4. 컬렉션

1. 리스트

```
a =[0,0,0,0]                        #리스트
b=[]                                 #리스트
print(b,len(b))
suma,sumb=0,0 #변수의 초기화
for i in range(0,len(a)) :  #i : 0 ~ 3
# str(int) : 정수형을 문자열형으로 변환 함수
    a[i]=int(input(str(i+1) + "번째 값:"))
    suma += a[i]
    b.append(a[i])                  #리스트 추가
    sumb += b[i]
print(a)
print(b,len(b))
print("a 리스트 합계:",suma)
print("b 리스트 합계:",sumb)
```

1. 리스트

```
mylist = [30,10,20]
# %s : 문자열을 지정하는 형식지정문자
print("리스트:%s" % mylist)
mylist.append(40)
print("mylist.append(40) 리스트:%s" % mylist)
#pop : LIFO(stack) 관련 함수.
print("pop() 메서드 결과:%s" % mylist.pop())
print("pop() 메서드 후 리스트 :%s" % mylist)

mylist.sort()
print("mylist.sort() 후 리스트 : %s" % mylist)
mylist.reverse()  #역순 재배치
print("mylist.reverse() 후 리스트 : %s" % mylist)
```

1. 리스트

```
print("20값의 위치 : %d" % mylist.index(20)) #20값의 인덱스값 리턴
mylist.insert(2,222)
print("mylist.insert(2,222) 후 리스트 : %s" % mylist)
mylist.remove(222)  #222 값을 제
print("mylist.remove(222) 후 리스트 : %s" % mylist)
mylist.extend([77,77,99])  #리스트 추가
print("mylist.extend([77,77,99]) 후 리스트 : %s" % mylist)
print("77값의 갯수 : %d" % mylist.count(77)) # 77요소의 갯수 리턴
#mylist의 요소 중 200값이 없다. 실행 오류 발생
#print("200값의 위치 : %d" % mylist.index(200)) #200값의 인덱스값 리턴
# list 객체에는 find 메서드 없다. 오류 발생
#print("20값의 위치 : %d" % mylist.find(20)) #20값의 인덱스값 리턴
```

1. 리스트

```
ss = "2021/02/05"
print("10년전 년도 출력하기")
list = ss.split("/")
print("10년전 :",int(list[0])-10 )
ss = "(홍길동)"  #   홍#길#동# 출력하기
for i in range(1,len(ss)-1) :
    print(ss[i],end="#")
print()
# for문 사용 없이  동길홍 출력하기
print(ss[len(ss)-2:0:-1])
# ss 문자열이 (시작하지 않으면 ( 추가하기  ) 끝나지 않으면  ) 추가하기
ss = "(홍길동)"
if ss.startswith("(") == False :
    print("(", end="")
print(ss,end="")
if ss.endswith(")") == False :
    print(")")
```

2.튜플

```
tp1=(10,20,30)
print(tp1)
#tp1.append(40)  #tuple은 변경 할 수 없음
list1 = list(tp1) #tuple을 list로 변경
list1.append(40)  #list 객체에 요소 추가
tp1 = tuple(list1) # list를 tuple로 변
print(tp1)
print("tp1의 크기=",len(tp1))
print("tp[1:3]=",tp1[1:3])
print("tp[:3]=",tp1[:3])
print("tp[2:]=",tp1[2:])
print("tp[::2]=",tp1[::2])
print(tp1[0],tp1[1],tp1[2])  #인덱스를 이용하여 접근 가능

a,b,c,d=tp1   #tuple의 각 요소를 각각의 변수에 저장
print(a,b,c,d)
```

3. 딕셔너리

```
singer = {}  #dictionary 객체로 초기
#       키       value
singer['이름']='트와이스'  #{"이름":"트와이스"}
singer['구성원수']=9     #{"이름":"트와이스","구성원수":9}
singer['데뷔곡']='우아하게'
singer['소속사']='JYP'
# i : key
for i in singer.keys() :  #키들만 조
    print("%s=>%s" % (i,singer[i]))
print("singer 자료형:",type(singer))
print("singer:",singer)
print("singer.keys() 자료형:",type(singer.keys()))
print("singer.keys():",singer.keys())
print(list(singer.keys()))
```

3. 딕셔너리

화면에서 등록된 음식을 입력받아서 궁합음식을 출력하기

```
foods = {"떡볶이":"오뎅","짜장면":"단무지","라면":"김치","맥주":"치킨"}
for i in foods.keys() :
   print("%s=>%s" % (i,foods[i]))
while True :
   myfood = input(str(list(foods.keys())) + "중 음식이름 입력하세요:")
   if myfood == "종료" :
      print("등록된 음식:",list(foods.keys()))  #map.keySet()
      print("등록된 궁합음식:",list(foods.values()))  #map.values()
      print("등록된 음식,궁합음식:",list(foods.items()))  #map.entrySet()
      break
   if myfood in foods : #myfood값이 foods의 key에 존재?
     print("<%s> 궁합음식은 <%s>입니다." % (myfood,foods[myfood]))
   else :
     print("등록된 음식이 아닙니다.")
     yn = input("좋아하는 음식으로 등록하시겠습니까(y)")
     if yn =='Y' or yn=='y' :
       f = input("궁합음식을 등록하세요")
       foods[myfood] = f
```

