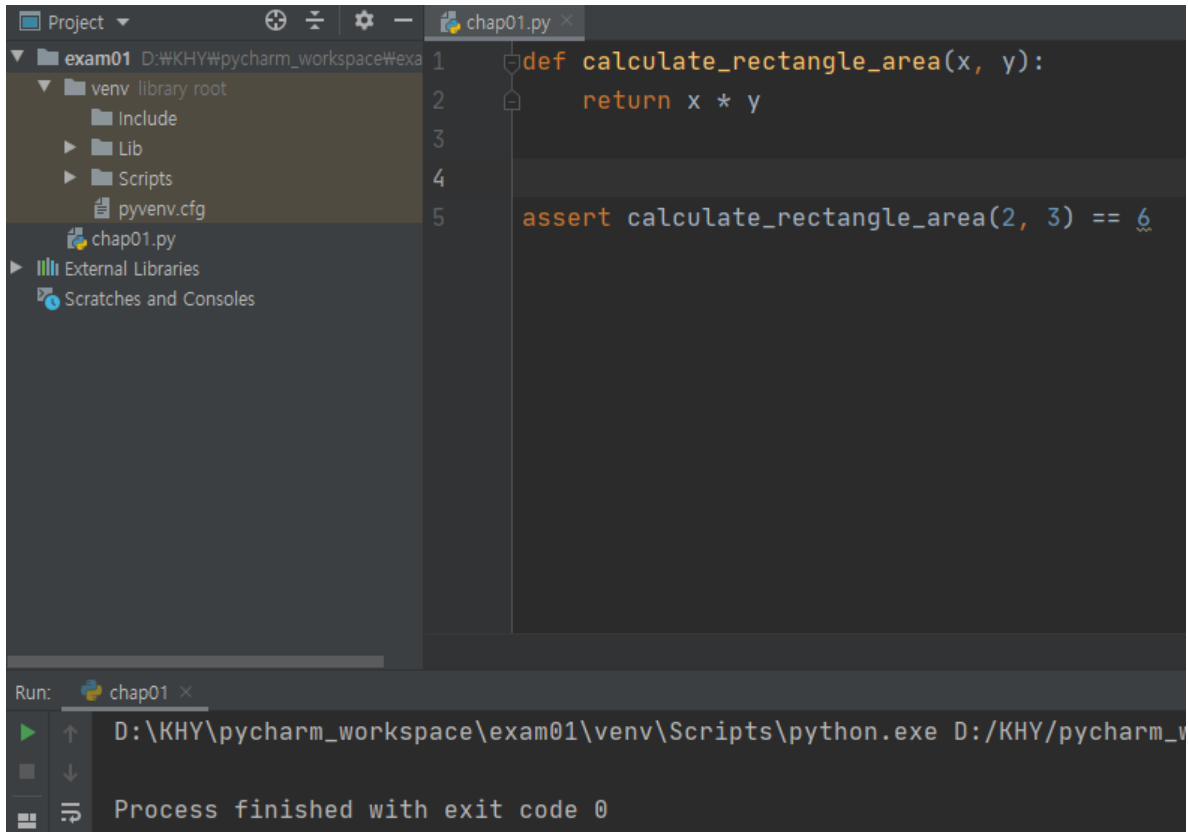


# 파이썬에서의 함수

- def

```
def 함수이름 (인자1, 인자2, ...옵션)
    실행문1
    실행문2
    return 반환값(옵션)
```

리턴타입을 명시해주지 않음



함수 정의 후 실행문 쓸 땐 2줄 이상 띄우는 것이 좋음(가독성)

assert문(가정 설정문): 뒤의 조건이 True가 아니면 AssertionError 발생 시킴(단위 테스트)

The image shows the PyCharm IDE interface. The left sidebar displays the project structure for 'exam01', including a virtual environment 'venv' with its 'library root' containing 'Include', 'Lib', and 'Scripts' folders, and a 'pyvenv.cfg' file. The main editor window shows a file named 'chap01.py' with the following code:

```
1 def calculate_rectangle_area(x, y):
2     return x * y
3
4
5 assert calculate_rectangle_area(2, 3) == 7
```

A yellow lightbulb icon is positioned above line 5, indicating a warning or error. The bottom 'Run' console shows the execution command and a traceback for an 'AssertionError'.

```
Run: chap01 x
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_wor
Traceback (most recent call last):
  File "D:/KHY/pycharm_workspace/exam01/chap01.py", line 5, in <module>
    assert calculate_rectangle_area(2, 3) == 7
AssertionError
```

실행문이 틀렸을 경우

The image shows the PyCharm IDE interface with the same project structure as the first image. The main editor window shows the 'chap01.py' file with the following code:

```
1 def calculate_rectangle_area(x, y):
2     return x * y
3
4
5 assert calculate_rectangle_area(2, 3) == 7, '이 부분이 에러'
```

A yellow lightbulb icon is positioned above line 5. The bottom 'Run' console shows the execution command and a traceback for an 'AssertionError' with a custom message.

```
Run: chap01 x
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_workspace/exam01/
Traceback (most recent call last):
  File "D:/KHY/pycharm_workspace/exam01/chap01.py", line 5, in <module>
    assert calculate_rectangle_area(2, 3) == 7, '이 부분이 에러'
AssertionError: 이 부분이 에러
```

assert문에 위 처럼 작성하면 에러를 좀 더 쉽게 찾을 수 있음

```
1 def calculate_rectangle_area(x, y):
2     return x * y
3
4 # assert calculate_rectangle_area(2, 3) == 7. '이 부분 에러'
5
6 rectangle_x = 10
7 rectangle_y = 20
8 print("사각형 x의 길이: ", rectangle_x)
9 print("사각형 y의 길이: ", rectangle_y)
10 print("사각형의 넓이: ", calculate_rectangle_area(rectangle_x, rectangle_y))
```

Run: chap01 x

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm\_workspace/exam01/chap01.py

사각형 x의 길이: 10  
사각형 y의 길이: 20  
사각형의 넓이: 200

## 실행결과

- 변수의 scope 영역

```
1 def shopping_cart(goods): # 곱지는 로컬 변수
2     goods.append('coupon')
3     goods = [1, 2]
4     print("In shopping_cart", goods)
5
6 shopping_list = ['bean', 'salt', 'beef'] # 리스트의 레퍼런스를 갖는 글로벌 변수
7 shopping_cart(shopping_list)
8 print("Goods I bought: ", shopping_list)
```

