2025.04.08. SR 파이썬 스터디 2주차

조코딩 02(2). 튜플, 딕셔너리, 집합, 불

✅ 튜플(tuple)

- 리스트와의 공통점: 자료형을 묶은 자료형.
- 리스트와의 차이점: 구성요소 변경이 불가능함. 소괄호 ()를 사용해서 묶음.

인덱싱, 슬라이싱은 가능.
이유: 인덱싱(a[0]), 슬라이싱(a[0:1])은 변수 자체를 바꾸는 기능이 아니기 때문!

sort, remove, del, pop 등은 불가능!!

실습 예시

```
t = (1, 2, 3, 4)
print(t[0])
print(t[0:4])
print(t[0::2])
print(t)

t = (1, 2, 3, 4)
t[0] = 1 # 에러 날 것임!!
```

In [2]:
```
t = (1, 2, 3, 4)
print(t[0: 3])
```

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

✅ 딕셔너리(dict)

- 형태: {Key1: Value1, Key2: Value2, ...}
- 리스트와의 공통점: 자료형을 묶은 자료형. 구성요소 추가 가능. 구성요소 중 Value 값 변경 가능(Key는 변경 불가).
- 리스트와의 차이점: 쌍으로 묶여 있음(Key, Value). 중괄호 {}와 콜론 :을 사용해서 묶음. 순서 개념이 없고 대신 Key가 있음음.

주요 함수

d = {1: 'a', 2: 'b', 3: 'c'}

- **1, 2, 3: Key**
- **'a', 'b', 'c': Value**
- a[1]: 1이라는 Key에에 해당하는 Value를 반환함(불러옴).
- a[4] = 'd': 4라는 Key, 'd'라는 Value를 가진 새로운 구성요소를 추가함.
- del a[1]: 1이라는 Key를 가진 구성요소를 삭제함.
- del a[2], a[3]: 여러 개 삭제함.

- d.keys(): d의 Key들만 불러옴.
- d.values(): d의 Value들만 불러옴.
- d.items(): d의 Key와 Value들을 iterable(반복문에서 사용 가능)한 형태로 불러옴.
- d.get('a'): d에 'a'라는 Key가 있으면 그 Value를 반환함. 없어도 에러가 나지 않고 None을 반환함.

주의사항

- Key는 유일해야 함.
- Key값엔 변경 불가능한 자료형(숫자형, 문자열, 튜플 등)만 들어갈 수 있음.
- 리스트, 튜플엔 존재했던 순서 개념이 없음. 그래서 d[0] 같은 식으로 불러올 수 없음. 대신 Key를 사용함.

실습 예시

```
d = {'박석주': '계량팀', '박성민': '투자팀', '박주연': '투자팀',
'송수암': '투자팀',
     '이제성': '계량팀', '최성환': '계량팀', '최인재': '계량팀'}
print(d)
type(d)

d[0] # 에러 뜸

d['박석주']

d['홍길동'] = '계량팀'
d

del d['홍길동']
d

print(d.keys())
print(d.values())

d['박석주'] == '투자팀'
```

In [ ]:

✅ 집합(set)

- 형태: {1, 2, 3}. 중괄호에 값만 나열. 고등학교 때 배운 집합 표현 방법과 동일함.
- 리스트와의 공통점: 자료형을 묶은 자료형. 구성요소 삭제 및 추가 가능
- 리스트와의 차이점: 순서 개념이 없음(위치 확인 불가). 중괄호 {}을 사용해서 묶음. 값 중복 저장 안 됨. 집합 연산 가능.

집합 연산

- a & b, a.intersection(b): 교집합
- a | b, a.union(b): 합집합
- a - b, a.difference(b): 차집합

주요 함수

- a.add(b): 집합 a에 자료 b를 추가함
- a.remove(b): 집합 a에서 자료 b를 제거함

주의사항

- 집합 안의 값들은 유일함. 만약 같은 값이 여러 개 입력됐다면 하나만 저장됨.
- 순서 개념이 없음. 그래서 종종 변수를 저장하고 인출해 보면 처음 입력한 순서와 다르게 나오는 걸 확인할 수 있음.
- 변경 불가능한 자료형만 들어갈 수 있음.

In [3]:
```
a = {1, 2, 3}
b = {3, 4, 5}

a - b
```

Out[3]:
```
{1, 2}
```

✅ 불(bool, boolean)

- 불리언, 논리값이라고도 불림.
- 수학, 논리학에선 진릿값, 진리치라고도 불림.
- True, False 두 가지 형태만 가짐.

비교연산자

- 주어진 명제가 참이면 True 반환, 거짓이면 False 반환
- 파이썬에서 '='는 변수를 지정하는 기호이기 때문에, '=' 대신에 '=='를 사용
- a == b: a와 b는 같은가?
- a != b: a와 b는 다른가?
- a > b
- a < b
- a <= b
- a >= b

논리연산자

- and: 둘 다 참이면 True
- or: 둘 중 하나 이상 참이면 True
- not: 참, 거짓을 뒤바꿈

각종 자료형의 논리값

- 0: False
- 1, 2, ...: True
- '': False
- '들어있음': True
- []: False

- ['들어있음']: True
- (): False
- ('들어있음'): True
- {}: False
- {'들어있음': '들어있음'}: True
- {}: False
- {"들어있음"}: True
- None: False

실습 예시

```
a = 0
b = 1
type(a)
type(bool(a))

bool(a) == bool(b)

a == b

a is b
a is not b

a = 1 == 2
print(a)
type(a)
```

In [ ]: