

## [8장 과제]

맑은 고딕 12 ☞ 학번 : 20203103 이름 : 임정민

실습 8-1 (p. 201 도전문제 8-1) 여러 분의 핸드폰에 있는 다음과 같은 연락처 정보를 contact라는 이름의 딕셔너리 구조로 표현해 보자. 여러 분의 정보를 그림과 같은 키 값을 가지도록 항목을 생성하고, for문을 사용해 출력하시오.

### [결과 화면]

```
python practice/ex8-1.py
8-1. 20203103 임정민
FirstName : 임
LastName : 정민
work : 동의대학교
```

### [프로그램 소스]

```
##8.1.py
print("8-1. 20203103 임정민")
contact= {'FirstName': '임', 'LastName': '정민', 'work': '동의대학교'}
for i,j in contact.items():
    print(i, ' : ', j)
```

실습 8-2 (p.204, 도전문제8.2) 다음 결과 화면처럼 편의점의 재고를 관리하는 프로그램을 작성하시오. 재고를 증가 또는 감소시킬 수 있도록 코드를 추가하여 보자. 재고조회, 입고, 출고와 같은 간단한 메뉴도 만들어 보자. 강의 동영상 참고할 것.

### [결과 화면]

```
python practice/ex8-2.py
메뉴를 선택하세요 1) 재고조회, 2) 입고, 3)출고, 4) 종료 : 1
[재고조회] 물건의 이름을 입력하시오 : 펜
재고 : 3
메뉴를 선택하세요 1) 재고조회, 2) 입고, 3)출고, 4) 종료 : 2
[입고] 물건의 이름과 수량을 입력하시오 : 펜 2
재고 : 5
메뉴를 선택하세요 1) 재고조회, 2) 입고, 3)출고, 4) 종료 : 3
[출고] 물건의 이름과 수량을 입력하시오 : 펜 1
재고 : 4
메뉴를 선택하세요 1) 재고조회, 2) 입고, 3)출고, 4) 종료 : 4
```

### [프로그램 소스]

```
##ex 8-2.py

items = {"커피음료": 7, "펜": 3, "종이컵":2, "우유": 1,}

#name = input("물건의 이름을 입력하시오:")
#print('재고 :', items[name])

w = 0
while w!= 1:
    m = int(input("메뉴를 선택하세요 1) 재고조회, 2) 입고, 3)출고, 4) 종료: "))
    if m == 1 :
        n = input("[재고조회] 물건의 이름을 입력하시오 : ")
        print('재고:', items[n])
    elif m == 2:
        n , count = input("[입고] 물건의 이름과 수량을 입력하시오 : ").split()
        items[n] = items[n] + int(count)
```

```

    print('재고:', items[n])
elif m == 3:
    n, count = input("[출고] 물건의 이름과 수량을 입력하시오 : ").split()
    items[n] = items[n] - int(count)
    print('재고:', items[n])
elif m == 4:
    w = 1

```

실습 8-3 (p.205 LAB8-2 응용) 결과 화면과 같이 영어 단어를 “영어:한국어” 쌍으로 입력하면 사전에 저장하고, 영어 단어를 치면 해당 한국어를 출력하는 프로그램을 작성하시오. p를 치면 모든 단어가 출력되도록 하시오.

### [결과 화면]

```

$ q
사전 프로그램을 종료합니다.
~/Doc/co/p/test1 python main.py
사전 프로그램 시작.. 종료는 q를 입력
$ < one:하나
$ < two:둘
$ < car:자동차
$ > car
자동차
$ p
dict_items([('one', '하나'), ('two', '둘'), ('car', '자동차')])
$ q
사전 프로그램을 종료합니다.