Run: chap01 x

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm\_workspace/exam01/chap01.py

In shopping\_cart [1, 2]  
Goods I bought: ['bean', 'salt', 'beef', 'coupon']

함수 인자에 레퍼런스 타입이 들어가면 call by reference로 호출

```
public class MyClass{ // java 예시
    private int field1;
    ...
    public setField1(int field1){
        this.field1 = field1; // 뒤의 field1은 로컬 변수(함수 내에서 선언)
    }
}
```

1. 스코프 영역이 다름(goods와 shopping\_list)
2. 매개인자 타입이 레퍼런스 타입이면 오리지널 객체에 수정을 해줄 수 있음(goods를 [1, 2] 리스트로 재참조 했으므로 shopping\_list와의 연결고리가 없어짐)

stack에 shopping\_list / Heap에 리스트 내용

```
1 def test(x):
2     print(x)
3     t = 20
4     a = 300
5     print("In test t, a", t, a)
6
7     a = 10
8     test(a)
9     print("In main x:", a)
10    # print("In main t: ", t) # t에러
```

Run: chap01 x

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pyc  
10  
In test t, a 20 300  
In main x: 10

```
1 def f():
2     s = "I love Doosan!"
3     print(s)
4
5     s = "I love Hanhwa!"
6     f()
7     print(s)
```

Run: chap01 x

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe  
I love Doosan!  
I love Hanhwa!

static 타입의 언어에서 메소드 혹은 함수에서 글로벌 변수에 접근이 가능함

하지만 파이썬과 같은 다이내믹 타입의 언어에서는 타입으로 변수를 만들지 않고 바로 만들기 때  
문에 함수 내부에서 글로벌 변수를 접근하고 싶을 때 문제 발생

```
1 def f():
2     global s
3     s = "I love Doosan!"
4     print(s)
5
6 s = "I love Hanhwa!"
7 f()
8 print(s)
```

Run: chap01 ×

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe

I love Doosan!

I love Doosan!

global ~: 앞으로의 ~변수는 글로벌 영역의 ~변수라는 것을 선언

```
1 def calculate(x, y):
2     total = x + y
3     print("In Function")
4     print("a: ", str(a), "b: ", str(b), "a + b: ", str(a + b), "total: ", str(total))
5     return total
6
7
8 a = 5
9 b = 7
10 total = 0
11 print("In Main")
12 print("a: ", str(a), "b: ", str(b), "a + b: ", str(a + b))
13
14 sum = calculate(a, b)
15 print("After Calculation")
16 print("Total: ", str(total), "Sum: ", str(sum))
17
```

Run: chap01 ×

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm\_workspace/exam01/chap01.py

In Main

a: 5 b: 7 a + b: 12

In Function

a: 5 b: 7 a + b: 12 total: 12

After Calculation

Total: 0 Sum: 12

```
1 def print_name(my_name, your_name):
2     print("Hello {0}, My name is {1}".format(your_name, my_name))
3
4
5 print_name("Tom", "Jane")
6 print_name(your_name="Jane", my_name="Tom")
```

Run: chap01 ×

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm\_workspace/exam01/chap01.py

Hello Jane, My name is Tom

Hello Jane, My name is Tom

직접 명시해서 파라미터에 참조값을 넣을 수 있음



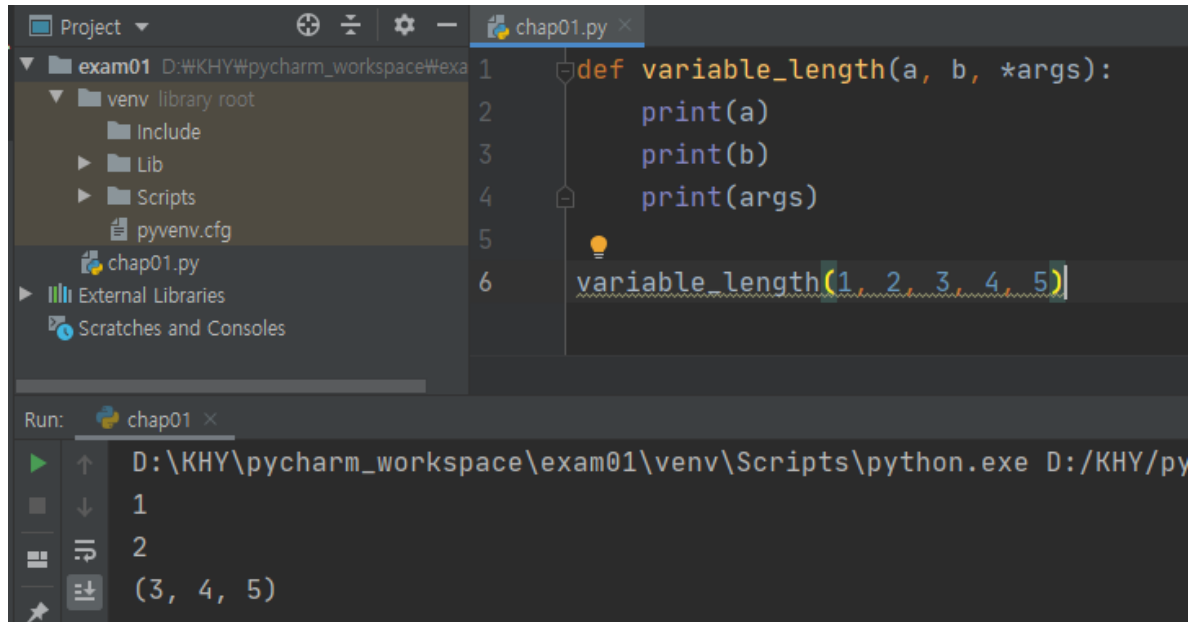
The screenshot shows the PyCharm IDE with a project named 'exam01'. The file explorer on the left shows the project structure, including a 'venv' directory. The main editor displays a Python script 'chap01.py' with the following code:

```
1 def print_name(my_name, your_name="Jane"):
2     print("Hello {0}, My name is {1}".format(your_name, my_name))
3
4
5 print_name("Tom", "Jane")
6 print_name("Tom")
```

The Run console at the bottom shows the execution of the script, outputting:

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_workspace/exam01/chap01.py
Hello Jane, My name is Tom
Hello Jane, My name is Tom
```

디폴트값 주기



The screenshot shows the PyCharm IDE with a project named 'exam01'. The file explorer on the left shows the project structure, including a 'venv' directory. The main editor displays a Python script 'chap01.py' with the following code:

