

파이썬 기초

## 09 딕셔너리, 집합, 컬렉션



홍필두 교수  
(파이썬기초)



## 학습내용

- 01 딕셔너리
- 02 집합
- 03 컬렉션





## 학습목표

- 컬렉션 형중 하나인 딕셔너리에 대하여 이해하고 사용할 수 있다.
- 컬렉션 형중 하나인 집합에 대하여 이해하고 사용할 수 있다.
- 컬렉션 형의 특징적인 사항에 대하여 이해하고 사용할 수 있다.



# 사전 학습

## “딕셔너리와 집합”

우리가 사전을 찾아 볼 때 ‘a, b, c, d’ 순이나 ‘가, 나, 다, 라’ 순으로 정렬된 순서로 사전의 내용을 찾게 됩니다. 사전의 내용배열은 검색을 쉽게 해주고 있습니다.

여러분들이 수학과목에서 경험한 집합의 정의와 연산처리 등을 다시 떠올려 보시기 바랍니다.

이러한 자료구조는 컴퓨터알고리즘이나 수치분석, 데이터분석에서 유용하게 사용되고 있습니다.

당연히 파이썬 프로그래밍 언어에서도 해당 내용이 존재합니다.

# 사전 학습

## “딕셔너리와 집합”

이번 강의에서는 **사전 (딕셔너리)**, **집합**, **컬렉션**의 특징적인 몇가지 사항을 배웁니다.

여러분은 먼저 해당 내용에 대하여 검색을 통하여 미리 알아보고 조사해 본 후, 본 강의를 듣도록 합시다.



# 01

## 딕셔너리

---

- 1) 딕셔너리의 정의
- 2) 딕셔너리 예제

## 1) 딕셔너리의 정의

### 딕셔너리

키(key)와 값(value)을 한 쌍으로 연관지어 사용하는 자료형

## 1) 딕셔너리의 정의



dic\_name[key] 방식으로 사용함

```
>>> a={"이름":"홍길동", "국어":100, "영어":88, "수학": 95}
>>> a["수학"]
95
>>> a["이름"]
'홍길동'
```



## 1) 딕셔너리의 정의



keys(), values(), items()는 다음과 같음

```
>>> a.keys()
dict_keys(['이름', '국어', '영어', '수학'])
>>> a.values()
dict_values(['홍길동', 100, 88, 95])
>>> a.items()
dict_items([('이름', '홍길동'), ('국어', 100),
            ('영어', 88), ('수학', 95)])
```

## 1) 딕셔너리의 정의



### keys(), values(), items()의 활용 방법

```
>>> a={"이름":"홍길동", "국어":100, "영어":88, "수학": 95}
>>> for i in a.items():
    print (i,end=",")

('이름', '홍길동'),('국어', 100),('영어', 88),('수학', 95),
>>> for i in a.keys():
    print (i,end=",")

이름,국어,영어,수학,
>>> for i in a.values():
    print (i,end=",")

홍길동,100,88,95,
>>>
```

## 1) 딕셔너리의 정의



get()은 해당 키의 값을 검색하며 없는 경우의 출력내용을 지정 가능

```
>>> a={"이름":"홍길동", "국어":100, "영어":88, "수학": 95}
>>> print(a.get("과학"))
None
>>> print(a.get("과학","없는과목입니다"))
없는과목입니다
>>> print(a.get("수학","없는과목입니다"))
95
```