```

### [프로그램 소스]

##ex8-3.py

```
print("사전 프로그램 시작.. 종료는 q 를 입력")
```

```
dictionary = {}
```

```
while True:
```

```
    st = input('$ ') ## st 에 입력받기
```

```
    command = st[0]      ## '<' '>' 명령어 구분을 위해 command 에 첫문자를 저장한다.
```

```
    if command == '<': ## 입력 명령문
```

```
        st = st[1:]      ## 첫문자 '<' '>' 를 제외한 문자열을 저장
```

```
        inputStr = st.split(':') ## split 함수로 : 기준으로 두 문자열로 분리해 딕셔너리로
```

저장

```
        if len(inputStr) < 2 :
```

```
            print('입력 오류가 발생했습니다.')
```

```
        else:
```

```
            dictionary[inputStr[0].strip()] = inputStr[1].strip()
```

```
    elif command == '>': ##검색 명령문
```

```
        st = st[1:] ## 첫문자를 제외한 문자열을 저장한다.
```

```
        inputStr = st.strip() ##검색할 단어 문자열의 공백을 제거한다.
```

```
        if inputStr in dictionary: ## 만약 인풋이 딕셔너리에 있으면
```

```
            print(dictionary[inputStr]) ## 해당단어 뜻 출력
```

```
        else :
```

```
            print('{}가 사전에 없습니다. '.format(inputStr)) ## 없으면 키값출력
```

```

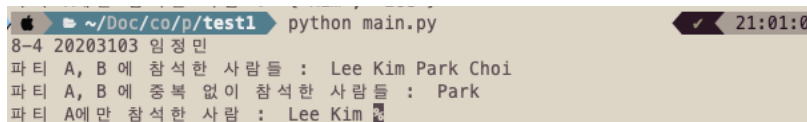
elif command == 'p':    ##모든 단어 출력문
    print(dictionary.items())
elif command == 'q': ## q 가 입력되면 종료
    break
else :
    print('입력 오류가 발생했습니다.')
print ("사전 프로그램을 종료합니다.")

```

실습 8-4 (p.210 도전 8-3) 다음은 partyA에 참석한 사람과 B에 참석한 사람의 집합이다. 집합 연산자나 메소드를 사용하여 다음의 [결과 화면]과 같이 출력되도록 프로그램을 완성하시오.

```
partyA = set(["Park", "Kim", "Lee"]), partyB = set(["Park", "Choi"])
```

#### [결과 화면]



```

8-4 20203103 임정민
파티 A, B 에 참석한 사람들 : Lee Kim Park Choi
파티 A, B 에 중복 없이 참석한 사람들 : Park
파티 A에만 참석한 사람 : Lee Kim

```

#### [프로그램 소스]

#ex8-4.py

```
print("8-4 20203103 임정민")
```

```
partyA = set(["Park", "Kim", "Lee"])
partyB = set(["Park", "Choi"])
```

```
print("파티 A, B 에 참석한 사람들 : ",end = ' ')
for i in partyA.union(partyB):
    print(i,end=' ')

```

```
print("\n 파티 A, B 에 중복 없이 참석한 사람들 : ",end = ' ')
for j in partyA.intersection(partyB):
    print(j,end = ' ')

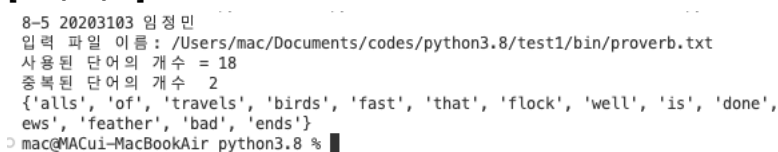
```

```
print("\n 파티 A에만 참석한 사람 : ", end = ' ')
for k in partyA.difference(partyB):
    print(k,end = ' ')

```

실습 8-5 (p.212 LAB8-4 응용) [결과 화면]처럼 중복된 단어의 개수를 추가하여 출력하는 프로그램을 작성하시오.

#### [결과 화면]



```

8-5 20203103 임정민
입력 파일 이름: /Users/mac/Documents/codes/python3.8/test1/bin/proverb.txt
사용된 단어의 개수 = 18
중복된 단어의 개수 2
{'alls', 'of', 'travels', 'birds', 'fast', 'that', 'flock', 'well', 'is', 'done',
'ews', 'feather', 'bad', 'ends'}
mac@MACui-MacBookAir python3.8 %

