10장. 사전과 집합

파이썬 정복

Contents

❖ 목차

1. 사전

❖ 사전 (Dictionary)

- 키와 값의 쌍을 저장하는 대용량 자료구조
- 맵 / 연관배열
- 중괄호 안에 키:값 형태로 콤마로 구분하여 나열

dic

```
dic = { 'boy':'소년', 'School':'학교', 'book':'책' }
print(dic)
```

실행결과

```
{'book': '책', 'School': '학교', 'boy': '소년'}
```

- 빠른 검색 가능
  - [키]

dicread

```
dic = { 'boy':'소년', 'School':'학교', 'book':'책' }
print(dic['boy'])
print(dic['book'])
```

실행결과

```
소년
책
```

1. 사전

- 찾는 키가 없을 경우 예외 발생
  - 예외 처리 구문
  - get 메서드

dicget

```
dic = { 'boy':'소년', 'school':'학교', 'book':'책' }
print(dic.get('student'))
print(dic.get('student', '사전에 없는 단어입니다.'))
```

실행결과

```
None
사전에 없는 단어입니다.
```

- 특정 키 검색 시에는 in 구문 사용

1. 사전

❖ 사전 관리

- 실행 중 삽입, 삭제, 수정 등 편집 가능

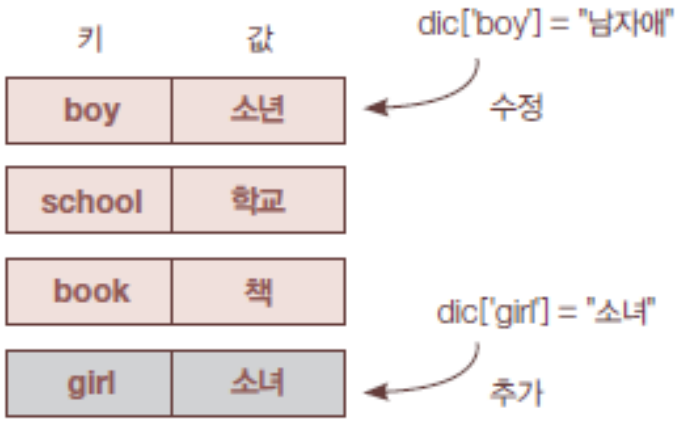
dicchange

```
dic = { 'boy':'소년', 'School':'학교', 'book':'책' }
dic['boy'] = '남자애'
dic['girl'] = '소녀'
del dic['book']
print(dic)
```

실행결과

```
{'boy': '남자애', 'girl': '소녀', 'School': '학교'}
```

- 사전[키]
  - 키의 존재 여부에 따라 동작 다름
    - 존재할 경우 : 기존 값의 변경
    - 존재하지 않을 경우 : 키를 추가



1. 사전

- **del**
  - 해당 키를 찾아 값과 함께 삭제
- **keys / values 메서드**
  - 사전의 키 / 값 목록 얻음

keys

```
dic = { 'boy':'소년', 'school':'학교', 'book':'책' }
print(dic.keys())
print(dic.values())
print(dic.items())
```

실행결과

```
dict_keys(['book', 'boy', 'school'])
dict_values(['책', '소년', '학교'])
dict_items([('book', '책'), ('boy', '소년'), ('school', '학교')])
```

1. 사전

- dict_* 객체
  - 리스트처럼 순회하여 값 순서대로 읽음

keylist

```
dic = { 'boy':'소년', 'School':'학교', 'book':'책' }
keylist = dic.keys()
for key in keylist:
    print(key)
```

실행결과

```
boy
School
book
```

- update 메서드
  - 두 개 사전을 병합
- dict 함수
  - 빈 사전 만들거나 다른 자료형을 사전으로 변환

1. 사전

❖ 사전의 활용

- Ex) 노래 가사의 특정 알파벳 출현 횟수

alphanum

```
song = """by the rivers of babylon, there we sat down
yeah we wept, when we remember zion.
when the wicked carried us away in captivity
required from us a song
now how shall we sing the lord's song in a strange land"""

alphabet = dict()
for c in song:
    if c.isalpha() == False:
        continue
    c = c.lower()
    if c not in alphabet:
        alphabet[c] = 1
    else:
        alphabet[c] += 1

print(alphabet)
```

실행결과

```
{'b': 4, 'y': 5, 't': 9, 'h': 9, 'e': 22, 'r': 12, 'i': 10, 'v':
2, 's': 10, 'o': 10, .....
```

2. 집합

❖ 집합 정의

- 여러 가지 값의 모임

set

```
asia = { 'korea', 'china', 'japan', 'korea' }
print(asia)
```

실행결과

```
{'korea', 'china', 'japan'}
```

- set() 함수
  - 빈 집합 만들거나 다른 컬렉션을 집합형으로 변환

set2

```
print(set("sanghyung"))
print(set([12, 34, 56, 78]))
print(set(("신지희", "한주완", "김태륜")))
print(set({'boy':'소년', 'School':'학교', 'book':'책'}))
print(set())
```

실행결과

```
{'u', 'n', 'a', 's', 'y', 'h', 'g'}
{56, 34, 12, 78}
{'김태륜', '신지희', '한주완'}
{'boy', 'School', 'book'}
set()
```

2. 집합

- 인수 없이 set() 함수 호출
  - 공집합 만들기

- **add 메서드**
  - 집합에 원소 추가
- **update 메서드**
  - 집합끼리 결합하여 합집합 만들기
  - 중복 허용되지 않음에 유의

2. 집합

❖ 집합 연산

▪ 연산을 통한 집합 간 조합

연산	기호	메서드	설명
합집합	|	union	두 집합의 모든 원소
교집합	&	intersection	두 집합 모두에 있는 원소
차집합	-	difference	왼쪽 집합의 원소 중 오른쪽 집합의 원소를 뺀 것
배타적 차집합	^	symmetric_difference	한쪽 집합에만 있는 원소의 합

연산	기호	메서드	설명
부분집합	<=	issubset	왼쪽이 오른쪽의 부분집합인지 조사한다.
진성 부분집합	<		부분집합이면서 여분의 원소가 더 있음
포함집합	>=	issuperset	왼쪽이 오른쪽 집합을 포함하는지 조사한다.
진성 포함집합	>		포함집합이면서 여분의 원소가 더 있음

2. 집합

setop

```
twox = { 2, 4, 6, 8, 10, 12 }
threex = { 3, 6, 9, 12, 15 }

print("교집합", twox & threex)
print("합집합", twox | threex)
print("차집합", twox - threex)
print("차집합", threex - twox)
print("배타적 차집합", twox ^ threex)
```

실행결과

```
교집합 {12, 6}
합집합 {2, 3, 4, 6, 8, 9, 10, 12, 15}
차집합 {8, 2, 10, 4}
차집합 {9, 3, 15}
배타적 차집합 {2, 3, 4, 8, 9, 10, 15}
```

Thank You !

파이썬 정복