파이썬 프로그래밍

21차시

딕셔너리 메소드

⚠ 학습개요

… 딕셔너리 메소드

- keys(), items(), values(), get(), clear(), update()

⚠ 학습목표

… 딕셔너리에서 키 목록을 활용할 수 있다.
… 딕셔너리에서 키와 값 목록을 활용할 수 있다.
… 딕셔너리에서 값 목록을 활용할 수 있다.
… 딕셔너리에서 키로 값을 알 수 있다.
… 딕셔너리에서 get()으로 값을 알 수 있다.
… 딕셔너리에서 clear()로 모든 목록을 지울 수 있다.
… 딕셔너리에서 update()를 활용할 수 있다.

Chapter 1.

딕셔너리 메소드

PYTHON PROGRAMMING

⚠ 메소드 keys()

+ 키로만 구성된 리스트를 반환

- for문에서 시퀀스 위치에 메소드 keys()를 사용하면 **딕셔너리의 모든 항목을 참조하는 구문을 사용**
  다음에서 월, 화, … 키 위치에는 숫자는 올 수 없음

```
>>> day = dict(월='monday', 화='tuesday', 수='wednesday', 목='thursday')
>>> print(day)
{'월': 'monday', '화':'tuesday', '수':'wednesday', '목': 'thursday'}
>>> print(day.keys())
dict_keys(['월', '화', '수', '목'])

>>> for key in day.keys():
...     print('%요일 %s' % (key, day[key]))
...
월요일 monday
화요일 tuesday
수요일 wednesday
목요일 thursday
```

⚠ 메소드 items()

+ (키, 값) 쌍의 튜플이 들어 있는 리스트를 반환

- 각 튜플의 첫 번째 항목은 키, 두 번째 항목은 키 값

```
>>> day = dict(월='monday', 화='tuesday', 수='wednesday', 목='thursday')
>>> print(day.items())
dict_items([('월','monday'),('화', 'tuesday'),('수', 'wednesday'),('목', 'thursday')])

>>> for key, value in day.items():
...     print('%요일 %s' % (key, value))
...
월요일 monday
화요일 tuesday
수요일 wednesday
목요일 thursday
```

⚠ 메소드 values()와 for문

+ 딕셔너리 메소드 values()

- 딕셔너리 메소드 values()는 값으로 구성된 리스트를 반환

```
>>> day = dict(월='monday', 화='tuesday', 수='wednesday', 목='thursday')
>>> print(day.values())
dict_values(['monday', 'tuesday', 'wednesday', 'thursday'])
```

⚠ 딕셔너리만으로 for문 사용

+ 반복 for문

- 시퀀스 위치에 있는 딕셔너리 변수만으로도 모든 키를 순회

```
>>> game = dict(일월='소나무', 이월='매화', 삼월='벚꽃', 사월='등나무')
>>>
>>> for key in game:
...     print('%s: %s' % (key, game[key]))
...
일월: 소나무
이월: 매화
삼월: 벚꽃
사월: 등나무
```

⚠ 일상 코딩: 사계절 단어의 한영 사전을 만들어 항목 순회

[코딩실습] 사계절의 영어 사전 생성과 항목 순회 난이도 응용

```
1  season = {'봄':'spring', '여름':'summer', '가을':'autumn', '겨울':
2  'winter'}
3  print(season.keys())
4  print(season.items())
5  print(season.values())
6
7  # 메소드 keys()로 항목 순회
8  for key in season.keys():
9      print('%s %s  ' % (key, season[key]))
10
11 for item in season.items():
12     print('{} {}  '.format(item[0], item[1]), end= ' ')
13 print()
14 # 메소드 items()의 반환 값인 튜플을 한 변수에 저장한 경우, 항목 순회 2
15 for item in season.items():
16     print('{} {}  '.format(*item), end= ' ')
17 print()
```

⚠ 일상 코딩: 사계절 단어의 한영 사전을 만들어 항목 순회

[코딩실습] 사계절의 영어 사전 생성과 항목 순회 난이도 응용

결과

```
dict_keys(['봄', '여름', '가을', '겨울'])
dict_items([('봄', 'spring'), ('여름', 'summer'), ('가을',
'autumn'), ('겨울', 'winter')])
dict_values(['spring', 'summer', 'autumn', 'winter'])
봄 spring
여름 summer
가을 autumn
겨울 winter
봄 spring   여름 summer   가을 autumn   겨울 winter
봄 spring   여름 summer   가을 autumn   겨울 winter
```

⚠ 메소드 get()

✚ 키로 조회하는 딕셔너리 메소드 get(키[, 키가_없을_때_반환 값])

```
>>> city = {'대한민국':'부산', '뉴질랜드':'웰링톤', '캐나다':'몬트리올'}
>>> city.get('대한민국')
'부산'
```