```
1 def variable_length(a, b, *args):
2     print(a)
3     print(b)
4     print(args)
5
6 variable_length(1, 2, 3, 4, 5)
```

The Run console at the bottom shows the execution of the script, outputting:

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/py
1
2
(3, 4, 5)
```

가변인자(\*~)

가변인자는 항상 마지막에 위치해야 함

The image shows the PyCharm IDE with a project named 'exam01'. The file 'chap01.py' is open, showing the following code:

```

1 s = "BLACK LIVES MATTER"
2 print(s.upper())
3 print(s.lower())
4 print(s.title())
5 print(s.capitalize())
6 print(s.count("T"))
7 print(s.startswith("B"))
8
9 k = "123"
10 print(k.isdigit())

```

The Run window shows the output of the script:

```

D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe
BLACK LIVES MATTER
black lives matter
Black Lives Matter
Black lives matter
2
True
True

```

isdigit() : 숫자로 되어 있는 지 검사

The image shows the PyCharm IDE with a project named 'exam01'. The file 'chap01.py' is open, showing the following code:

```

1 print(1, 2, 3)
2 print("%d %d %d" % (1, 2, 3)) # ,가 없음
3 print("%s--%s--%s" % ('abc', 'def', 'ghi')) # ,가 없음
4 print("{} {} {}".format('abc', 'def', 'ghi')) # String 클래스의 format메서드
5 print("{1} {0} {2}".format('abc', 'def', 'ghi'))
6 print(f"{'def'} {'abc'} {'ghi'}") # formatted String
7 print("{0:>10s}".format('python')) # 전체 10칸을 잡고 우측정렬
8 print("{0:<10s}".format('python')) # 전체 10칸을 잡고 좌측정렬
9 print(f"{'python':>10s}") # formatted String에도 정렬 사용 가능
10 print(f"{'python':<10s}")

```

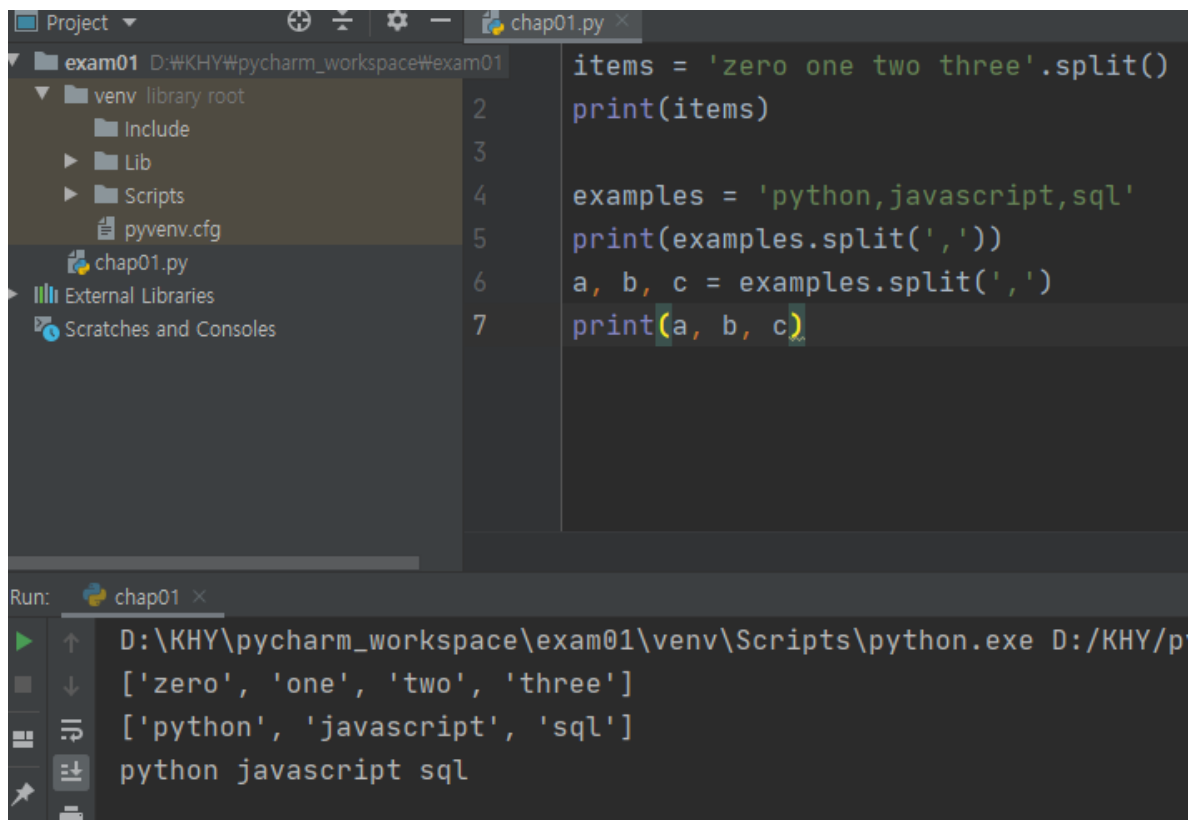
The Run window shows the output of the script:

```

D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_workspace/exam01/chap01.py
1 2 3
1 2 3
abc--def--ghi
abc def ghi
def abc ghi
def abc ghi
python
python
python
python
python

