데이터프로그래밍 기초 3일차

2026-1 DS Bootcamp

부산대학교
데이터사이언스전문대학원
석사과정 박민서

PUSAN NATIONAL UNIVERSITY
Graduate School of Data Science

CONTENTS

List, Dictionary, Tuple, Set

▪ 리스트 (List)

- 여러 가지 자료를 저장할 수 있는 자료형
- [요소(element), 요소, …]

273	32	103	“문자”	True	False
[0]	[1]	[2]	[3]	[4]	[5]
[-6]	[-5]	[-4]	[-3]	[-2]	[-1]

```
my_list = [273, 32, 103, "문자열", True, False]
print(my_list)
print(type(my_list))
```

✓ 0.0s

```
[273, 32, 103, '문자열', True, False]
<class 'list'>
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ 리스트 (List)

- 여러 가지 자료를 저장할 수 있는 자료형
- [요소(element), 요소, …]

273	32	103	“문자”	True	False
[0]	[1]	[2]	[3]	[4]	[5]
[-6]	[-5]	[-4]	[-3]	[-2]	[-1]

```
print(my_list[0])
print(my_list[1])
print(my_list[2:4])

print(my_list[-1])
print(my_list[-2])
```

✓ 0.0s

```
273
32
[103, '문자열']
False
True
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ 리스트 (List)

- 여러 가지 자료를 저장할 수 있는 자료형
- [요소(element), 요소, …] ☞ 리스트가 한 리스트의 요소가 될 수 있음

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

print(my_list[1])
print(my_list[1][0])
```

✓ 0.0s

```
[4, 5, 6]
4
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ list형 연산자, 함수

- 리스트 연결(+), 반복(*), 길이 len( )

```
A = [1, 2, 3]
B = [4, 5, 6]

print(A + B)
print(A * 3)
print(len(A))
```

✓ 0.0s

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

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ list형 연산자, 함수

- append( ): list의 맨 뒤에 요소 추가
- insert( ): list의 중간 위치에 요소 추가 ☞ (리스트).insert(삽입할 위치, 요소)

```
A = [1, 2, 3]

A.append(4)      # [1, 2, 3, 4]
A.append(5)      # [1, 2, 3, 4, 5]
print(A)

A.insert(3, 100)   # [1, 2, 3, 100, 4, 5]
print(A)
```

✓ 0.0s

```
[1, 2, 3, 4, 5]
[1, 2, 3, 100, 4, 5]
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ list형 연산자, 함수

- append( ): list의 맨 뒤에 요소 추가
- insert( ): list의 중간 위치에 요소 추가 ☞ (리스트).insert(삽입할 위치, 요소)

```
A = [1, 2, 3]

A.append(4)      # [1, 2, 3, 4]
A.append(5)      # [1, 2, 3, 4, 5]
print(A)

A.insert(3, 100)   # [1, 2, 3, 100, 4, 5]
print(A)
```

✓ 0.0s

```
[1, 2, 3, 4, 5]
[1, 2, 3, 100, 4, 5]
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ list형 연산자, 함수

- extend( ): 리스트 뒤에 리스트를 연결(확장) ☞ 원본 값 변경

```
A = [1, 2, 3]
B = [4, 5, 6]

A.extend(B) # [1, 2, 3, 4, 5, 6]
print(A)
A.extend(7) # 에러
print(A)
```

⊗ 0.0s

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

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[122], line 6
      4 A.extend(B) # [1, 2, 3, 4, 5, 6]
      5 print(A)
----> 6 A.extend(7) # 에러
      7 print(A)

TypeError: 'int' object is not iterable
```

▪ list형 연산자, 함수

- del: 특정 요소(변수) 제거
- pop( ): 리스트의 요소 중 해당 인덱스 제거 (기본 = -1, 맨 뒤)
- remove( ): 리스트 내부의 값을 지정해서 제거
  (여러 요소가 존재한다면 맨 앞에서 부터 제거)
- clear( ): 리스트 내부 요소 전부 제거

