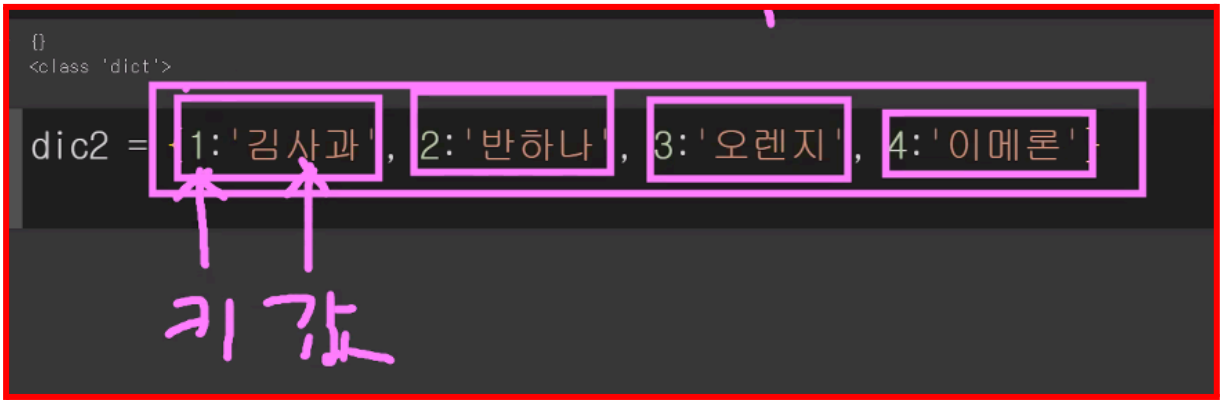
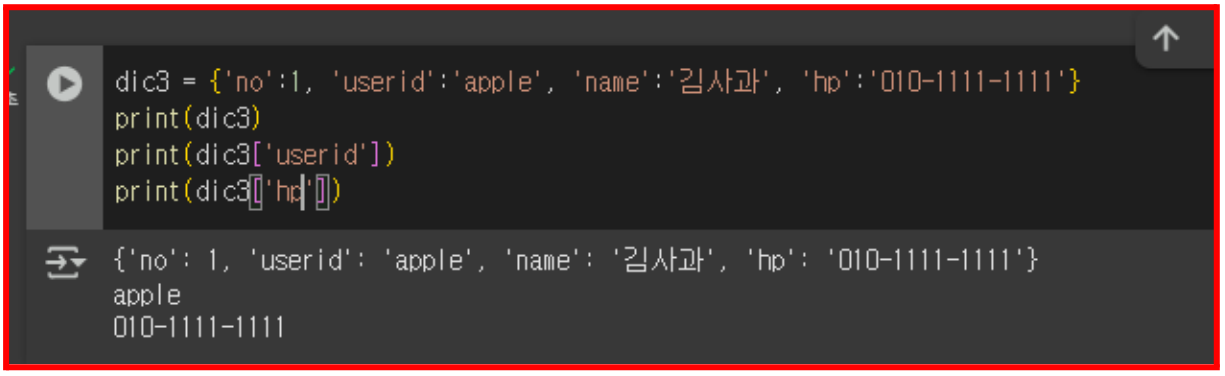
# Day 5 - Python Collection Type (Dictionary)




- Dict은 키와 값으로 저장된다 Key:Value
- 순서가 없다 (set과 같이)




- Key는 자료형에 구애받지 않으며 뭐든 넣을 수 있다 (문자열, 숫자 등).

**Dict 타입은 변경 가능(mutable)하다.**

```
dic4 = {1:'apple'}
print(dic4)

dic4[100] = 'banana' # 값이 없으면 생성된다.
print(dic4)

dic4[100] = 'orange' # 키값이 할당되어 있음으로 값을 변경한다.
print(dic4)

dic4[50] = 'melon'
print(dic4)

del dic4[100] # 삭제
print(dic4)
```

```
{1: 'apple'}
{1: 'apple', 100: 'banana'}
{1: 'apple', 100: 'orange'}
{1: 'apple', 100: 'orange', 50: 'melon'}
{1: 'apple', 50: 'melon'}
```

- 값을 생성, 변경가능하다.
- 또한 삭제도 가능하다.