```

print 출력 방법



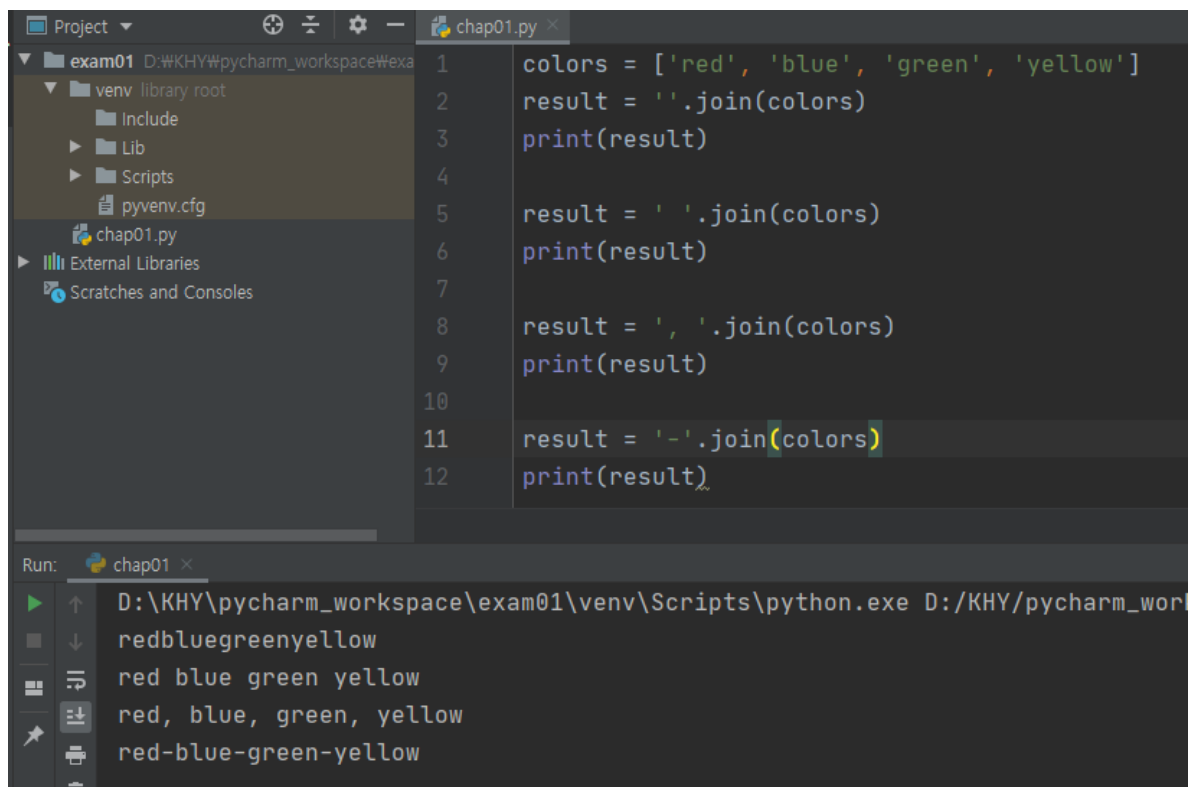
```
Project ▾
exam01 D:\KHY\pycharm_workspace\exam01
├── venv library root
│   ├── Include
│   ├── Lib
│   ├── Scripts
│   └── pyvenv.cfg
├── chap01.py
├── External Libraries
└── Scratches and Consoles

2 items = 'zero one two three'.split()
3 print(items)
4
5 examples = 'python,javascript,sql'
6 print(examples.split(','))
7 a, b, c = examples.split(',')
8 print(a, b, c)
```

Run: chap01 ×

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/p
['zero', 'one', 'two', 'three']
['python', 'javascript', 'sql']
python javascript sql
```

split()의 결과는 리스트



```
Project ▾
exam01 D:\KHY\pycharm_workspace\exam01
├── venv library root
│   ├── Include
│   ├── Lib
│   ├── Scripts
│   └── pyvenv.cfg
├── chap01.py
├── External Libraries
└── Scratches and Consoles

1 colors = ['red', 'blue', 'green', 'yellow']
2 result = ''.join(colors)
3 print(result)
4
5 result = ' '.join(colors)
6 print(result)
7
8 result = ', '.join(colors)
9 print(result)
10
11 result = '-'.join(colors)
12 print(result)
```

Run: chap01 ×

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_wor
redbluegreenyellow
red blue green yellow
red, blue, green, yellow
red-blue-green-yellow
```

join() : 리스트를 문자열로 구분자를 지정해서 넣는 메서드인 듯

## 파이썬의 자료구조



## ● 파이썬이 제공하는 자료구조

자료구조	특징
Stack	LIFO
Queue	FIFO
Tuple	리스트와 비슷, 데이터 변경이 안됨
Set	데이터 중복을 허용하지 않음
Dictionary	Key:Value 형태로 저장
collections 모듈	위 자료구조를 지원하는 파이썬 내장 모듈

Set : 집합(중복데이터 허용하지 않음), 순서가 없음(따라서 첨자 없음)

- stack

```

1 # 리스트를 스택 구조로 쓸 때 append()와 pop() 사용
2 a = [1, 2, 3, 4, 5]
3 a.append(10)
4 print(a)
5 a.append(20)
6 print(a)
7 print(a.pop())
8 print(a)
9 print(a.pop())
10 print(a)

```

Run: chap01 ×

```

D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_wor
[1, 2, 3, 4, 5, 10]
[1, 2, 3, 4, 5, 10, 20]
20
[1, 2, 3, 4, 5, 10]
10
[1, 2, 3, 4, 5]

```

append() pop()

```

1 word = input("Input a word: ")
2 word_list = list(word)
3 print(word_list)
4
5
6 result = []
7 for _ in range(len(word_list)): # _로 변수 할당을 하지 않겠다는 코드(range값 숫자가 나오므로)
8     result.append(word_list.pop())
9
10 print(result)
11 print(word[::-1])

```

Run: chap01 ×

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm\_workspace/exam01/chap01.py

Input a word: How you like that

['h', 'o', 'w', ' ', 'y', 'o', 'u', ' ', 'l', 'i', 'k', 'e', ' ', 't', 'h', 'a', 't']

['t', 'a', 'h', 't', ' ', 'e', 'k', 'i', 'l', ' ', 'u', 'o', 'y', ' ', ' ', 'w', 'o', 'h']

taht ekil uoy woh

- Queue

```

1 # 리스트를 큐 구조로 쓸 때 pop()의 인덱스 값을 0으로 줌
2 a=[1, 2, 3, 4, 5]
3 a.append(10)
4 a.append(20)
5 print(a)
6 print(a.pop(0))
7 print(a.pop(0))
8 print(a)

```

Run: chap01 ×

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm\_workspace/exam01/chap01.py

[1, 2, 3, 4, 5, 10, 20]

1

2

[3, 4, 5, 10, 20]

append() pop(0)

- 튜플

```

1 # 튜플
2 t = (1, 2, 3)
3 print(t+t, t*2)
4 print(len(t))
5 t[2] = 4
6

```

Run: chap01 ×

D:\KHY\pycharm\_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm\_workspace/exam01/chap01.py

Traceback (most recent call last):