```

#### [프로그램 소스]

#ex8-5.py

```

print("8-5 20203103 임정민")

##단어에서 구두점을 제거하고 소문자로 만든다.
def process(w):
    output = ""
    for ch in w:
        if ch.isalpha():
            output += ch
    return output.lower()

words = set() # 중복을 방지하기 위해 집합 자료형에 단어를 넣자

fname = input("입력 파일 이름: ")
file = open(fname, "r") # 파일을 연다.

# 파일의 모든 줄에 대하여 반복한다.
count = int(0)
for line in file:
    lineWords = line.split()
    for word in lineWords:
        tmp = process(word)
        if tmp in words: ## 만약 tmp 가 이미 words 셋에 있다면 count ++
            count = count +1
        words.add(tmp)

# 단어를 집합에 추가한다.

print("사용된 단어의 개수 =", len(words))
print("중복된 단어의 개수 ",count )
print(words)

```

실습 8-6 (p.215, 심화문제 8-1, 8-2 종합) 심화문제 8-1, 8-2의 내용을 종합하여 다음과 같이 [결과 화면]처럼 출력결과가 출력되도록 프로그램을 완성하시오.

#### [결과 화면]

```

mac@MACui-MacBookAir python3.8 % python -u "/Users/mac/Documents/codes/python3.8/test1/main.py"
8-6 20203103 임정민
사과, 배, 수박, 귤, 포도 가격을 공백으로 구분하여 입력:100 200 300 400 500
-----오늘의 과일 가격-----
사과 : 100 원
배 : 200 원
수박 : 300 원
귤 : 400 원
포도 : 500 원
구매를 원하시는 과일의 이름을 입력하시오 (종료 q):귤
오늘의 귤 가격은 400원 입니다.
구매를 원하시는 과일의 이름을 입력하시오 (종료 q):q
mac@MACui-MacBookAir python3.8 %

```

#### [프로그램 소스]

```
##ex 8-6.py
```

```

print("8-6 20203103 임정민")

fruits_dic = {'사과':0, '배': 0, '수박':0, '귤': 0, '포도':0}
fruits_list = ['사과', '배', '수박', '귤', '포도']
frStr = input ("사과, 배, 수박, 귤, 포도 가격을 공백으로 구분하여 입력:",)

frStr_split = frStr.split(' ')

for i in range(len(fruits_dic)):
    fruits_dic[str(fruits_list[i])] = frStr_split[i]

print("-----오늘의과일가격-----")
for m in fruits_dic.items():
    print("{0} : {1} 원".format(m[0],m[1]))

while True:
    st = input("구매를 원하시는 과일의 이름을 입력하시오 (종료 q) :")
    if st == 'q':
        break
    print("오늘의 {0} 가격은 {1}원 입니다.".format(st,fruits_dic[st]))

```

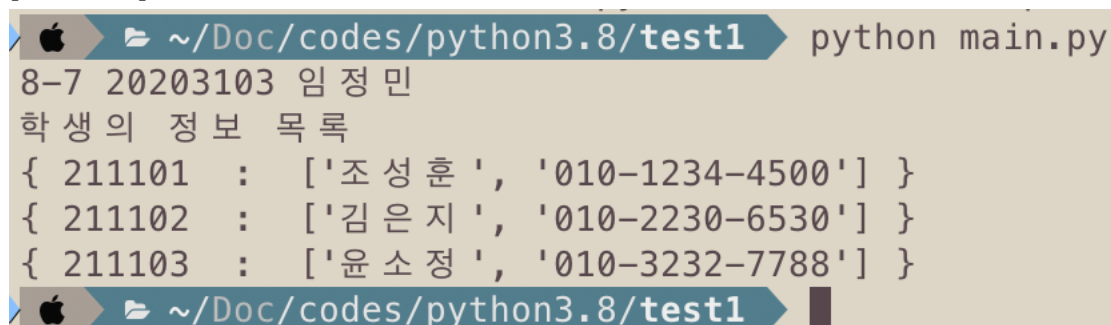
실습 8-7 (p.215, 심화문제 8-3 1).번) 학번, 이름, 전화번호의 3쌍의 요소를 가지는 student\_tup라는 튜플이 다음과 같이 존재한다. 이 튜플을 수정하여 {학번 : [이름, 전화번호]} 쌍으로 이루어진 딕셔너리를 만들어서 출력하시오.

```

student_tup = (('211101', '최성훈', '010-1234-4500'), ('211102', '김은지', '010-2230-6540'), ('211103', '이세은', '010-3232-7788'))