✚ 메소드 get(키)은 딕셔너리에 키가 없어도 오류가 발생하지 않고 아무것도 없다는 의미인 None을 반환

- 만일 키 뒤에 다른 인자를 넣으면 딕셔너리에 키가 없을 때 이 지정된 값을 반환

```
>>> city.get('미국')
>>> city.get('미국‘,'없네요')
'없네요'
```

⚠ 메소드 clear()와 문장 del

+ 모든 항목을 제거하는 딕셔너리 메소드 clear()

- 딕셔너리 메소드 clear()는 기존의 모든 키 : 값 항목을 삭제

```
>>> city = {'대한민국':'부산', '뉴질랜드':'웰링톤', '캐나다':'몬트리올'}
>>> city.clear()
>>> city
{}
```

+ 문장 del로 딕셔너리 변수 자체 제거

```
>>> city = {'대한민국':'부산', '뉴질랜드':'웰링톤'}
>>> del city
>>> city
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
NameError: name 'city' is not defined
```

⚠ 메소드 pop(키)와 popitem()

✚ 키의 값을 반환하고 키 항목 삭제 메소드 pop(키)

```
>>> city = {'대한민국':'부산', '뉴질랜드':'웰링톤', '캐나다':'몬트리올'}
>>> city.pop('뉴질랜드')
웰링턴
```

✚ 마지막으로 삽입된 키의 (키, 값) 튜플을 반환하고 키 항목 삭제 메소드 popitem()

```
>>> city = {'대한민국':'부산', '뉴질랜드':'웰링톤'}
>>> city.popitem()
('뉴질랜드', '웰링톤')
```

⚠ 일상 코딩: 사계절 단어의 한영 사전을 만들어 항목 순회

[코딩실습] 색상 사전의 조회와 삭제 난이도 기본

```
1  color = dict(검은색='black', 흰색='white', 녹색='green', 파란색='blue')
2  print(color)
3
4  # 항목 조회
5  print(color.get('녹색'))
6  print(color.get('노란색'))
7  print()
8
9  # 항목 추가
10 color['노란색'] = 'yellow'
11 print(color)
12 print()
13
14 # 항목 삭제
15 c = '흰색'
16 print('삭제: %s %s' % (c, color.pop('흰색')))
17 print(color)
```

⚠ 일상 코딩: 사계절 단어의 한영 사전을 만들어 항목 순회

[코딩실습] 색상 사전의 조회와 삭제

난이도 기본

```
18 c = '빨간색'
19 print('삭제: %s %s' % (c, color.pop(c, '없어요')))
20
21 print('임의 삭제: {} '.format(color.popitem()))
22 print('임의 삭제 후: {} '.format(color))
23
24 c = '검은색'
25 del color[c]
26 print('{} 삭제 후: {}'.format(c, color))
27
28 # 모든 항목 삭제
29 color.clear()
30 print(color)
31
```

⚠ 일상 코딩: 사계절 단어의 한영 사전을 만들어 항목 순회

[코딩실습] 색상 사전의 조회와 삭제 난이도 기본

결과

```
{'검은색': 'black', '흰색': 'white', '녹색': 'green', '파란색':
'blue'}
green
None

{'검은색': 'black', '흰색': 'white', '녹색': 'green', '파란색':
'blue', '노란색': 'yellow'}

삭제: 흰색 white
{'검은색': 'black', '녹색': 'green', '파란색': 'blue', '노란색':
'yellow'}
삭제: 빨간색 없어요
임의 삭제: ('노란색', 'yellow')
임의 삭제 후: {'검은색': 'black', '녹색': 'green', '파란색': 'blue'}
검은색 삭제 후: {'녹색': 'green', '파란색': 'blue'}
{}
```

⚠ 딕셔너리를 결합하는 메소드 update( )

+ 메소드 update(다른 딕셔너리)

- 인자인 다른 딕셔너리를 합병

```
>>> kostock = {'Samsung Elec.': 40000, 'Daum KAKAO': 80000}
>>> usstock = {'MS': 150, 'Apple': 180}
>>> kostock.update(usstock)
>>> kostock
{'Samsung Elec.': 40000, 'Daum KAKAO': 80000, 'MS': 150, 'Apple': 180}
```

+ 인자 딕셔너리에 원 딕셔너리와 동일한 키가 있다면 인자 딕셔너리의 값으로 대체

```
>>> usstock.update({'MS': 200})
>>> usstock
{'MS': 200, 'Apple': 180}
```

⚠ 딕셔너리 메소드

- keys()
- items()
- values()
- get()
- clear()
- pop() popitem()
- update()