File "D:/KHY/pycharm\_workspace/exam01/chap01.py", line 5, in <module>

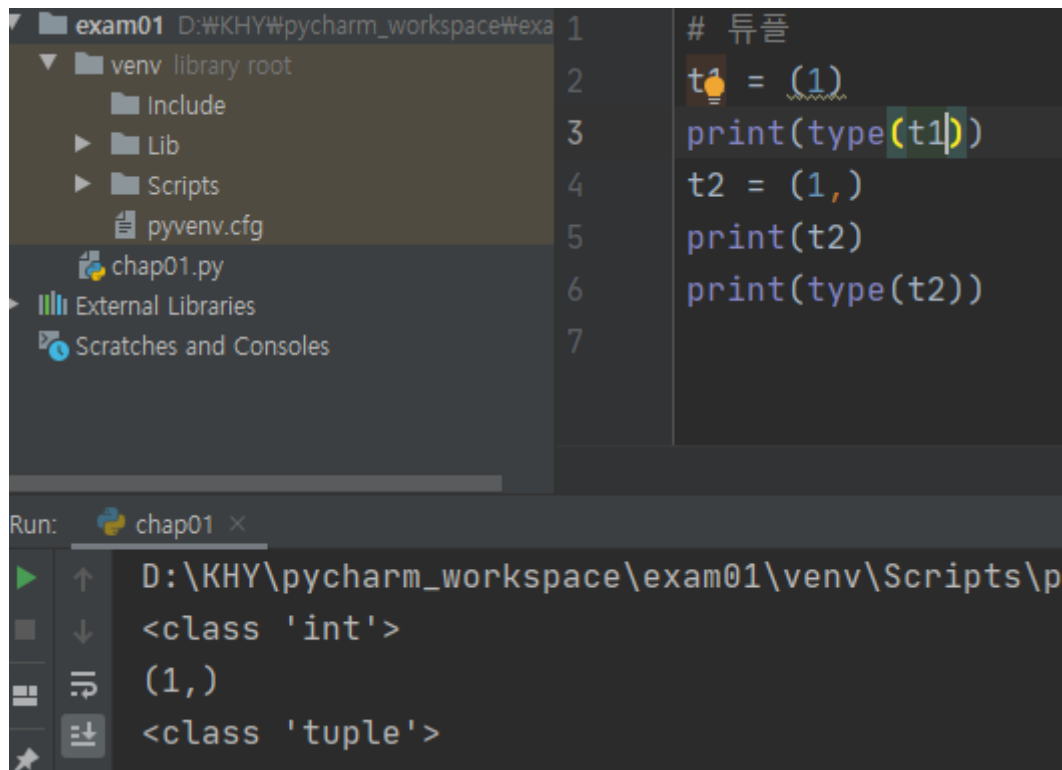
t[2] = 4

TypeError: 'tuple' object does not support item assignment

(1, 2, 3, 1, 2, 3) (1, 2, 3, 1, 2, 3)

3

## 항목 바꾸기 불가능



The image shows a PyCharm IDE window with a project named 'exam01'. The left sidebar shows the project structure with a 'venv' directory. The main editor shows a Python file 'chap01.py' with the following code:

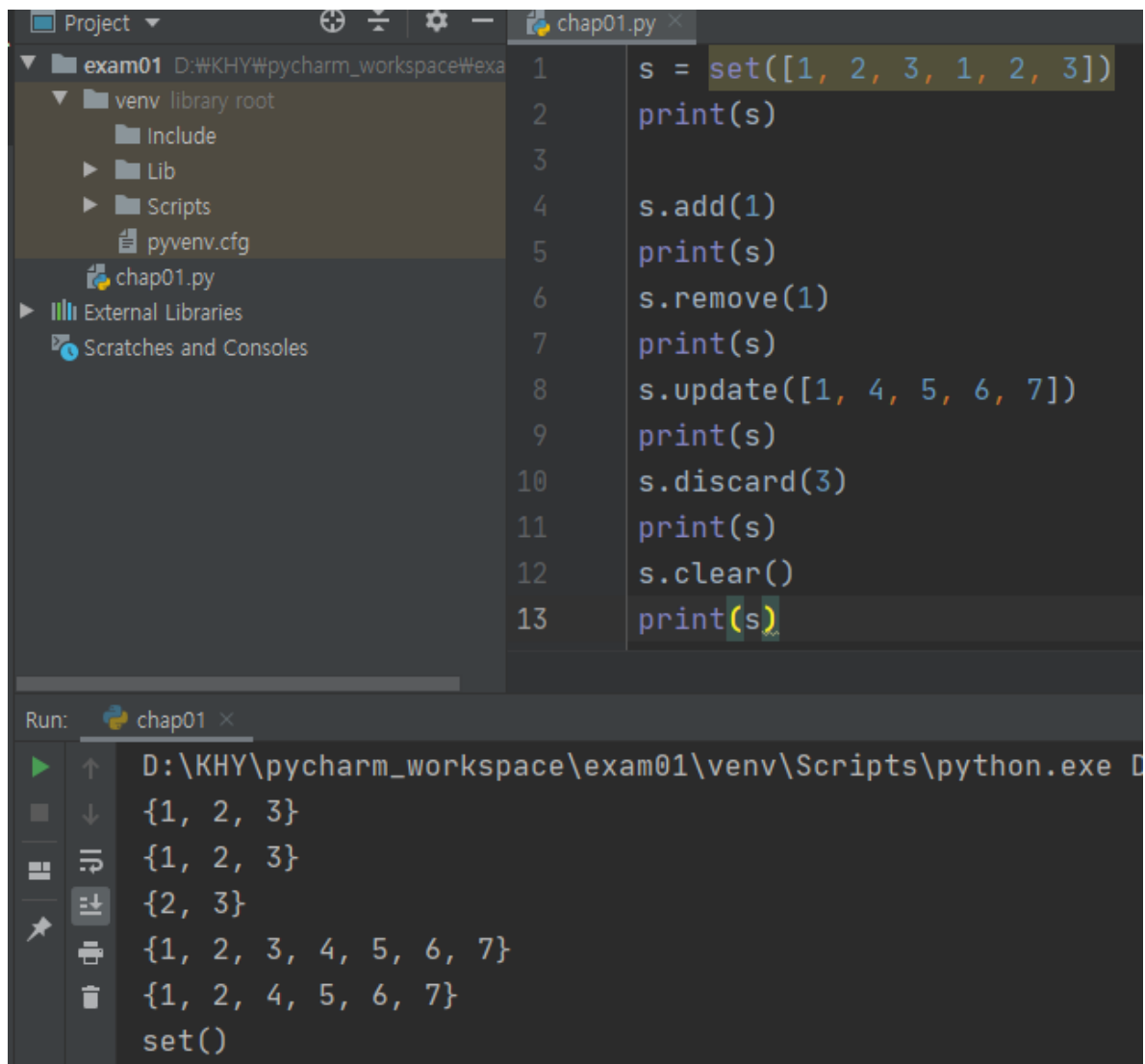
```
1 # 튜플
2 t1 = (1)
3 print(type(t1))
4 t2 = (1,)
5 print(t2)
6 print(type(t2))
7
```