```
my_list = [1, 2, 3, 4, 5, 1, 2, 1, 2]
del my_list[4]      # [1, 2, 3, 4, 1, 2, 1, 2]
print(my_list)

my_list.pop()       # [1, 2, 3, 4, 1, 2, 1]
print(my_list)

my_list.pop(1)      # [1, 3, 4, 1, 2, 1]
print(my_list)

my_list.remove(1)   # [3, 4, 1, 2, 1]
print(my_list)

my_list.remove(-1)  # 오류
print(my_list)
```

0.0s

```
[1, 2, 3, 4, 1, 2, 1, 2]
[1, 2, 3, 4, 1, 2, 1]
[1, 3, 4, 1, 2, 1]
[3, 4, 1, 2, 1]

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[123], line 14
     11 my_list.remove(1)   # [3, 4, 1, 2, 1]
     12 print(my_list)
---> 14 my_list.remove(-1)  # 오류
     15 print(my_list)

ValueError: list.remove(x): x not in list
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ list형 연산자, 함수

- sort( ): 리스트 요소 정렬 (기본은 오름차순)
- sorted( ): 정렬된 리스트 반환 (기본은 오름차순)
- reverse( ): 리스트를 역순으로 정렬

```
my_list = [1, 5, 2, 4, 3]

my_list.sort() # 오름차순 정렬
print(my_list)  # [1, 2, 3, 4, 5]

my_list.sort(reverse=True) # 내림차순 정렬
print(my_list)

new_list = sorted(my_list) # sorted(my_list, reverse=True)도 가능
print(new_list)

new_list.reverse()  # [5, 4, 3, 2, 1]
print(new_list)
```

✓ 0.0s

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

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ list형 연산자, 함수

- sort( ): 리스트 요소 정렬 (기본은 오름차순), sorted( ): 정렬된 리스트 반환 (기본은 오름차순)
- reverse( ): 리스트를 역순으로 정렬

```
my_list = [1, 5, 2, 4, 3]

my_list.sort() # 오름차순 정렬
print(my_list)  # [1, 2, 3, 4, 5]

my_list.sort(reverse=True) # 내림차순 정렬
print(my_list)

new_list = sorted(my_list) # sorted(my_list, reverse=True)도 가능
print(new_list)

new_list.reverse()  # [5, 4, 3, 2, 1]
print(new_list)
```

✓ 0.0s

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

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ list형 연산자, 함수

- in, not in: 리스트 내부에 특정 값이 있는지 (없는 지) 확인

```
my_list = [10, 20, 30, 40, 50]

print(10 in my_list)    # True
print(1 in my_list)     # False
print(5 not in my_list) # True
```

✓ 0.0s

```
True
False
True
```

▪ list형 연산자, 함수

- min( ), max( ), sum( ): 리스트 내부의 최솟값, 최댓값, 합 출력 연산자

```
my_list = [10, 20, 30, 40, 50]
print(min(my_list))
print(max(my_list))
print(sum(my_list))
```

✓ 0.0s

```
10
50
150
```

▪ list형 연산자, 함수

- enumerate( ): 리스트 내의 요소의 인덱스와 함께 반환

```python
my_list = [10, 20, 30, 40, 50]

# enumerate 없이 index와 value 함께 출력
for idx in range(len(my_list)):
    print(f"idx = {idx}, value = {my_list[idx]}")

print("-" * 20)

# enumerate 이용해서 index와 value 함께 출력
for idx, value in enumerate(my_list):
    print(f"idx = {idx}, value = {value}")
```

✓ 0.0s

```
idx = 0, value = 10
idx = 1, value = 20
idx = 2, value = 30
idx = 3, value = 40
idx = 4, value = 50
--------------------
idx = 0, value = 10
idx = 1, value = 20
idx = 2, value = 30
idx = 3, value = 40
idx = 4, value = 50
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ list형 연산자, 함수

- '*': 전개 연산자
- *(list) = list[0], list[1], …

```
my_list = [10, 20, 30, 40, 50]

print(my_list)
print(*my_list)
```

✓ 0.0s

```
[10, 20, 30, 40, 50]
10 20 30 40 50
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

Loop Statement (Review)

▪ 반복문 (for 구문)

- **for 반복할 변수 in 반복할 수 있는 자료:**
  반복할 코드

※ '반복할 수 있는 자료' = list(리스트), dict(딕셔너리), string(문자열), range(범위)

```
my_list = [10, 20, 30, 40, 50]

for num in my_list:
    print(num)
```