## 2) 딕셔너리 예제



다음 문장에서 각각이 쓰인 글자에 대한 계수를 하는 예제

```
text="""안녕하세요. 오늘은 날씨가 매우 화창합니다.
그래서 기분이 매우 좋네요
너도 기분이 좋을까요 랄라랄랄라라라라라 abc임마"""

a=dict() #하나의 딕셔너리 자료형을 정의
for c in text: #위에 정의한 텍스트를 하나씩 추출하여 처리
    if c not in a: #해당 추출글자가 지금까지의 딕셔너리에 없다면
        a[c]=1 #그때는 최초 1값으로 시작
    else: #이 경우는 해당 추출글자의 딕셔너리가 있다
        a[c] +=1 #그럼 해당 딕셔너리 키에 해당되는 값을 하나더함
print(a)
```

```
>>>
== RESTART: ***.py ==
{'안': 1, '녕': 1, '하': 1, '서': 1, '요': 3, '.': 2, ' ': 11, '오': 1,
'날': 1, '은': 2, '날': 1, '씨': 1, '가': 2, '매': 2, '우': 2, '화': 1, '창':
1, '합': 1, '니': 1, '다': 1, '\n': 2, '그': 1, '래': 1, '서': 1, '기': 2,
'분': 2, '이': 2, '좋': 2, '네': 1, '너': 1, '도': 1, '랄': 3, '라': 7, 'a':
1, 'b': 1, 'c': 1, '임': 1, '마': 1}
```

## 2) 딕셔너리 예제



앞에 출력된 내용을 가나다 순으로 다시 처리

```
#아래서 부터는 보기 힘드니 각 글자(키)를 소트해 본다
b=dict()          #딕셔너리 하나 더 정의
l=list(a.keys())   #앞에 딕셔너리에 키만 뽑아 리스트를 만듦
l.sort()           #해당 리스트 소트
for c in l:        #각 리스트에 대하여 (결국 키값) 하나씩 뽑음
    b[c]=a[c]      # 새로운 딕셔너리는 키값 순서대로 다시 만듦
print(b)
```

```
>>>
== RESTART: ***.py ==
{'\n': 2, ' ': 11, '.': 2, 'a': 1, 'b': 1, 'c': 1, '가': 2, '그': 1, '기':
2, '날': 1, '너': 1, '네': 1, '녕': 1, '늘': 1, '니': 1, '다': 1, '도': 1,
'라': 7, '랄': 3, '래': 1, '마': 1, '매': 2, '분': 2, '서': 1, '셔': 1, '씨':
1, '안': 1, '오': 1, '요': 3, '우': 2, '은': 2, '이': 2, '임': 1, '줄': 2,
'창': 1, '하': 1, '합': 1, '화': 1}
```



# 02

## 집합

- 
- 1) 집합의 정의
  - 2) 집합의 연산

## 1) 집합의 정의

### 집합

- 수학적 의미의 집합과 동일함
- 대괄호 {}로 원소를 표현하며 순서에 상관없이 표시 및 사용됨(순서의 의미가 없음)
- 원소가 중복되어도 하나로 인식됨
- 다른 자료형으로부터 `set()` 함수를 통하여 집합으로 변환함  
(단 딕셔너리 타입은 키만 집합으로 변환됨)

## 1) 집합의 정의

```
>>> a={1,2,2,2,2,3,3,3,4,5}
>>> a
{1, 2, 3, 4, 5}
>>> set(("a",1,2))
{1, 2, 'a'}
>>> set("a")
{'a'}
>>> set({"이름":"홍길동", "국어":100,
"영어":88, "수학": 95})
{'국어', '영어', '이름', '수학'}
>>>
```



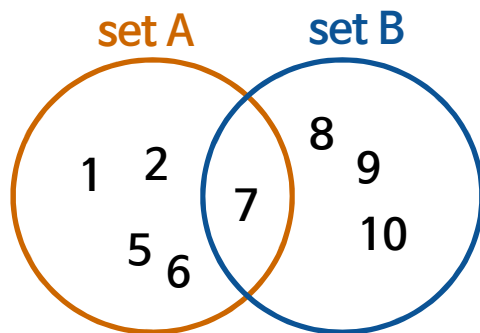
## 1) 집합의 정의

구분	설명
add(값)	원소 추가 시 사용함
remove(값)	원소 제거 시 사용함

```
>>> a={1,2,3,4,5}
>>> a
{1, 2, 3, 4, 5}
>>> a.add(0)
>>> a
{0, 1, 2, 3, 4, 5}
>>> a.add(2.4)
>>> a
{0, 1, 2, 3, 4, 5, 2.4}
>>> a.remove(3)
>>> a
{0, 1, 2, 4, 5, 2.4}
>>>
```