The Run console at the bottom shows the output of the code:

```
Run: chap01 ×
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe
<class 'int'>
(1,)
<class 'tuple'>
```

## 원소가 하나 일 때 콤마 필요

- set



The image shows a PyCharm IDE window with a project named 'exam01'. The left sidebar shows the project structure with a 'venv' directory. The main editor shows a Python file 'chap01.py' with the following code:

```
1 s = set([1, 2, 3, 1, 2, 3])
2 print(s)
3
4 s.add(1)
5 print(s)
6 s.remove(1)
7 print(s)
8 s.update([1, 4, 5, 6, 7])
9 print(s)
10 s.discard(3)
11 print(s)
12 s.clear()
13 print(s)
```

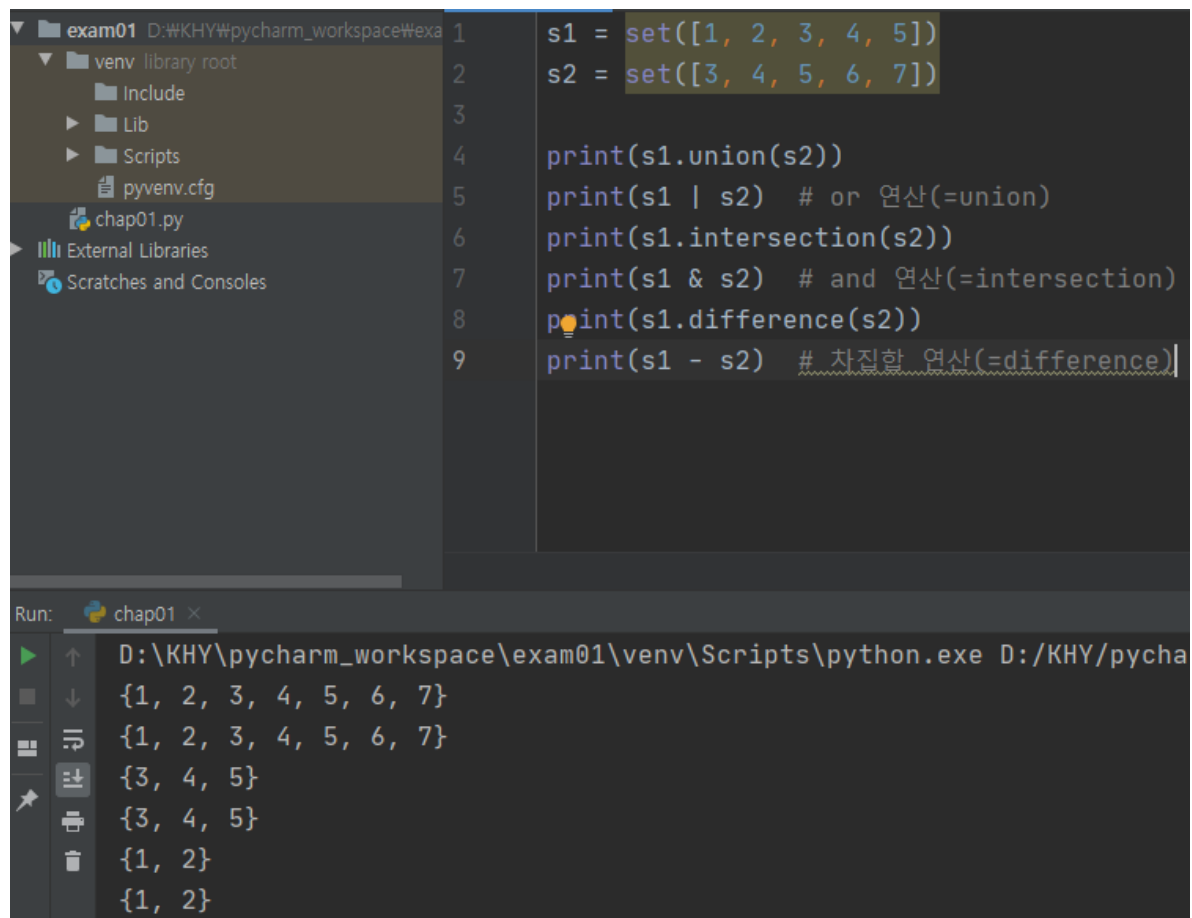
The Run console at the bottom shows the output of the code:

```
Run: chap01 ×
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D
{1, 2, 3}
{1, 2, 3}
{2, 3}
{1, 2, 3, 4, 5, 6, 7}
{1, 2, 4, 5, 6, 7}
set()
```

