컴퓨터시스템입문

김경민

변수와 자료형

- **변수**
  - 값을 저장하는 공간
- **변수명**
  - 숫자, 알파벳, 밑줄 기호(_), 한글 등 다양한 문자를 사용
  - 연산자로 사용되는 기호(+, -, = 등)와 공백 문자()는 사용할 수 없음
  - 숫자로 시작해서는 안됨
  - 시스템 예약어(for, if, while 등)를 이름으로 사용할 수 없음
- **변수 선언 및 초기화**
  - 자료형을 명시하지 않아도 자동으로 결정
  - 자료형 확인
    - Type(변수명)

```
x = 3              #정수형 변수 선언
x = 1.3            #실수형 변수 선언
x = "안녕"         # 문자형 변수 선언
x = True           # bool형 변수 선언
x, y = 10, 20.5
```

이스케이프 시퀀스(escape sequence)

- ₩와 특정 문자를 결합하여 문자를 표시해 주는 문자

escape sequence	의미
₩₩	백슬래시
₩'	작은 따옴표
₩"	큰 따옴표
₩n	개행(엔터)
₩t	탭

문자열 연산

```
str1 = "Hello"
str2 = "World!!"

#문자열 더하기
print(str1 + str2)

#문자열 곱하기
print("=" * 20)

#문자열 길이 구하기
print(len(str1))

#문자열 인덱싱
print(str1[0])
print(str1[-1])

#문자열 슬라이싱
print(str1[1:4])    #마지각 항목 제외
print(str1[:3])     #처음부터 추출 마지각 항목 제외
print(str1[3:])     #마지막까지 추출

#문자열의 요소 값은 변경 불가
#str1[0] = "h"      #오류
```

```
HelloWorld!!
====================
5
H
o
ell
Hel
lo
```

문자열 관련 함수

```
name = input("이 름 을  입 력 하 세 요 .")
print(name)

#문 자 열  길 이
print(len(name))

#문 자  갯 수
print(name.count('김 '))

#위 치  찾 기 c
print(name.find('김 '))   #찾 는  문 자 없 으 면  -1
#print(name.index('김 '))  #찾 는  문 자 없 으 면  오 류

#대 소 문 자  변 경
print(name.upper())
print(name.lower())

#공 백  지 우 기
print(name.strip())      #양 쪽  공 백  제 거
print(name.lstrip())     #왼 쪽  공 백  제 거
print(name.rstrip())     #오 른 쪽  공 백  제 거

#문 자 열  바 꾸 기
print(name.replace(" ","."))

#문 자 열  삽 입
name = ",".join(name.strip().replace(" ",""))

#문 자 열  나 누 기
print(name.split(","))
```

```
 a b c
7
0
-1
 A B C
 a b c
a b c
a b c
 a b c
,a,b,c,
['a', 'b', 'c']
```

리스트(list)

- 변수에 값을 저장할 때 [ ](대괄호)로 묶고 각 값을 ,(콤마)로 구분
  - 요소(element) : 리스트에 저장된 각 값

```
#빈 리스트 만들기 0
item1 = []
item2 = list()

#range()함수를 이용하여 만들기
item1 = list(range(3))
print(item1)

item1 = list(range(1,4))
print(item1)

item2 = list(range(3, 0, -1))
print(item2)

#리스트 연산
item3 = item1 + item2
print(item3)

item3 = item1 * 3
print(item3)

#리스트 값 수정
item1[0] = 100
print(item1)

item2[:2] = [200, 300]
print(item2)

#리스트 값 삭제
del(item3[:3])
print(item3)
```

```
[0, 1, 2]
[1, 2, 3]
[3, 2, 1]
[1, 2, 3, 3, 2, 1]
[1, 2, 3, 1, 2, 3, 1, 2, 3]
[100, 2, 3]
[200, 300, 1]
[1, 2, 3, 1, 2, 3]
```