## 2) 집합의 연산

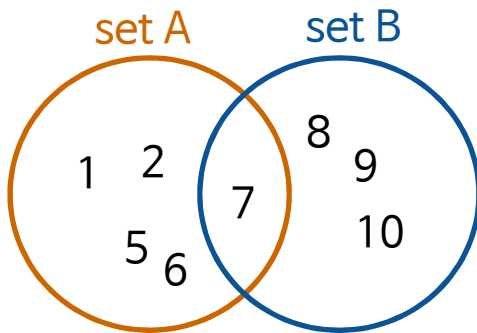
집합연산의 합집합( $\cup$ ), 교집합( $\cap$ ), 차집합( $-$ ), 배타적 차집합( $\Delta$ )을 표시



구분	수학기호	파이썬	결과값	비고
집합 A	A	A	{1,2,3,4,5,6,7}	
집합 B	B	B	{7,8,9,10}	
합집합	$A \cup B$	A B	{1,2,3,4,5,6,7,8,9,10}	두 집합의 모든 원소
교집합	$A \cap B$	A&B	{7}	두 집합의 공통 원소
차집합	$A - B$	A-B	{1,2,3,4,5,6}	A의 원소 중 B의 원소인 것을 제외한 원소

## 2) 집합의 연산

집합연산의 합집합( $\cup$ ), 교집합( $\cap$ ), 차집합( $-$ ), 배타적 차집합( $\Delta$ )을 표시



```
>>> A={1,2,3,4,5,6,7}
>>> B={7,8,9,10}
>>> A|B
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
>>> A&B
{7}
>>> A-B
{1, 2, 3, 4, 5, 6}
>>>
```

## 2) 집합의 연산



부등식을 사용하여 비교가 가능함

```
>>> {1,2,3} < {1,2}
False
>>> {1,2,3} > {1,2}
True
>>> {1,2,3} >= {1,2}
True
>>> {1,2,3} == {1,2}
False
>>> {1,2,3} >= {3,1,2}
True
>>> {1,2,3} == {3,1,2}
True
```

# 03

## 컬렉션

- 1) 컬렉션으로 변환
- 2) 함수인자에 컬렉션 값을 차례로 대입
- 3) 컬렉션 주의사항

## 1) 컬렉션으로 변환



리스트 `a=["a","b","c","d","e"]`를 딕셔너리 `b={1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e'}`로 변환한다면 다음과 같이 처리

```
>>> a=["a","b","c","d","e"]
>>> b={}
>>> for i in range (len(a)):
        b[i]=a[i]    #a는 리스트 b는 딕셔너리

>>> b
{0: 'a', 1: 'b', 2: 'c', 3: 'd', 4: 'e'}
```

## 1) 컬렉션으로 변환



`enumerate()` 함수는 요소를 0부터 차례의 숫자로 대입해 줌



보기에서 `b`는 딕셔너리, `c`는 튜플을 원소로 가진 리스트로 변환 되었음

## 1) 컬렉션으로 변환

```
>>> a=["a","b","c","d","e"]
>>> b=enumerate(a) #여기 까지는 중간단계
>>> b
<enumerate object at 0x0000017D47D6CFC0>
>>> b=dict(enumerate(a)); c=list(enumerate(a)) #딕셔너리와 리스트를 만든다
>>> b
{0: 'a', 1: 'b', 2: 'c', 3: 'd', 4: 'e'}
>>> c
[(0, 'a'), (1, 'b'), (2, 'c'), (3, 'd'), (4, 'e')]
```