✓ 0.0s

```
10
20
30
40
50
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ 2차원 list

- 중첩된 list 형태 ☞ list 안의 list

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

# len(my_list) = 3
for i in range(len(my_list)):
    for j in range(len(my_list[i])):
        print(f"list[{i}][{j}] = {my_list[i][j]}")
```

✓ 0.0s

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

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ 딕셔너리 (Dictionary)

- 키(Key)를 기반으로 값(Value)를 저장하는 것
- {키1: 값1, 키2: 값2, …}

```
my_dict = {
    "키": 175,
    "나이": 28,
    "학과": "데이터사이언스학과",
    1: 50,  # 숫자형도 key로 사용 가능
    False: 100  # bool형도 key로 사용 가능
}

print(my_dict)
print(type(my_dict))
```

✓ 0.0s

```
{'키': 175, '나이': 28, '학과': '데이터사이언스학과', 1: 50, False: 100}
<class 'dict'>
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ 딕셔너리 (Dictionary)

- 키(Key)를 기반으로 값(Value)를 저장하는 것
- {키1: 값1, 키2: 값2, …}

```
my_dict = {
    "키": 175,
    "나이": 28,
    "학과": "데이터사이언스학과",
    1: 50,  # 숫자형도 key로 사용 가능
    False: 100  # bool형도 key로 사용 가능
}

print(my_dict["키"])    # 175
print(my_dict[False])   # 100
print(my_dict[2])       # 오류 발생 (없는 키)
```

0.0s

```
175
100

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[10], line 11
      9 print(my_dict["키"])    # 175
     10 print(my_dict[False])   # 100
---> 11 print(my_dict[2])       # 오류 발생 (없는 키)

KeyError: 2
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ 딕셔너리 (Dictionary)

- 키(Key)를 기반으로 값(Value)를 저장하는 것
- {키1: 값1, 키2: 값2, …}

```
my_dict = {
    "키": 175,
    "나이": 28
}

my_dict["나이"] = 26    # "나이"에 대한 값 26으로 변경
my_dict["만나이"] = 24  # "만나이"라는 키 추가
print(my_dict)
```

✓ 0.0s

```
{'키': 175, '나이': 26, '만나이': 24}
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ dict형 연산자, 함수

- 딕셔너리 값 추가, 변경, 삭제

```
my_dict = {
    "키": 175,
    "나이": 28
}

my_dict["나이"] = 26    # "나이"에 대한 값 26으로 변경
print(my_dict)

my_dict["만나이"] = 24  # "만나이"라는 키 추가
print(my_dict)

del my_dict["키"]       # "키" key 삭제 (값과 같이)
print(my_dict)
```

✓ 0.0s

```
{'키': 175, '나이': 26}
{'키': 175, '나이': 26, '만나이': 24}
{'나이': 26, '만나이': 24}
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ dict형 연산자, 함수

- get( ): 딕셔너리 내의 key에 대한 접근 연산자
- 딕셔너리.get("접근하려는 키", 기본값 [선택])

```
my_dict = {
    "키": 175,
    "나이": 28,
    "학과": "데이터사이언스학과",
    1: 50,  # 숫자형도 key로 사용 가능
    False: 100  # bool형도 key로 사용 가능
}

value1 = my_dict.get("키")
print(value1)       # 175

value2 = my_dict.get("만나이")
print(value2)       # None

value3 = my_dict.get("만나이", "-1")
print(value3)
```

✓ 0.0s

```
175
None
-1
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ dict형 연산자, 함수

- keys( ): 딕셔너리 내부의 key들만 반환, values( ): 딕셔너리 내부의 values들만 반환
- items( ): 딕셔너리 내부의 key, value 같이 반환

```
my_dict = {
    "키": 175,
    "나이": 28,
    "학과": "데이터사이언스학과",
    1: 50,  # 숫자형도 key로 사용 가능
    False: 100  # bool형도 key로 사용 가능
}

keys = my_dict.keys()
print(keys)     # dict_keys(["키", "나이", "학과", 1, False])

values = my_dict.values()
print(values)   # dict_values([175, 28, "데이터사이언스학과", 50, 100])

items = my_dict.items()
print(items)
# dict_items([("키", 175), ("나이", 28), ...])
```

✓ 0.0s