중복되지 않음

discard(): 지우려는 요소가 없어도 정상종료

remove(): 지우려는 요소가 없으면 KeyError



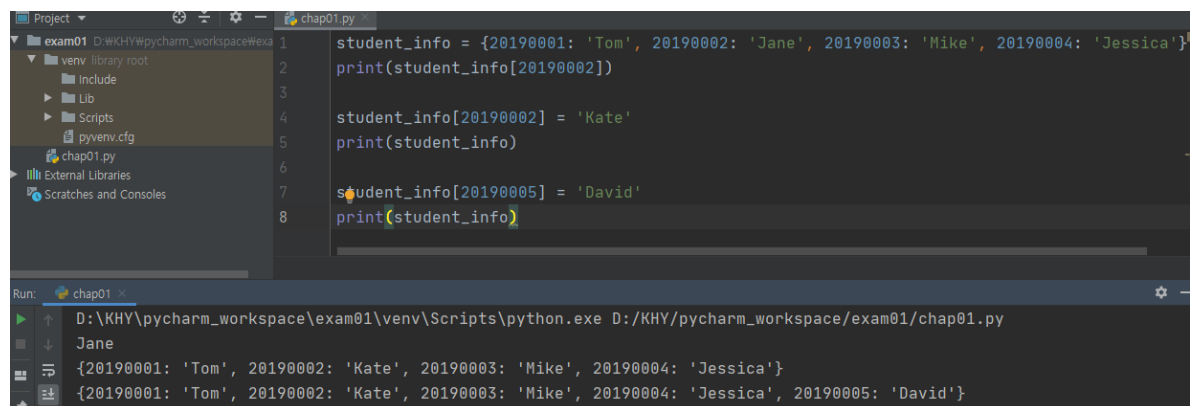
The screenshot shows a PyCharm IDE with a project named 'exam01'. The file explorer on the left shows a 'venv' directory and a 'chap01.py' file. The main editor displays the following Python code:

```
1 s1 = set([1, 2, 3, 4, 5])
2 s2 = set([3, 4, 5, 6, 7])
3
4 print(s1.union(s2))
5 print(s1 | s2) # or 연산(=union)
6 print(s1.intersection(s2))
7 print(s1 & s2) # and 연산(=intersection)
8 print(s1.difference(s2))
9 print(s1 - s2) # 차집합 연산(=difference)
```

The Run console at the bottom shows the output of the code:

```
Run: chap01 x
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycha
{1, 2, 3, 4, 5, 6, 7}
{1, 2, 3, 4, 5, 6, 7}
{3, 4, 5}
{3, 4, 5}
{1, 2}
{1, 2}
```

- dictionary



The screenshot shows a PyCharm IDE with a project named 'exam01'. The file explorer on the left shows a 'venv' directory and a 'chap01.py' file. The main editor displays the following Python code:

```
1 student_info = {'20190001': 'Tom', '20190002': 'Jane', '20190003': 'Mike', '20190004': 'Jessica'}
2 print(student_info['20190002'])
3
4 student_info['20190002'] = 'Kate'
5 print(student_info)
6
7 student_info['20190005'] = 'David'
8 print(student_info)
```

The Run console at the bottom shows the output of the code:

```
Run: chap01 x
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_workspace/exam01/chap01.py
Jane
{'20190001': 'Tom', '20190002': 'Kate', '20190003': 'Mike', '20190004': 'Jessica'}
{'20190001': 'Tom', '20190002': 'Kate', '20190003': 'Mike', '20190004': 'Jessica', '20190005': 'David'}
```

딕셔너리 벨류를 명시적으로 변경 및 추가

```
Project ▾
exam01 D:\KHY\pycharm_workspace\exam01
├── venv
│   ├── library root
│   ├── include
│   ├── Lib
│   ├── Scripts
│   └── pyenv.cfg
└── chap01.py
External Libraries
Scratches and Consoles

1 country_code = {'USA': 1, 'Korea': 82, 'China': 86, 'Malaysia': 60}
2 print(country_code)
3
4 print(country_code.keys())
5
6 country_code['German'] = 49
7 print(country_code)
8
9 print(country_code.values())
10 print(country_code.items())
11 print()
12
13 for k, v in country_code.items(): # 딕셔너리의 키와 값을 이렇게 받을 수 있음
14     print('Key:', k)
15     print('Value:', v)
16
17 print('Korea' in country_code.keys())
18 print(85 in country_code.values())
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_workspace/exam01/chap01.py
{'USA': 1, 'Korea': 82, 'China': 86, 'Malaysia': 60}
dict_keys(['USA', 'Korea', 'China', 'Malaysia'])
{'USA': 1, 'Korea': 82, 'China': 86, 'Malaysia': 60, 'German': 49}
dict_values([1, 82, 86, 60, 49])
dict_items([('USA', 1), ('Korea', 82), ('China', 86), ('Malaysia', 60), ('German', 49)])

Key: USA
Value: 1
Key: Korea
Value: 82
Key: China
Value: 86
```

```
Key: Malaysia
Value: 60
Key: German
Value: 49
True
False
```

key() 키를 리스트를 만들어 줌

values() 값을 리스트로 만들어 줌

items() 키와 값 쌍을 튜플 타입으로 만든 후 리스트로 만들어 줌

in 연산자로 참/거짓 값 판별

## Collection 모듈

- 기본 자료구조를 확장하여 미리 제작하고 파이썬 모듈로 제공
- deque, OrderedDict, defaultdict, Counter, namedtuple
- defaultdict과 Counter를 많이 사용함

### deque

```

from collections import deque # collections라는 컬렉션 안의 deque 객체 사용
💡
deque_list = deque() # deque은 양쪽 모두 입출력이 가능(queue는 한 쪽만 입출력 가능)
for i in range(5):
    deque_list.append(i)

print(deque_list)

print(deque_list.pop())
print(deque_list.pop())
print(deque_list)

deque_list.appendleft(5) # 양쪽 입출력이 가능하므로 appendleft() 메서드 이용
deque_list.appendleft(6)
print(deque_list)

print(deque_list.popleft()) # 양쪽 입출력이 가능하므로 popleft() 메서드 이용
print(deque_list.popleft())
print(deque_list)

```

```

deque([0, 1, 2, 3, 4])
4
3
deque([0, 1, 2])
deque([6, 5, 0, 1, 2])
6
5
deque([0, 1, 2])

Process finished with exit code 0
|

```

```

from collections import deque # collections라는 컬렉션 안의 deque 객체 사용

deque_list = deque() # deque은 양쪽 모두 입출력이 가능(queue는 한 쪽만 입출력 가능)
for i in range(5):
    deque_list.append(i)

print(deque_list)

deque_list.rotate(2)
print(deque_list)

deque_list.rotate(-2)
print(deque_list)
|
deque_list.extend([5, 6, 7])
print(deque_list)

deque_list.extendleft([9, 10])
print(deque_list)

```

```

D:\KHY\pycharm_workspace\exam01\venv\Scripts>python deque.py
deque([0, 1, 2, 3, 4])
deque([3, 4, 0, 1, 2])
deque([0, 1, 2, 3, 4])
deque([0, 1, 2, 3, 4, 5, 6, 7])
deque([10, 9, 0, 1, 2, 3, 4, 5, 6, 7])