```

#### [결과 화면]



```

~/Doc/codes/python3.8/test1 python main.py
8-7 20203103 임 정 민
학 생 의 정 보 목 록
{ 211101 : ['조 성 훈 ', '010-1234-4500'] }
{ 211102 : ['김 은 지 ', '010-2230-6530'] }
{ 211103 : ['윤 소 정 ', '010-3232-7788'] }

```

#### [프로그램 소스]

```

##ex 8-7.py
print("8-7 20203103 임정민")

```

```
student_tup = (('211101','조성훈','010-1234-4500'), ('211102', '김은지', '010-2230-6530'),('211103','윤소정','010-3232-7788'))
```

```
std_dict = {} ## 빈 딕셔너리 생성
```

```
c = int(0)
```

```
for i in student_tup:
```

```
    tmp = '[W]' + i[1] + '[W, W]' + i[2] + '[W]'
```

```
    ##tmp = "[{0}', '{1}']".format(i[1], i[2]) ##밸류로 사용할 문자열 생성
```

```
    std_dict[i[0]] = tmp ##{ 학번 : [이름, 전화번호]}
```

```
print("학생의 정보 목록")
```

```
for i in std_dict.items():
```

```
    print("{},i[0]," : ", i[1],")" ## items로 호출해 각 index를 출력
```

실습 8-8 (p.215, 심화문제 8-3 2)번) 학생의 학번을 입력으로 받아서 이름과 전화번호를 출력하는 프로그램을 작성하시오.

#### [결과 화면]

```
~/Doc/co/p/test1 ➤ python main.py
8-8 20203103 임정민
학번을 입력하시오 : 211101
이름 : 조성훈
연락처 : 010-1234-4500
~/Doc/co/p/test1 ➤
```

#### [프로그램 소스]

```
##ex 8-8.py
```

```
print("8-8 20203103 임정민")
```

```
student_tup = (('211101','조성훈','010-1234-4500'), ('211102', '김은지', '010-2230-6530'),('211103','윤소정','010-3232-7788'))
```

```
std_dict = {} ## 빈 딕셔너리 생성
```

```
c = int(0)
```

```
for i in student_tup:
```

```
    tmp = '[W]' + i[1] + '[W, W]' + i[2] + '[W]'
```

```
    ##tmp = "[{0}', '{1}']".format(i[1], i[2]) ##밸류로 사용할 문자열 생성
```

```
std_dict[i[0]] = tmp ##{ 학번 : [이름, 전화번호]}
```

```
searchingID = input("학번을 입력하시오 : ")
```

```
resultString = std_dict[searchingID].split(',') ##rs
```

```
resultName = resultString[0]
```

```
resultTel = resultString[1]
```

```
resultName = resultName.strip(' ')
```

```
resultName = resultName.strip('"')
```

```
resultName = resultName.strip(" ")
```

```
resultTel = resultTel.strip(' ')
```

```
resultTel = resultTel.strip('[]')
```

```
resultTel = resultTel.strip('"')
```

```
resultTel = resultTel.strip(' ')
```

```
print("이름 :", resultName)
```

```
print("연락처 :", resultTel)
```

실습 8-9 (p.216, 심화문제 8-3. 3)번, 4)번 종합) 세 학생의 학점 추가, 세 학생의 평균 계산

**[결과 화면]**

```
python main.py
8-9 20203103 임정민
{'211101' : ['조성훈', '010-1234-4500', 4.3]}
{'211102' : ['김은지', '010-2230-6530', 3.9]}
{'211103' : ['윤소정', '010-3232-7788', 4.25]}
전체 학생의 학점 평균 : 4.15
```