```
dict_keys(['키', '나이', '학과', 1, False])
dict_values([175, 28, '데이터사이언스학과', 50, 100])
dict_items([('키', 175), ('나이', 28), ('학과', '데이터사이언스학과'), (1, 50), (False, 100)])
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

List, Dictionary, Tuple, Set

▪ 튜플 (Tuple)

- list와 비슷한 자료형 + 요소 변경 X
- (데이터, 데이터, 데이터, …)

```
my_list = [10, 20, 30] # list 선언
my_tuple = (10, 20, 30) # tuple 선언

print(type(my_list))
print(type(my_tuple))
```

✓ 0.0s

```
<class 'list'>
<class 'tuple'>
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

List, Dictionary, Tuple, Set

▪ 튜플 (Tuple)

- list와 비슷한 자료형 + 요소 변경 X
- (데이터, 데이터, 데이터, …)

```
my_list = [10, 20, 30]
my_tuple = (10, 20, 30)

my_list[1] = 40     # list 요소 값 변경
print(my_list)

my_tuple[1] = 40    # tuple 요소 값 변경 X]
print(my_tuple)
```

0.4s

```
[10, 40, 30]

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[18], line 7
      4 my_list[1] = 40     # list 요소 값 변경
      5 print(my_list)
----> 7 my_tuple[1] = 40    # tuple 요소 값 변경 X]
      8 print(my_tuple)

TypeError: 'tuple' object does not support item assignment
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

▪ 튜플 (Tuple)

- list와 비슷한 자료형 + 요소 변경 X
- (데이터, 데이터, 데이터, …)

```
# tuple 다양한 선언
t1 = 1,          # tuple
print(type(t1))

t2 = (1,)        # tuple
print(type(t2))

t3 = (1)         # int
print(type(t3))

t4 = 10, 20, 30 # tuple
print(t4)
print(type(t4))
```

✓ 0.0s

```
<class 'tuple'>
<class 'tuple'>
<class 'int'>
(10, 20, 30)
<class 'tuple'>
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

List, Dictionary, Tuple, Set

▪ 튜플 (Tuple)

- list와 비슷한 자료형 + 요소 변경 X
- (데이터, 데이터, 데이터, …)

```
# tuple, list을 통한 여러 변수 할당
[a, b] = [10, 20]   # a = 10, b = 20
(c, d) = (30, 40)   # c = 30, d = 40

print(a, b, c, d)
```

✓ 0.0s

```
10 20 30 40
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

▪ 튜플 (Tuple)

- list와 비슷한 자료형 + 요소 변경 X
- (데이터, 데이터, 데이터, …)

```
# 값 교환 코드
a, b = 10, 20
print(a, b)

# 기존 코드 (ex. C언어)
temp = a          # temp = 10
a = b             # a = b = 20
b = temp          # b = temp = 10
print(a, b)

# Python 만의 코드
a, b = 10, 20
a, b = b, a       # a = 20, b = 10
print(a, b)
```

✓ 0.0s

```
10 20
20 10
20 10
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

List, Dictionary, Tuple, Set

▪ 집합 (Set)

- 수학에서의 집합과 동일
- {요소, 요소, … }

```
my_list = [1, 2, 3, 4, 1, 2, 3, 4]
print(my_list)
print(type(my_list))

my_set = {1, 2, 3, 4, 1, 2, 3, 4}   # 중복 제거
print(my_set)
print(type(my_set))
```

✓ 0.0s

```
[1, 2, 3, 4, 1, 2, 3, 4]
<class 'list'>
{1, 2, 3, 4}
<class 'set'>
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

List, Dictionary, Tuple, Set

▪ set형 연산자, 함수

- add( ): 집합에 요소 추가, remove( ): 집합에 요소 제거, update( ): 집합에 여러 요소 수정

```
set_a = {1, 2, 3}
set_a.add(4)    # {1, 2, 3, 4}
print(set_a)

set_a.add(3)    # {1, 2, 3, 4}
print(set_a)

set_a.remove(4) # {1, 2, 3}
print(set_a)

set_a.update([3, 4, 5]) # {1, 2, 3, 4, 5}
print(set_a)
```

