2021.02.23

[데이터 구조-II]

3. 딕셔너리(dictionary)

- {"key1":"value1", "key2":"value2" ...}의 형식으로 구성되며, key는 중복이 불가
- 인덱스를 이용한 참조는 지원하지 않는다.

ex) my_dic 딕셔너리에서 key 및 value 출력하기

```
my_dic = { 'name' : 'park' , 'age' : 30, 'tel': '010-0000-0000' }

# 키 목록 출력                    my_dic.keys( )    => dict_keys( ['name', 'age, 'tel'] )
# 값 목록 출력                    my_dic.values( )  => dict_values( ['park', 30, '010-0000-0000'] )
# 키와 값을 튜플 형식으로 출력    my_did.items( )
                                  => dict_items( [('name' : 'park') , ('age' : 30), ('tel': '010-0000-0000')] )
```

4. 셋(set)

- 중복을 허용하지 않는 집합(순서가 존재하지 않는다.)
- { } 혹은 set( )함수를 이용해서 생성
- 연산자: &(교집합) / | (합집합) / - (차집합)

ex) s1, s2 셋에서 교집합, 합집합, 차집합 찾기

```
s1 = {1, 2, 3, 4, 5, 6}   / s2 = {4, 5, 6, 7, 8, 9}
# 교집합      s1 & s2          => {4, 5, 6}
# 합집합      s1 | s2          => {1, 2, 3, 4, 5, 6, 7, 8, 9}
# 차집합      s1 – s2          => {1, 2, 3}
```

5. enumerate

- 반복자 또는 순서 객체로 반복문을 처리할 때, 인덱스 처리를 해결한다.

ex)

```
names = {'kim', 'lee', 'park'}
for idx, name in enumerate(names):
    print( "{ }번째: { }".format (idx, name)
=>  0번째: kim
    1번째: lee
    2번째: park
```