## 1) 컬렉션으로 변환



zip은 두 컬렉션(리스트, 튜플, 집합 등)을 하나씩 대응관계를 만들어 줌



단 서로의 개수가 다를 때 적은 개수에 맞추어 구성됨

## 1) 컬렉션으로 변환

```
>>> a={1,2,3,4,5,6,7,8} #집합a
>>> b={"a","b","c","d","e"} #집합b
>>> zip(a,b) #zip만으로는 단순히 object임
<zip object at 0x0000029D7D441A08>
>>> c=dict(zip(a,b));d=list(zip(a,b))
>>> c
{1: 'c', 2: 'e', 3: 'd', 4: 'a', 5: 'b'}
>>> d
[(1, 'c'), (2, 'e'), (3, 'd'), (4, 'a'), (5, 'b')]
>>> a=(1,2,3,4,5,6,7,8) #튜플a
>>> b=["a","b","c","d","e"] #리스트 b
>>> c=dict(zip(a,b));d=list(zip(a,b))
>>> c
{1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e'}
>>> d
[(1, 'a'), (2, 'b'), (3, 'c'), (4, 'd'), (5, 'e')]
```

- b는 딕셔너리, c는 튜플을 원소로 가진 리스트로 변환 되었음

## 2) 함수인자에 컬렉션 값을 차례로 대입



filter(함수명,인자\_컬렉션)은 인자 컬렉션의 값을 하나씩 대응하여 함수를 실행함



filter는 인자가 하나인 경우만 가능

```
def func(x):  
    return x/2  
  
x=(1,2,3,4,5,6)  
for i in filter(func,x):  
    print(i)
```



```
>>>  
== RESTART: ***.py ==  
1  
2  
3  
4  
5  
6  
>>>
```

## 2) 함수인자에 컬렉션 값을 차례로 대입



filter는 인자가 하나인 경우만 가능



인자를 여러 개를 전달할 경우 map함수를 사용

```
def func(x,y):  
    return x+y  
  
x=(1,2,3,4,5,6)  
y=(7,8,9,10,11,12)  
#for i in filter(func,x,y): #filter를 사용하면 예러  
for i in map(func,x,y):  
    print(i)
```

```
>>>  
== RESTART: ***.py ==  
8  
10  
12  
14  
16  
18>>>
```

## 2) 함수인자에 컬렉션 값을 차례로 대입



lambda는 전달한 함수가 간단한 경우 한 줄 안에 기입 가능



lambda 인자는 함수내용 으로 한 줄에 기입가능

```
x=(1,2,3,4,5,6)
y=(7,8,9,10,11,12)
for i in map(lambda x,y:x+y, x,y):
    print(i)
```

```
>>>
== RESTART: ***.py ==
8
10
12
14
16
18
```

### 3) 컬렉션 주의사항



{ 컬렉션을 대입하면 컬렉션의 값이 복사되는 것이  
아니라 주소가 복사됨 }

```
a=[1,2,3]
>>> b=a
>>> b[2]=5
>>> a
[1, 2, 5]
>>> b
[1, 2, 5]
>>> b[1]=3
>>> a
[1, 3, 5]
```

### 3) 컬렉션 주의사항



대입된 컬렉션을 변경하면 원 컬렉션도 변경됨



이때는 `b=a [:]` 또는 `b=a.copy()`를 사용(열은 복사)

### 3) 컬렉션 주의사항



```
>>> a=[1,2,3]
>>> b=a[:]
>>> b[2]=5
>>> a;b
[1, 2, 3]
[1, 2, 5]
```

```
>>> a=[1,2,3]
>>> b=a.copy()
>>> b[2]=5
>>> a;b
[1, 2, 3]
[1, 2, 5]
```



### 3) 컬렉션 주의사항

#### deepcopy



컬렉션 안에 컬렉션이 있을 때 copy를 사용시  
첫 번째 컬렉션만 복사됨



이 경우 아래레벨의 모든 컬렉션을 복사할 경우  
deepcopy를 사용

### 3) 컬렉션 주의사항

#### deepcopy

```
>>> a=[[1,2],3,4]
>>> b=a.copy()
>>> b
[[1, 2], 3, 4]
>>> b[0][1]=3
>>> b
[[1, 3], 3, 4]
>>> a
[[1, 3], 3, 4] #컬렉션안에 컬렉션이 있는경우 copy만으로 2레벨이상의
                #리스트까지 복사되지 않는다
>>> a=[[1,2],3,4]
>>> b=a[:]
>>> b
[[1, 2], 3, 4]
>>> b[0][1]=3
>>> a
[[1, 3], 3, 4] #b=a[:]도 마찬가지로
```

### 3) 컬렉션 주의사항

#### deepcopy



아래레벨의 모든 컬렉션을 복사할 경우 `deepcopy`를 사용




`import copy`를 정의하여 가져다 `deepcopy`를 사용

### 3) 컬렉션 주의사항

#### deepcopy

```
import copy as c

a=[[1,2],3,4]
b=c.deepcopy(a)
b[0][1]=3
print(a,b)
```



```
>>>
== RESTART: ***.py ==
[[1, 2], 3, 4] [[1, 3], 3, 4]
>>>
```

# 04

## 실습하기 I

- 1) 딕셔너리 - 정의
- 2) 딕셔너리 - 예제
- 3) 집합 - 정의

# 실습내용

- 1) 딕셔너리 - 정의
- 2) 딕셔너리 - 예제
- 3) 집합 - 정의

# 05

## 실습하기 II

- 1) 집합 - 연산
- 2) 컬렉션 - 컬렉션으로 변환, 함수 인자 전달
- 3) 컬렉션 - copy, deepcopy

# 실습내용

- 1) 집합 - 연산
- 2) 컬렉션 - 컬렉션으로 변환, 함수 인자 전달
- 3) 컬렉션 - copy, deepcopy





## 학습활동

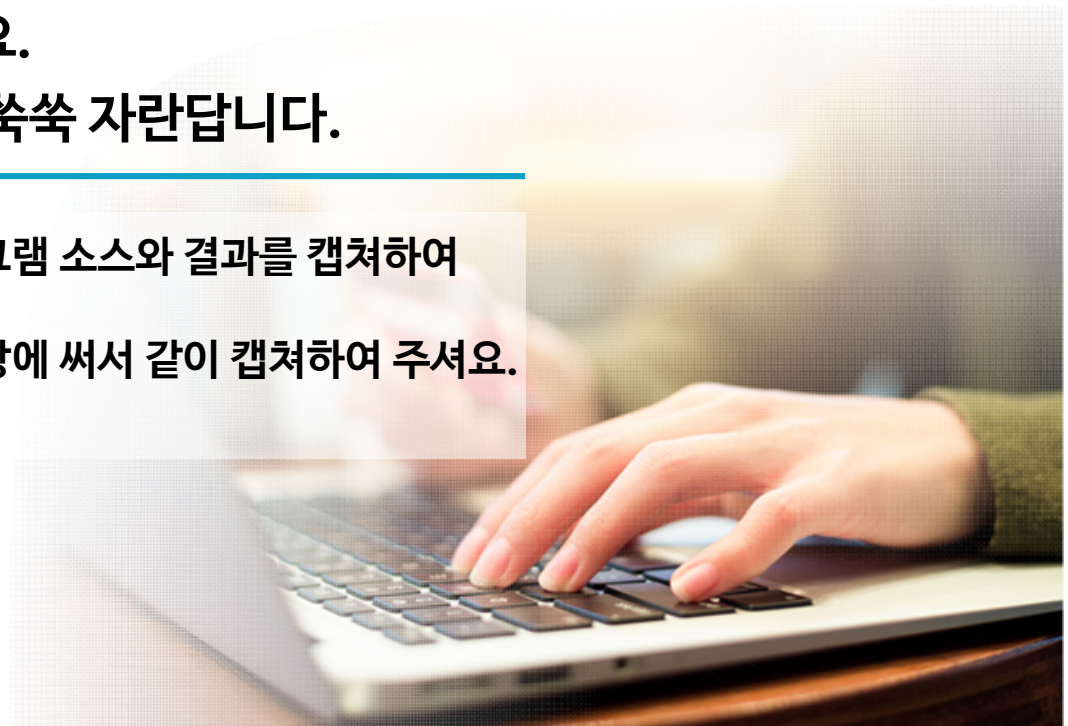
일시정지 버튼을 누른 후, 아래의 학습활동에 참여하세요.

Q

오늘 배운 내용을 스스로 실습하여  
자유게시판에 올려 주세요.

이렇게 정리하면 실력이 쑥쑥 자란답니다.

- ① 본인이 실습한 내용을 프로그램 소스와 결과를 캡처하여 올려주세요.
- ② 본인의 학번과 이름을 메모장에 써서 같이 캡처하여 주세요.
- ③ 그리고 설명도 달아 주세요.





## 학습활동에 대한 교수님 의견

Q

오늘 배운 내용을 스스로 실습하여 자유게시판에 올려 주세요.  
이렇게 정리하면 실력이 쑥쑥 자란답니다.

A

[ 오늘 학습한 내용의 실습 사항 ]

- ① 딕셔너리 - 정의
- ② 딕셔너리 - 예제
- ③ 집합 - 정의
- ④ 집합 - 연산
- ⑤ 컬렉션 - 컬렉션으로 변환, 함수 인자 전달
- ⑥ 컬렉션 - copy, deepcopy

# 학습 평가

Q1

Q1

Q2

Q3

Q4

Q5

다음 a를  $a = \{\text{"이름": "홍길동", "국어": 100, "영어": 88, "수학": 95}\}$ 로 정의하였을 때 a의 자료형은?

- 1 리스트
- 2 튜플
- 3 딕셔너리
- 4 집합



# 학습 평가

Q1

Q1

Q2

Q3

Q4

Q5

다음 a를  $a = \{\text{"이름": "홍길동", "국어": 100, "영어": 88, "수학": 95}\}$ 로 정의하였을 때 a의 자료형은?

- 1 리스트
- 2 튜플
- ☒ 3 딕셔너리
- 4 집합

정답

3번

해설

딕셔너리는 키(key)와 값(value)을 한 쌍으로 연관 지어 사용하는 자료형입니다.

# 학습 평가

Q1

Q2

Q3

Q4

Q5

## Q2

다음 중 딕셔너리에서 해당 키의 값을  
검색하며 없는 경우의 출력내용을  
지정 가능한 함수는?

- 1 items()
- 2 values()
- 3 get()
- 4 keys()



# 학습 평가

Q1

Q2

Q3

Q4

Q5

## Q2

다음 중 딕셔너리에서 해당 키의 값을  
검색하며 없는 경우의 출력내용을  
지정 가능한 함수는?

- 1 items()
- 2 values()
- ☒ 3 get()
- 4 keys()

정답

3번

해설

get()은 해당 키의 값을 검색하며 없는 경우의  
출력내용의 지정이 가능합니다.

# 학습 평가

Q1

Q2

Q3

Q4

Q5

## Q3

다음  $a$ 를  $a=\{1,2,3,4\}$ 로 정의하였을 때  
 $a$ 의 자료형은?

- 1 리스트
- 2 튜플
- 3 딕셔너리
- 4 집합



# 학습 평가

Q1

Q2

Q3

Q4

Q5

Q3

다음  $a$ 를  $a=\{1,2,3,4\}$ 로 정의하였을 때  
 $a$ 의 자료형은?

1 리스트

2 튜플

3 딕셔너리

☒ 4 집합

정답

4번

해설

집합은 대괄호  $\{\}$ 로 원소를 표현하며 순서에  
상관없이 표시 및 사용됩니다.



# 학습 평가

Q1

Q2

Q3

Q4

Q5

## Q4

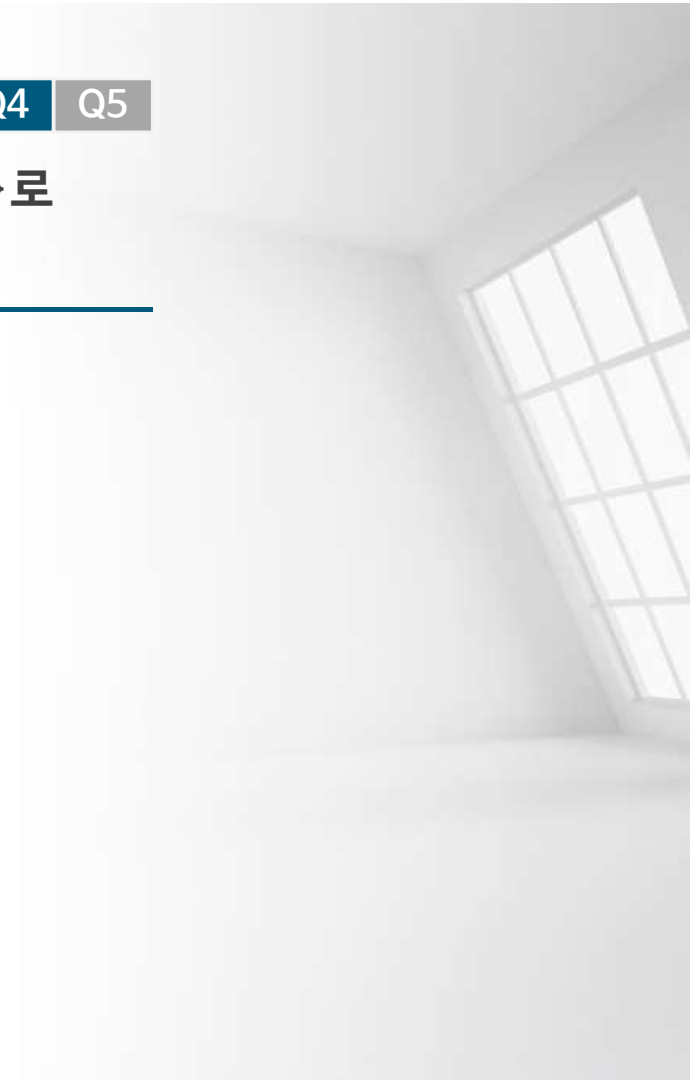
다음  $a$ 를  $a=\{1,2,3,4\}$ ,  $b=\{3,4,5,6\}$ 로 정의하였을 때 다음 중 올바른 것은?

1  $A \setminus B = \{1,2\}$

2  $A - B = \{3,4\}$

3  $A \cap B = \{3,4\}$

4  $A \cap B = \{3,4\}$



# 학습 평가

Q1

Q2

Q3

Q4

Q5

## Q4

다음  $a$ 를  $a=\{1,2,3,4\}$ ,  $b=\{3,4,5,6\}$ 로 정의하였을 때 다음 중 올바른 것은?

1  $A \setminus B = \{1,2\}$

2  $A - B = \{3,4\}$

☒ 3  $A \cap B = \{3,4\}$

4  $A \cup B = \{3,4\}$

정답

3번

해설

집합연산에서 합집합 ( $\cup$ ), 교집합 ( $\cap$ ), 차집합 ( $-$ ) 을 표시합니다.