✓ 0.0s

```
{1, 2, 3, 4}
{1, 2, 3, 4}
{1, 2, 3}
{1, 2, 3, 4, 5}
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

▪ set형 연산자, 함수

- 합집합(|), 교집합(&), 차집합(-), 대칭차집합(^)

```
set_a = {1, 2, 3}
set_b = {2, 3, 4}

print(set_a | set_b) # {1, 2, 3, 4}
print(set_a & set_b) # {2, 3}
print(set_a - set_b) # {1}
print(set_b - set_a) # {4}
print(set_a ^ set_b) # {1, 4}
```

✓ 0.0s

```
{1, 2, 3, 4}
{2, 3}
{1}
{4}
{1, 4}
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

List, Dictionary, Tuple, Set

▪ set형 연산자, 함수

- union( ), intersection( ), difference( ), symmetric_difference( )와 동치

```
set_a = {1, 2, 3}
set_b = {2, 3, 4}

print(set_a.union(set_b)) # {1, 2, 3, 4}
print(set_a.intersection(set_b)) # {2, 3}
print(set_a.difference(set_b)) # {1}
print(set_b.difference(set_a)) # {4}
print(set_a.symmetric_difference(set_b)) # {1, 4}
```

✓ 0.0s

```
{1, 2, 3, 4}
{2, 3}
{1}
{4}
{1, 4}
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 316-321쪽

Q&A

Comprehension

▪ 리스트 내포 (List Comprehension)

- list 생성 시 반복문, 조건문을 활용하여 간결하게 코드 작성 ☞ 가독성 ↑

```
# list comprehension 없이 [1, 4, 9, 16, 25] 생성
list1 = []
for i in range(1, 6):
    list1.append(i ** 2)
print(f"list1: {list1}")

# list comprehension으로 [1, 4, 9, 16, 25] 생성
list2 = [i ** 2 for i in range(1, 6)]
print(f"list2: {list2}")
```

✓ 0.0s

```
list1: [1, 4, 9, 16, 25]
list2: [1, 4, 9, 16, 25]
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

Comprehension

▪ 리스트 내포 (List Comprehension)

- list 생성 시 반복문, 조건문을 활용하여 간결하게 코드 작성 ☞ 가독성 ↑

```
og_list = ["사과", "바나나", "초콜릿", "체리", "자두"]

# list comprehension 없이 "초콜릿" 제외한 list 생성
new_list1 = []
for fruit in og_list:
    if fruit != "초콜릿":
        new_list1.append(fruit)
print(f"new_list1 = {new_list1}")

# list comprehension 사용
new_list2 = [fruit for fruit in og_list if fruit != "초콜릿"]
print(f"new_list2 = {new_list2}")
```

✓ 0.0s

```
new_list1 = ['사과', '바나나', '체리', '자두']
new_list2 = ['사과', '바나나', '체리', '자두']
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

Comprehension

▪ 딕셔너리 내포 (Dictionary Comprehension)

- dict 생성 시 반복문, 조건문을 활용하여 간결하게 코드 작성 ☞ 가독성 ↑

```python
fruit_color = {"apple" : "red", "banana" : "yellow", "peach":"pink"}
my_dict = {}

# dictionary comprehension 없이 banana 제외한 dict 저장
for key in fruit_color:
    if key != "banana":
        my_dict[key] = fruit_color[key]

print(my_dict)

# dictionary comprehension 활용 banana 제외한 dict 저장
my_dict2 = {k:v for k,v in fruit_color.items() if k != "banana"}
print(my_dict2)
```

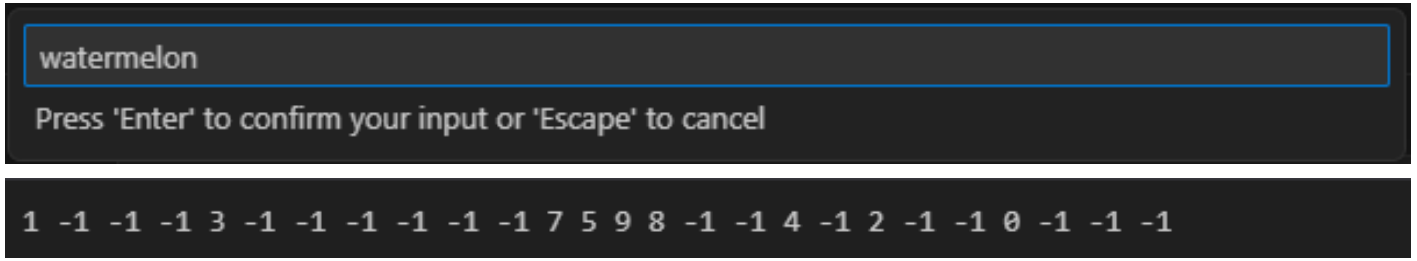
✓ 0.0s