**딕셔너리의 키 값 제약:**

```
[22] dic3['score'] = [100, 90, 40]
print(dic3)
```

```
{'no': 10, 'userid': 'apple', 'name': '김사과', 'hp'
```

```
# dic3[[10, 20, 30]] = ['십', '이십', '삼십']
# print(dic3)
dic3[(10, 20, 30)] = ['십', '이십', '삼십']
print(dic3)
```

- 키 값은 문자, 정수, 튜플로는 가능하지만 리스트는 불가하다
- 딕셔너리의 값은 어떤 타입이든 가능하다.

```
dic3['과일'] = {'사과':'🍎', '딸기':'🍓', '수박':'🍉'}
print(dic3)
```

**Dictionary의 함수와 메서드:**

**.key()**

```
dic3 = {'no':1, 'userid':'apple', 'name':'김사과', 'hp':'010-1111-1111'}
print(dic3)

# keys(): 딕셔너리의 모든 키를 반환
print(dic3.keys())
```

```
{'no': 1, 'userid': 'apple', 'name': '김사과', 'hp': '010-1111-1111'}
dict_keys(['no', 'userid', 'name', 'hp'])
```

- 모든 키를 반환
- 순서가 있는 리스트로 반환된다. (반복문 사용 가능하며, 리스트와 똑같이 작동한다.)

**.values()**

```
dic3 = {'no':1, 'userid':'apple', 'name':'김사과', 'hp':'010-1111-1111'}
print(dic3)

# values(): 딕셔너리의 모든 값을 반환
print(dic3.values())
```

```
{'no': 1, 'userid': 'apple', 'name': '김사과', 'hp': '010-1111-1111'}
dict_values([1, 'apple', '김사과', '010-1111-1111'])
```

- 모든 값을 리스트 형태로 반환

**.items()**

```
dic3 = {'no':1, 'userid':'apple', 'name':'김사과', 'hp':'010-1111-1111'}
print(dic3)

# items(): 딕셔너리의 모든 키-값을 튜플로 반환
print(dic3.items())
```

```
{'no': 1, 'userid': 'apple', 'name': '김사과', 'hp': '010-1111-1111'}
dict_items([('no', 1), ('userid', 'apple'), ('name', '김사과'), ('hp', '010-1111-1111')])
```

- key:value를 튜플로 반환

**.get()**

```
dic3 = {'no':1, 'userid':'apple', 'name':'김사과', 'hp':'010-1111-1111'}
print(dic3)

# get(): 특정 키에 대한 값을 반환. 만약 키가 딕셔너리에 없으면 None을 반환
# None을 치환할 수 있는 문자열을 설정할 수 있음

print(dic3['userid'])
# print(dic3['gender']) # 이 키의 값이 할당되어 있지 않을경우 에러가 난다.

print(dic3.get('userid'))
print(dic3.get('gender'))
print(dic3.get('gender', '성별 알수없음'))
print(dic3.get('name', '이름 알수없음'))
```

```
{'no': 1, 'userid': 'apple', 'name': '김사과', 'hp': '010-1111-1111'}
apple
apple
None
성별 알수없음
김사과
```

- 특정 키에 대한 값을 반환시키며, 없으면 none을 반환
- 만약 dic3['gender']이런식으로 찾으면 에러가 발생한다.
- 또한 get을 통해 키값이 없을경우 메시지를 반환 가능하다.

**pop() method**

```
dic3 = {'no':1, 'userid':'apple', 'name':'김사과', 'hp':'010-1111-1111'}
print(dic3)

# pop(): 특정 키에 대한 값을 제거하고 제거된 값을 반환. 키가 없다면 에러
temp = dic3.pop('hp')
print(dic3)
print(temp)
```

```
{'no': 1, 'userid': 'apple', 'name': '김사과', 'hp': '010-1111-1111'}
{'no': 1, 'userid': 'apple', 'name': '김사과'}
010-1111-1111
```

- 키에대한 값을 제거하고 반환시킨다.
- 리스트에 있는 pop과 같은 기능이다.

**in function**

```
dic3 = {'no':1, 'userid':'apple', 'name':'김사과', 'hp':'010-1111-1111'}
print(dic3)

# in: 딕셔너리에 특정 키가 있는지 확인
print('hp' in dic3)
print('010-1111-1111' in dic3)
```

```
{'no': 1, 'userid': 'apple', 'name': '김사과', 'hp': '010-1111-1111'}
True
False
```

- 특정 키가 있는지 확인
- True / False 반환