# 학습 평가

Q1

Q2

Q3

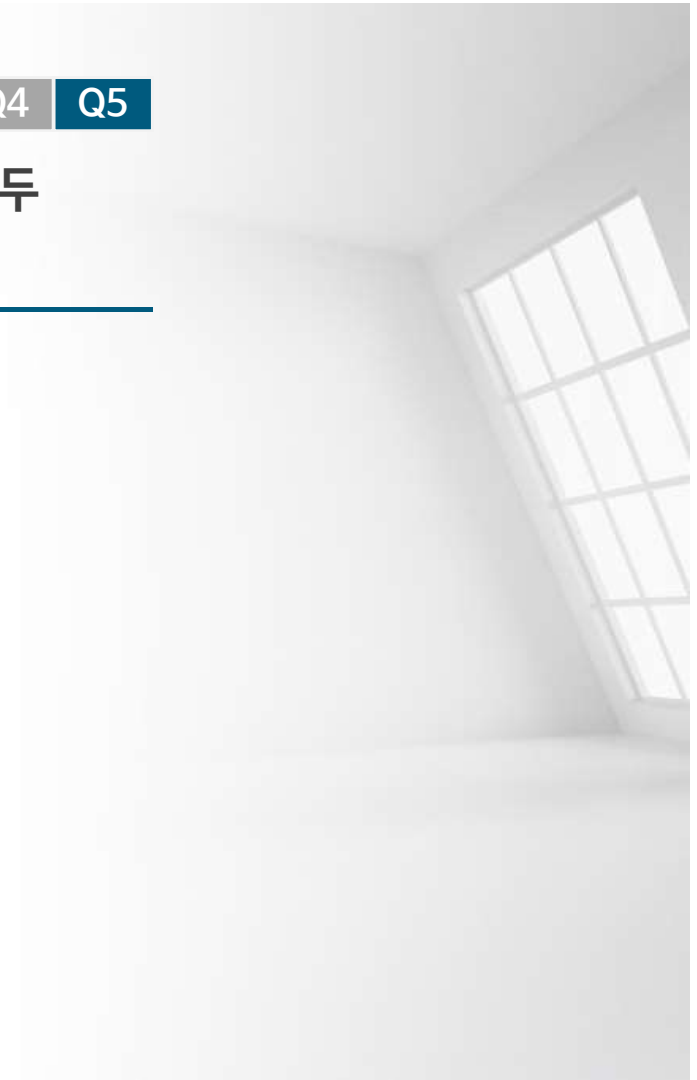
Q4

Q5

## Q5

다음 중 컬렉션 내부의 컬렉션까지 모두 복사하는 함수는?

- 1 `b=a{;}`
- 2 `b=a[:]`
- 3 `b=a.copy()`
- 4 `B=copy.deepcopy(a)`



# 학습 평가

Q1

Q2

Q3

Q4

Q5

## Q5

다음 중 컬렉션 내부의 컬렉션까지 모두 복사하는 함수는?

1 `b=a{;}`

2 `b=a[:]`

3 `b=a.copy()`



`B=copy.deepcopy(a)`

정답

4번

해설

아래레벨의 모든 컬렉션을 복사할 경우 `deepcopy`를 사용합니다.



## 정리하기

### 딕셔너리

- ✓ 딕셔너리는 키(key)와 값(value)을 한 쌍으로 연관지어 사용하는 자료형
- ✓ `keys()`, `values()`, `items()`, `get()` 함수를 활용함

### 집합

- ✓ 수학적 의미의 집합과 동일하며 대괄호 `{}`로 원소를 표현하며 순서에 상관없이 표시 및 사용됨
- ✓ `set()`, `add()`, `remove()` 함수를 사용함
- ✓ 집합연산의 합집합(`|`), 교집합(`&`), 차집합(`-`) 연산을 사용함





## 정리하기

### 컬렉션

- ✓ `enumerate()`, `map()`을 활용하여 쌍을 이루는 컬렉션으로 변환함
- ✓ `filter()`, `map()`, `lambda`를 활용하여 함수인자에 컬렉션값을 차례로 대입함
- ✓ `copy`, `deepcopy()`를 활용하여 컬렉션의 값을 대입함