```
{'apple': 'red', 'peach': 'pink'}
{'apple': 'red', 'peach': 'pink'}
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽

▪ 실습 1

- 아래와 같이 작동하는 프로그램의 코드를 작성하세요.
  - 입력: 알파벳 소문자로만 이루어진 단어 (길이 100 이하)
  - 출력: 각각의 알파벳에 대해서 처음 등장하는 위치를 공백으로 구분해서 출력
  - 입력 예시: watermelon
  - 출력 예시: 1 -1 -1 -1 3 -1 -1 -1 -1 -1 -1 7 5 9 8 -1 -1 4 -1 2 -1 -1 0 -1 -1 -1



윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 84-271쪽
https://www.acmicpc.net/problem/10809

▪ 함수 (Function)

- 어떤 일을 수행하는 명령들의 묶음 + 이들을 호출하여 사용할 수 있게 하는 프로그래밍 방법

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수 (Function)

- 어떤 일을 수행하는 명령들의 묶음 + 이들을 호출하여 사용할 수 있게 하는 프로그래밍 방법
- “def”를 이용해 함수를 정의

```
def my_func():
    # 수행할 명령 코드 작성
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수 (Function)

- 어떤 일을 수행하는 명령들의 묶음 + 이들을 호출하여 사용할 수 있게 하는 프로그래밍 방법
- “def”를 이용해 함수를 정의

```
def hello():    # 인사말 출력하는 함수 정의
    print("안녕하세요")
    print("Hello")
    print("안녕히가세요")
    print("Bye")

