파이썬 프로그래밍

Sets, Tuples, Dictionaries

광운대학교
KwangWoon University

Sets

- **집합(Set)**: 수학에서 배운 집합의 성격을 따름
- 여러 element들이 모여있으며, **순서가 없음**
- 모든 element가 **달라야 함**

이러한 집합을 파이썬에서 구현한 것이 **set**

set 예제

```
my_set = {1, 2, 3, 4, 5}
print(my_set)  # {1, 2, 3, 4, 5}

# 중복된 요소는 자동으로 제거
my_set = {1, 2, 2, 3, 4, 4, 5}
print(my_set)  # {1, 2, 3, 4, 5}
```

set 예제

```
fruits = set(["mango", "banana", "apple"])

for element in fruits:
  print(element)
```

set 예제

```
num = set([1,2,3,4,5,6])
odd = set([1,3,5,7,9,11])

num.add(7)

num.add(7)
num.remove(3)
num.clear()
num.issubset(odd)
num.difference(odd)
num.intersection(odd)
num.union(odd)
```

set의 mutability와 immutability

- **Mutability (변경 가능성)**: 객체의 상태를 변경할 수 있는 속성
- **Immutability (변경 불가능성)**: 객체의 상태를 변경할 수 없는 속성

set은 Mutability와 Immutability를 모두 가지고 있다.

```
subject_list = ["korean", "math"]

subject = {subject_list, "eng"} # TypeError: unhashable type: 'list'
subject[0] = "P.E." #TypeError: 'set' object does not support item assignment
```

예제

team_A와 team_B에 둘 다 속한 사람을 찾는 코드

```
team_A = ["alice@example.com", "bob@example.com", "charlie@example.com"]
team_B = ["bob@example.com", "david@example.com", "alice@example.com"]

def checkDuplication(list1, list2):
    # 코드 작성
    return

print(checkDuplication(list1, list2))
# 출력: {'alice@example.com', 'bob@example.com'}
```

예제

team_A와 team_B에 둘 다 속한 사람을 찾는 코드

```
team_A = ["alice@example.com", "bob@example.com", "charlie@example.com"]
team_B = ["bob@example.com", "david@example.com", "alice@example.com"]

def checkDuplication(list1, list2):
    set1 = set(team1)
    set2 = set(team2)
    return set1 & set2

print(checkDuplication(list1, list2))
```

Tuples

- **변경 불가능**: 생성 후 요소를 변경, 추가, 삭제가 불가능
- **순서가 있음**: 인덱스를 통해 요소에 접근 가능
- **중복 허용**: 동일한 값을 여러 번 포함 가능.
- **소괄호** `()` 를 사용하여 정의

Tuple 사용 방법

1. 빈 Tuple 생성: `tup = ()`
2. 하나의 요소를 가진 Tuple 생성: `tup = (1,)`
3. 여러 요소를 가진 Tuple 생성: `tup = (1, 2, 3)`
4. Tuple의 요소에 접근: `print(tup[1])`

tuple의 immutability

```
class = ("korean", "math", "eng")
class[0] = "P.E." # SyntaxError: invalid syntax

classNgrade = (["korean", 80], ["math", 85])
classNgrade[0][1] = 90
```

Tuple과 리스트 비교

특징	Tuple	리스트
변경 가능성	변경 불가능	변경 가능
기호	()	[]
용도	고정된 데이터 사용	자주 변경되는 데이터 사용

Dictionarys

- 파이썬에서 자주 사용하는 자료구조
- 순서가 있고 mutable한 element를 중복 없이 저장
- 키(key)와 값(value)의 쌍으로 데이터를 저장

dictionary 특징

- key값은 고유해야 하고, immutable 하며, value는 mutalbe하다.
- key값을 리스트에서의 인덱스라고 생각해도 무방

```
student = {
   "철수": 70,
   "영희": 90,
   "민수": 80,
}
```

dictionary 메서드

```
print(dict.keys())   # key들을 반환
print(dict.values()) # value들을 반환
print(dict.items())  # key-value 튜플 형태로 반환
print(dict.update({"민준": 95})) # 딕셔너리 업데이트
print(dict.pop("민준")) # 해당 key-value 삭제
print(dict.get("철수")) # 해당 key의 value 반환
print(dict.clear())  # 딕셔너리 초기화
```

예제

학생의 이름과 성적이 저장된 딕셔너리에서
최고 성적을 가진 학생의 이름과 성적을 출력

```
students = {
   "철수": 85,
   "영희": 90,
   "민수": 98,
   "민지": 92
}


print(f"최고 성적 학생: { }, 성적: { }")
```

예제

```
students = {
   "철수": 85,
   "영희": 90,
   "민수": 98,
   "민지": 92
}

best_student = max(students, key=students.get)
best_score = students[best_student]

print(f"최고 성적 학생: {best_student}, 성적: {best_score}")
```