3. 딕셔너리

정렬하기

```
import operator
dic,list = {},[]
dic = {"Thomas":"토마스","Edward":"에드워드","Henry":"헨리",  "Gothen":"고든","James":"제임스"}
print(dic.items())
# sorted : 정렬 함수
#dic.items() : (키,값)쌍인 객체. tuple객체들의 리스트. 정렬을 위한 데이터값
#operator.itemgetter(0) : 정렬의 기준 객체 설정 .
#    0 : 튜플의 첫번째 객체로 설정
#   reverse=True : 내림차순 정렬, 기본값 : False
list = sorted(dic.items(), key=operator.itemgetter(0),reverse=True)
print(list)
print("영문이름으로 정렬하기========= ")
list = sorted(dic.items(), key=operator.itemgetter(0),reverse=False)
print(list)
print("한글 이름으로 정렬하기========= ")
list = sorted(dic.items(), key=operator.itemgetter(1))
print(list)
print("한글 이름으로 역순 정렬하기========= ")
list = sorted(dic.items(), key=operator.itemgetter(1),reverse=True)
```

4. Set

집합

```
set1 = {1,2,3,4,5}
print(set1)
set2 = {1,2,3,4,5,1,2,3,4,5}
print(set2)
set3 = {5,6,7,8}
print(set3)

#교집합
print("set1과 set3의 교집합",set1 & set3)
print("set1과 set3의 교집합",set1.intersection(set3))
#합집합
print("set1과 set3의 합집합",set1 | set3)
print("set1과 set3의 합집합",set1.union(set3))
```

5. 컴프리헨션 (Comprehension)

패턴이 있는 list, dictionary, set을 간편하게 작성

```
numbers = []
for n in range(1,11) :
    numbers.append(n)
print(numbers)
#컴프리헨션 표현
print([x for x in range(1,11)])
clist = [x for x in range(1,11)]
print(clist)
```

5. 컴프리헨션 (Comprehension)

1 ~ 10까지의 짝수 리스트 작성하기. for 구문으로 작성하기

```
evenlist=[]
#for n in range(2,11,2) :
#    evenlist.append(n)
for n in range(2,11) :
    if(n%2==0) :
        evenlist.append(n)
print(evenlist)
#컴프리헨션 표현
evenlist=[x for x in range(1,11) if x%2==0]
print(evenlist)
```

5. 컴프리헨션 (Comprehension)

2의 배수이고, 3의 배수인 값만 리스트에 추가하기.

```
list23=[x for x in range(1,11) if x%2==0 if x%3==0]
print(list23)
```

중첩사용 컴프리헨션

```
matrix = [[1,2,3],[4,5,6],[7,8,9]]
print(matrix)
# row : [1,2,3]
# x : 1
list1 = [x for row in matrix for x in row]
print(list1)
# 컴프리헨션 표현식을 사용하지 않음
list1= []
for row in matrix:
    for x in row :
        list1.append(x)
print(list1)
```

5. 컴프리헨션 (Comprehension)

```
colorlist=['black','white']
sizelist=["S","M","L"]
dresslist=((c,s) for c in colorlist for s in sizelist)
print(dresslist)
for d in dresslist :
    print(d)
```

5. 컴프리헨션 (Comprehension)

컴프리헨션을 이용한 set

```
set1 = {x**2 for x in [1,1,2,2,3,3]}
print(set1) #{1,4,9}

list1 = [x**2 for x in [1,1,2,2,3,3]]
print(list1)

# 1부터 10까지의 수중 짝수의 제곱을 출력하기
set2 = { x**2 for x in range(1,11) if x % 2 == 0}
print(set2)
print(sorted(set2)) #리스트
```

5. 컴프리헨션 (Comprehension)

dictionary에서 컴프리헨션 사용하기

```
products = {"냉장고":220,"건조기":140,"TV":130,"세탁기":150, "오디오":50,"컴퓨터":250}
print(products)
product1 = {}
#200만원 미만의 제품만 product1에 저장하기
'''
for name in products.keys() :
    if products.get(name) < 200 :
        product1[name]=products.get(name)
'''
for p,v in products.items() : #(k,v) 쌍인 객체들
   if v < 200 :
      product1.update({p:v})
print(product1)
#컴퓨리헨션으로 구현하기
product2 = {p:v for p,v in products.items() if v < 200}
print(product2)
```

6. map

리스트의 각각의 요소를 변경함

```
before = ["2021","02","08"]
print(type(before[0])) #문자열
print(before)
#before의 요소 각각을 int 형변환한 후 다시 list 객체로 생성
after = list(map(int,before))
print(type(after[0])) #int
print(after)
print(int("123"))
```

6. map

람다를 이용하여 map 처리 하기

```
mylist = [1,2,3,4,5]
add = lambda num:num+10
mylist = list(map(add,mylist))
print(mylist)

#num : mylist의 요소값 한개.
mylist = list(map(lambda num:num-10,mylist))
print(mylist) #[1,2,3,4,5]
mylist = list(map(lambda num:num*10,mylist))
print(mylist)
```

6. map

람다를 이용하여 map 처리 하기

```
list1=[1,2,3,4]
list2=[10,20,30,40]
hap=lambda n1,n2:n1+n2
haplist=list(map(hap,list1,list2))
print(haplist)

# haplist = mylist + list1 + list2
# 리스트 중 최소요소갯수의 리스트의 갯수로 맞춤.
list1.append(0)
list2.append(0)
haplist = list(map(lambda n1,n2,n3:n1+n2+n3,mylist,list1,list2))
print(haplist)
```