hello()     # 함수 호출
```

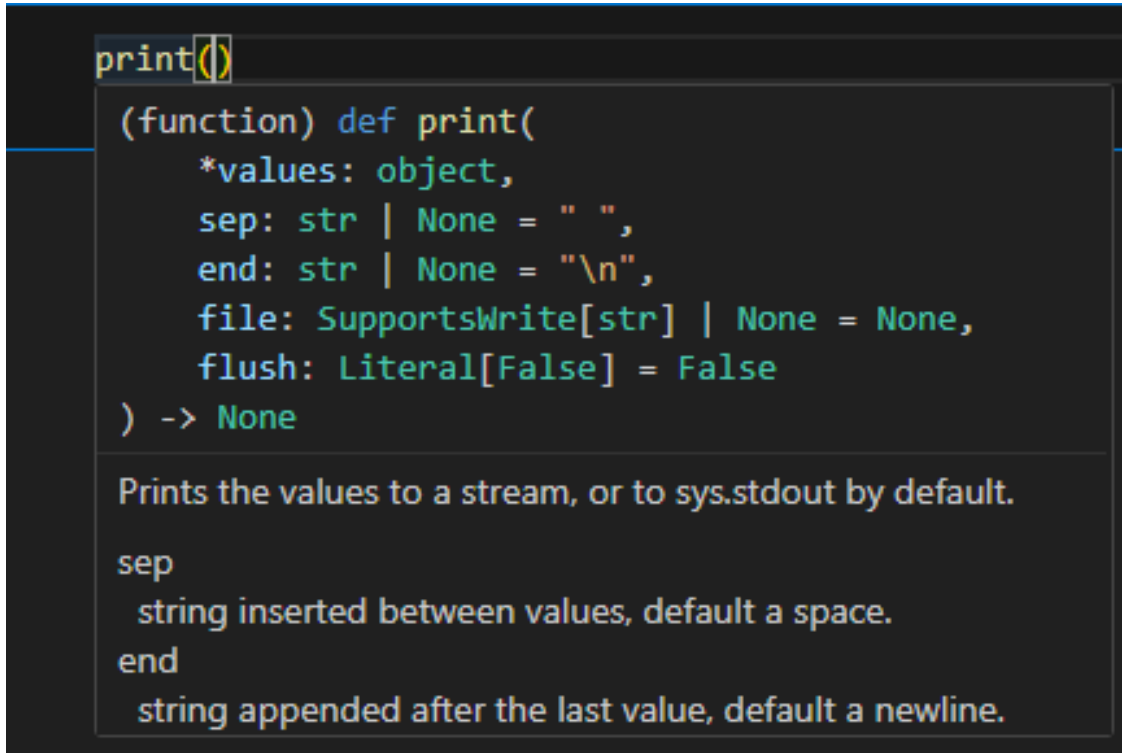
✓ 0.0s

```
안녕하세요
Hello
안녕히가세요
Bye
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 매개변수 (Parameter)와 인수(Arguments)

- 매개변수: 함수의 입력값으로 전달된 값
- 인수: 함수 호출 시 전달하는 입력값



윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 매개변수 (Parameter)와 인수(Arguments)

- 매개변수: 함수의 입력값으로 전달된 값
- 인수: 함수 호출 시 전달하는 입력값

```
def print_word(word, n):
    # word와 n이란 매개변수 설정
    # n번 동안 word 출력하는 함수
    for i in range(n):
        print(word)

print_word("파이썬", 4)
```

✓ 0.0s

```
파이썬
파이썬
파이썬
파이썬
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 매개변수 (Parameter)와 인수(Arguments)

- 매개변수: 함수의 입력값으로 전달된 값
- 인수: 함수 호출 시 전달하는 입력값

```
def print_word(word, n):
    # word와 n이란 매개변수 설정
    # n번 동안 word 출력하는 함수
    for i in range(n):
        print(word)

print_word("파이썬")    # 매개변수 n에 대한 값이 주어지지 않아 오류 발생
```

⊗ 0.0s

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[35], line 7
      4     for i in range(n):
      5         print(word)
----> 7 print_word("파이썬")

TypeError: print_word() missing 1 required positional argument: 'n'
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 매개변수 (Parameter)와 인수(Arguments)

- 매개변수: 함수의 입력값으로 전달된 값
- 인수: 함수 호출 시 전달하는 입력값

```
def print_word(word, n):
    # word와 n이란 매개변수 설정
    # n번 동안 word 출력하는 함수
    for i in range(n):
        print(word)

print_word("파이썬", 3, 6)    # word, n에 대한 매개변수 외의 매개변수를 입력 시도하여 오류
```

⊗ 0.0s

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[36], line 7
      4     for i in range(n):
      5         print(word)
----> 7 print_word("파이썬", 3, 6)    # word, n에 대한 매개변수 외의 매개변수를 입력 시도하여 오류

TypeError: print_word() takes 2 positional arguments but 3 were given
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 매개변수 (Parameter)와 인수(Arguments)

- 매개변수: 함수의 입력값으로 전달된 값
- 인수: 함수 호출 시 전달하는 입력값

```
def print_word(word, n=3):
    # word와 n이란 매개변수 설정 (n은 기본값 3)
    # n번 동안 word 출력하는 함수
    for i in range(n):
        print(word)

print_word("파이썬") # n이 주어지지 않아 기본값 3으로 설정
```

✓ 0.0s

```
파이썬
파이썬
파이썬
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 매개변수 (Parameter)와 인수(Arguments)

- 가변 매개변수(인수) (Variable Positional Arguments): 함수의 인수 개수의 제약이 없을 경우
- 가변 매개변수는 한 번만 사용 가능 + 가변 매개변수 뒤에 일반 parameter가 올 수 없음

```python
def print_n_times(n, *values):
    # values로 전달된 값들을 활용한 n번 반복문
    for i in range(n):
        for value in values:
            # values를 리스트 처럼 활용
            print(value)
        print()

print_n_times(2, "DS", "BootCamp", "Python", "Hello?")
```