리스트관련 함수

```
item1 = [10, 2, 3]

#리스트 자료 추가 -
item1.append(4)
print(item1)

item1.append([5,6])
print(item1)

#리스트 자료 삭제
del(item1[4])
print(item1)

#리스트 요소 뒤집기
item1.reverse()
print(item1)

#리스트 정렬
item1.sort()
print(item1)

item1.sort(reverse = True)
print(item1)
```

```
#리스트 요소 찾기
print(item1.index(10))      #없으면 오류

#리스트 요소 추기은
item1.insert(2,90)
print(item1)

#리스트 요소 제거
item1.remove(4)
print(item1)

#리스트 끄집어 내고 삭제
item1.pop()
print(item1)

item1.pop(0)
print(item1)

#리스트 확장
item1.extend([3,3,3,3])
print(item1)

#리스트 개수 세기,
print(item1.count(3))
```

```
[10, 2, 3, 4]
[10, 2, 3, 4, [5, 6]]
[10, 2, 3, 4]
[4, 3, 2, 10]
[2, 3, 4, 10]
[10, 4, 3, 2]
0
[10, 4, 90, 3, 2]
[10, 90, 3, 2]
[10, 90, 3]
[90, 3]
[90, 3, 3, 3, 3, 3]
5
```

해결문제

- 부산대역사.txt에서 부산대학교가 몇 번 사용되었는지 알아보세요.
  - 단, 부산대는 부산대학교로 처리

튜플(tuple)

- 변경할 수 없는 순서 있는 객체의 집합
  - 리스트와 비슷하지만 한번 생성되면 변경할 수 없음
- ()를 사용하여 정의
- 자료추가

```
t = (1,2,3)

'''
t[3] = 4
t.append(4)
t.extend((4))
'''


t = t + (4,)
print(t)
```

딕션너리(dictionary)

- Key와 Value라는 것을 한 쌍으로 갖는 자료형
- 리스트나 튜플처럼 순차적으로(sequential) 해당 요소값을 구하지 않고 Key를 통해 Value를 얻음
- Key는 고유한 값이므로 중복되는 Key 값을 설정해 놓으면 하나를 제외한 나머지 것들이 모두 무시
- {Key1:Value1, Key2:Value2, Key3:Value3, ...}

```
dic = {}

#값 추가
dic['a'] = 1
dic['b'] = 2

print(dic)
#해당 키값 가져오기r
print(dic['a'])

#키 가져오기
print(dic.keys())

#키값 가져오기
print(dic.values())

#키와 값 쌍으로 가져오기
print(dic.items())

#전체 지우기
dic.clear()
print(dic)
```

```
{'a': 1, 'b': 2}
1
dict_keys(['a', 'b'])
dict_values([1, 2])
dict_items([('a', 1), ('b', 2)])
{}
```

파이썬 내장함수

- **sum(), max(), min()**
  - 입력으로 받은 리스트나 튜플의 모든 요소의 합,최대, 최소값을 리턴하는 함수
- **zip()**
  - 동일한 개수로 이루어진 자료형을 묶어주는 역할을 하는 함수
  - Dictionary의 **{key: value}** 중 value 값으로 최대값과 최소값, 그리고 정렬을 하는데 사용

```
list1 = ['a', 'b','c']
list2 = [1, 3, 2]

#합 계
print(sum(list2))
#최 대
print(max(list2))
#최 소 값
print(min(list2))
#정 렬 (
print(sorted(list2))

#zip으 로  리 스 트  만 들 기
list3 = list(zip(list1, list2))
print(list3)

#zip으 로  딕 션 너 리  만 들 기 i
dict1 = dict(zip(list1, list2))
print(dict1)

#zip으 로  최 대 값 /최 소 값  찾 기 )
print(max(zip(dict1.values(), dict1.keys())))
print(min(zip(dict1.values(), dict1.keys())))

#zip으 로 정 렬 (
print(sorted(zip(dict1.values(), dict1.keys())))
```

```
6
3
1
[1, 2, 3]
[('a', 1), ('b', 3), ('c', 2)]
{'a': 1, 'b': 3, 'c': 2}
(3, 'b')
(1, 'a')
[(1, 'a'), (2, 'c'), (3, 'b')]
```

해결문제

- 도별미세먼지.csv 파일을 읽어서 딕션너리를 만들고 평균과 최대값을 가지는 도를 찾으시오.