```

rotate(n) : 회전함수, 리스트 내부에선 n칸 만큼 회전

## OrderedDict

```

from collections import OrderedDict # 집어넣은 순서를 보장해주는 OrderedDict

d = dict() # 딕셔너리 타입의 객체 생성
d['x'] = 100 # 키와 벨류 명시적으로 작성
d['y'] = 200
d['z'] = 300
d['a'] = 400
print(d)

od = OrderedDict(sorted(d.items(), key=lambda x: x[0]))
# items()로 인해 (키:벨류)로 튜플이 들어오므로
# x의 0번 째(키)로(lambda x: x[0] 부분) sort 하겠다는 뜻

print()
print(od)

od_list = sorted(od.items(), key=lambda x: x[1])
print()
print(od_list)

```

```

D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycha
{'x': 100, 'y': 200, 'z': 300, 'a': 400}

OrderedDict([('a', 400), ('x', 100), ('y', 200), ('z', 300)])

[('x', 100), ('y', 200), ('z', 300), ('a', 400)]

```

잘 쓰이지는 않음

## defaultdict

```

from collections import defaultdict

d = defaultdict(lambda: 100) # 기본 디폴트 값을 100으로 줌
print(d['first']) # 디폴트 값이 있기 때문에 아무 이름으로 지어서 출력해도 나옴

s = [('yellow', 1), ('blue', 2), ('yellow', 3), ('blue', 4), ('red', 1)]
# 튜플의 리스트 s
d2 = defaultdict(list) # d2의 타입을 리스트로 만들

d3 = defaultdict(int) # 인티저 타입의 기본 디폴트 값은 0으로 되어있음
print(d3['aaa'])

for k, v in s:
    d2[k].append(v) # d2가 리스트 타입이기 때문에 append() 가능

print(d2.items())

```



```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm
100
0
dict_items([('yellow', [1, 3]), ('blue', [2, 4]), ('red', [1])])
```

어떤 값을 누적할 때 많이 쓰임

## Counter

```
from collections import Counter

c = Counter('scientist') # 카운팅만을 위한 것
print(c)

print(c['i'])
print(c['n'])
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:
Counter({'s': 2, 'i': 2, 't': 2, 'c': 1, 'e': 1, 'n': 1})
2
1
```

```
from collections import Counter

c = Counter({'red': 4, 'blue': 2}) # 카운팅만을 위한 것
print(c)

print(list(c.elements())) # getlist를 만드는 것과 같음
```

```
Counter({'red': 4, 'blue': 2})
['red', 'red', 'red', 'red', 'blue', 'blue']
```

```
from collections import Counter

c = Counter(cats=4, dogs=6) # 카운팅만을 위한 것
print(c)

print(list(c.elements())) # getlist를 만드는 것과 같음
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pycharm_workspace/e
Counter({'dogs': 6, 'cats': 4})
['cats', 'cats', 'cats', 'cats', 'dogs', 'dogs', 'dogs', 'dogs', 'dogs', 'dogs']
```

키워드 매개변수

```
from collections import Counter

c = Counter(a=4, b=2, c=0, d=-2) # 0과 음수는 False로 음
d = Counter(a=1, b=2, c=3, d=4)

print(c + d)
print(c & d)
print(c | d)

c.subtract(d)
print(c)
```

```
Counter({'a': 5, 'b': 4, 'c': 3, 'd': 2})
Counter({'b': 2, 'a': 1})
Counter({'a': 4, 'd': 4, 'c': 3, 'b': 2})
Counter({'a': 3, 'b': 0, 'c': -3, 'd': -6})
```

## 실습

```
from collections import defaultdict
from collections import OrderedDict

text = """A press release is the quickest and easiest way to get free publicity. If
well written, a press release can result in multiple published articles about your
firm and its products. And that can mean new prospects contacting you
asking you to sell to them. ...""".lower().split()
print(text)

word_count = defaultdict(lambda: 0) # 단어의 출현 갯수를 카운팅하는 것이므로 defaultdict
# lambda : 0 은 값을 0으로 주겠다(int로 사용해도 무방)

for word in text:
    word_count[word] += 1
print(word_count)

for k, v in OrderedDict(sorted(word_count.items(), key=lambda t: t[1], reverse=True)).items():
    print(k, v)
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\
['a', 'press', 'release', 'is', 'the', 'quick']
defaultdict(<function <lambda> at 0x000002185
and 3
to 3
a 2
press 2
release 2
can 2
you 2
is 1
the 1
```

## 리스트 심화

---

- 리스트 컴프리헨션(한 줄 코드로 이뤄져있어 실행속도 향상)

```
result = []
for i in range(10):
    result.append(i)
print(result)
print()

result2 = [i for i in range(10)] # list comprehension
print(result2)
```

```
D:\KHY\pycharm_workspace\exam01\venv\
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
result = []
for i in range(10):
    if i % 2 == 0:
        result.append(i)
print(result)
print()

result2 = [i for i in range(10) if i % 2 == 0] # list comprehension
print(result2)
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe
[0, 2, 4, 6, 8]

[0, 2, 4, 6, 8]
|
```

```
result3 = [i if i % 2 == 0 else 99 for i in range(10)] # list comprehension
# else문이 추가되면 for문이 뒤로 빠짐
print(result3)
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe
[0, 99, 2, 99, 4, 99, 6, 99, 8, 99]
```

elif 문은 사용할 수 없음

```
word1 = 'Hello'
word2 = 'world'
result = [i + j for i in word1 for j in word2]
print(result)
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe
['Hw', 'Ho', 'Hr', 'Hl', 'Hd', 'ew', 'eo', 'er', 'lw', 'lo', 'lr', 'll', 'ld', 'ew', 'eo', 'er', 'lw', 'lo', 'lr', 'll', 'ld']
```

```
"""
result = []
for i in word1:
    for j in word2:
        result.append(i + j)
"""
```

이중 for문 컴프리헨션

```
case1 = ['A', 'B', 'C']
case2 = ['A', 'B', 'C']
result = [i + j for i in case1 for j in case2 if i != j]
print(result)
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe
['AB', 'AC', 'BA', 'BC', 'CA', 'CB']
```

```
case1 = ['A', 'B', 'C']
case2 = ['D', 'E', 'F']
result = [[i + j for i in case1] for j in case2] # 뒤에 있는 for문이 추가 됨
print(result)
```

```
D:\KHY\pycharm_workspace\exam01\venv\Scripts\python.exe D:/KHY/pyc  
[['AD', 'BD', 'CD'], ['AE', 'BE', 'CE'], ['AF', 'BF', 'CF']]
```

이중리스트

뒤에 작성한 for문 코드가 베이스