✓ 0.0s

```
DS
BootCamp
Python
Hello?

DS
BootCamp
Python
Hello?
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 매개변수 (Parameter)와 인수(Arguments)

- 키워드 매개변수 (Keyword Parameter): “변수 = 기본값” 형태로 정의하는 매개변수
- 일반 매개변수보다 뒤에 정의해야 함
- 참고 사이트: https://godoftyping.wordpress.com/2017/05/24/python-%ED%95%A8%EC%88%98-%EC%9D%B8%EC%9E%90%EA%B0%92-%EC%A0%95%EB%A6%AC/

```
def sum_three(a, b=10, c=100):
    print(a + b + c)

sum_three(10, 20, 30)   # 기본 호출
sum_three(a=10, b=100, c=200)   # 키워드 매개변수 활용
sum_three(c=10, b=100, a=200)   # 키워드 매개변수는 순서 상관 X
sum_three(10, c=300)        # 키워드 매개변수는 일부만 지정해도 상관 X
```

✓ 0.0s

```
60
310
310
320
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 반환값(Return)

- 함수 호출 시, 호출한 해당 부분으로 값이 반환되는 것

```
def calc(x):
    return 2 * x + 1

res = calc(3)   # calc(3)의 반환값인 7이 res에 저장
print(res)
```

✓ 0.0s

```
7
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 반환값(Return)

- 함수 호출 시, 호출한 해당 부분으로 값이 반환되는 것

```
def calc(x):
    return      # 반환값 없이 return 사용 -> None 반환

res = calc(3)   # calc(3)의 반환값인 None이 res에 저장
print(res)
```

✓ 0.0s

```
None
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 함수의 반환값(Return)

- 함수 호출 시, 호출한 해당 부분으로 값이 반환되는 것

```
def my_func():
    print("pos 1")
    print("pos 2")
    return      # 이 시점에서 return 실행
    print("pos 3")  # 뒤의 코드 무시

my_func()
```

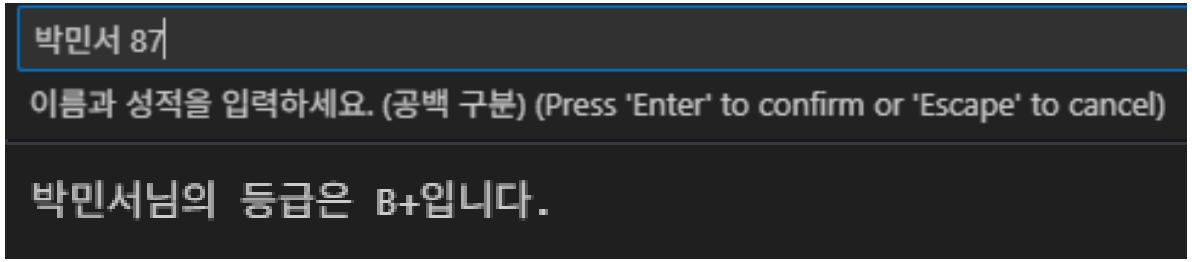
✓ 0.0s

```
pos 1
pos 2
```

윤인성, 「혼자 공부하는 파이썬 개정판」, 한빛미디어, 2022, 274-357쪽
https://www.nise.go.kr/sedu/pe/pe3/page3_07.html

▪ 실습 2

- 아래와 같이 작동하는 프로그램의 코드를 작성하세요.
  - 입력: 이름과 성적을 공백으로 구분하여 입력 (성적은 정수)
  - 출력: 성적의 등급을 반환하여 이름과 함께 결과 출력
  - 입력 예시: “박민서 87”
  - 출력 예시: “박민서님의 등급은 B+입니다.”
  - 조건:
    - 성적의 등급을 반환하는 함수를 사용
    - 출력 시 f-string 사용
    - 95점 이상: A+, 90~94점: A, 85~89점: B+, 80~84점: B
      75~79점: C+, 70~74점: C, 65~69점: D+, 60~64점: D, 59점 이하: F



▪ 3일차 과제

- GitHub 사이트에서 “3일차_과제.ipynb” 다운로드
- 코드 작성 후, “**본인이름**_3일차_과제.ipynb”로 저장
- 저장한 과제 파일 전송 (이메일 주소: minsuh99@pusan.ac.kr)

  기한: ~ 2/4 PM 23:59:59

Q&A