**[프로그램 소스]**

```
##ex 8-9.py
```

```
print("8-9 20203103 임정민")
```

```
student_tup = (('211101','조성훈','010-1234-4500', '4.3'), ('211102', '김은지', '010-2230-6530','3.9'),('211103','윤소정','010-3232-7788','4.25'))
```

```
std_dict = {} ## 빈 딕셔너리 생성
```

```
def cleanStr(s): ##문자열에서 특수기호 제거
```

```
    s = s.strip()
```

```
    s = s.strip("[")
```

```
    s = s.strip("]")
```

```
    s = s.strip('W')
```

```
    s = s.strip('')
```

```
    return s
```

```
for i in student_tup:
```

```
    ##tmp = '[W' + i[1] + 'W', W' + i[2] + 'W',
```

```
    tmp = "[{0}', '{1}', '{2}']".format(i[1], i[2], i[3]) ##밸류로 사용할 문자열 생성
```

```
    std_dict[i[0]] = tmp ##{ 학번 : [이름, 전화번호, 학점]}
```

```
resultName = []
```

```
resultTel = []
```

```
resultScore = []
```

```
resultID = []
```

```
c = int(0)
```

```
des = int(len(std_dict))
```

```
for i in std_dict:
```

```
    resultString = std_dict[i].split(',')
```

```
    resultID.append(i)
```

```
    resultName.append(cleanStr(resultString[0]))
```

```
    resultTel.append( cleanStr(resultString[1]))
```

```
    resultScore.append( cleanStr(resultString[2]))
```

```
sum = float(0)
```

```
avg = float(0)
```

```
for i in range(c,des):
```

```
    sum = sum + float(resultScore[i]) ## 합계
```

```
avg = sum / float(des) ## 평균 저장
```

```
for i in range(c,des):
```

```
    print('{ ' + "W'{0}W' : [W'{1}W', W'{2}W', {3}]" .format(resultID[i],resultName[i],resultTel[i],resultScore[i]) +
```



```
    })
print("전체 학생의 학점 평균 : ", round(avg, 2))
```

실습 8-10 (p.216, 심화문제 8-6) 튜플을 요소로 가지는 student\_tuple 리스트가 다음과 같이 있다. 이 튜플의 요소가 되는 튜플은 다음과 같이 (학번, 이름, 전화번호)로 이루어져 있다. 이를 이용하여 다음 [결과 화면]과 같이 결과가 출력되도록 프로그램을 완성하시오.

```
student_tuple : [('211101', '강이안', '010-123-1111'), ('211102', '박동민', '010-123-2222'), ('211103', '김수정', '010-123-3333')]
```

#### [결과 화면]

```
python main.py
8-10 20203103 임정민

1. student_tuple 리스트 출력
(('211101', '조성훈', '010-1234-4500'), ('211102', '김은지', '010-2230-6530'), ('211103', '윤소정', '010-3232-7788'))

2. (학번 이름)의 딕셔너리 출력
211101 : 조성훈
211102 : 김은지
211103 : 윤소정

3. 학번으로 조회 결과 출력
학번을 입력하시오 211101
211101 학생은 조성훈이며, 전화번호는 010-1234-4500 입니다.
```

#### [프로그램 소스]

```
##ex 8-10.py
print("8-10 20203103 임정민\n")

student_tup = (('211101', '조성훈', '010-1234-4500'), ('211102', '김은지', '010-2230-6530'), ('211103', '윤소정', '010-3232-7788'))
std_dict = {} ## 빈 딕셔너리 생성

print("1. student_tuple 리스트 출력")

print(student_tup)
def cleanStr(s): ##문자열에서 특수기호 제거
    s = s.strip()
    s = s.strip("[")
    s = s.strip("]")
    s = s.strip('\')
    s = s.strip('')
    s = s.strip(",")
    s = s.strip("\'")
    return s

for i in student_tup:
    ##tmp = '[' + i[1] + '\', ' + i[2] + '\', '
    tmp = "[" + i[1] + "," + i[2] + "]\n" ##백류로 사용할 문자열 생성
    std_dict[i[0]] = tmp ##{ 학번 : [이름, 전화번호, 학점]}
```

```
resultName = []
resultTel = []
resultID = []

c = int(0)
des = int(len(std_dict))
for i in std_dict:
    resultString = std_dict[i].split(',')
    resultID.append(i)
    resultName.append(cleanStr(resultString[0]))
    resultTel.append(cleanStr(resultString[1]))

print("\n2. (학번 이름)의 딕셔너리 출력")
for i in range(des):
    print(resultID[i], " : " ,resultName[i])

print("\n3. 학번으로 조회 결과 출력")
searching = input("학번을 입력하시오 ")

seResult = std_dict[searching].split()
seResultName = cleanStr(seResult[0])
seResultTel = cleanStr(seResult[1])

print("{} 학생은 {} 이며, 전화번호는 {} 입니다.".format(searching,seResultName,seResultTel))
```