7주차 실습

리스트, 튜플, 세트, 딕셔너리 실습

2019. 10. 14.

오늘의 실습

- 다음 각 자료형의 실습코드를 실행해 볼 것.
  - 리스트 실습 소스코드 – 1 ~ 7 (8쪽 ~ 14쪽)
  - 튜플 실습 소스코드 – 1 ~ 2 (17쪽 ~ 18쪽)
  - 세트 실습 소스코드 – 1 ~ 3 (22쪽 ~ 24쪽)
  - 딕셔너리 실습 소스코드 – 1 ~7 (28쪽 ~ 35쪽)

- 제출 : 10. 21.(월) 자정까지 (지각 제출 없음)

- 제출물 : 아래아한글 파일 (또는 pdf 파일) 보고서 1개로 제출
  - 소스코드는 실행해 볼 수 있도록 표 안에 붙여넣기로 제출
  - 실행결과는 소스코드와 실행결과를 볼 수 있게 화면을 capture하여 제출

리스트, 튜플, 세트, 딕셔너리

- 자료형 별 구분
  - 리스트 : 변경가능한 시퀀스 자료형
  - 튜플 : 변경불가능한 시퀀스 자료형
  - 세트 : 집합(set) 자료형
  - 딕셔너리 : 매핑(mapping) 자료형

```
>>> aList = []
>>> type(aList)
<class 'list'>
>>> aTuple = ()
>>> type(aTuple)
<class 'tuple'>
>>> aSet = set()
>>> type(aSet)
<class 'set'>
>>> aDic = {}
>>> type(aDic)
<class 'dict'>
```

- 생성 방법
  - 리스트 = [ ]
  - 튜플 = ( )
  - 세트 = { }
  - 딕셔너리 = { : }

singleton tuple만들 때 aTuple = (1,)

공백 set 만들 때 aSet1 = set()

리스트와 튜플

- 리스트(list)
  - 순서가 있는 일련의 값들
  - 인덱스를 이용하여 각 값을 구분함
  - 리스트를 이용하여 stack이나 queue 자료구조를 구현가능
    - 스택 : 리스트.append()와 리스트.pop() 이용
    - 큐 : 리스트.append()와 리스트.popleft() 이용
- 튜플
  - 변경불가능한 자료형
    - 변경불가능한 자료형을 사용하는 잇점 :
      - 생성과정이 간단 (왜냐하면, 데이터를 할당하는 공간의 내용이나 크기가 달라지지 않기 때문)
      - 데이터가 변경될 가능성이 없음
  - (문자열과 비슷하게) 일련의 요소들이 순서대로 나열된 것
  - 일련의 요소들의 데이터형이 혼합되어도 무방
  - 표현 : 괄호 안에 쉼표로 나누어진 리스트를 써서 표현
- 시퀀스 자료형
  - 리스트(list), 튜플(tuple), 문자열(str)

세트와 딕셔너리

- 세트(Set)
  - 순서와 중복이 없이 저장된 자료형
  - { }을 이용하여 생성

- 딕셔너리(Dictionary)
  - Key – Value 쌍으로 이루어진 mapping 자료형
  - { : }를 이용하여 생성, 첫번째 요소 key, 두번째 요소 value
    - key는 변경이 불가능
    - value는 변경 가능
  - key : 변수 또는 튜플
  - value : 거의 모든 자료형 가능

1. 리스트

리스트의 메소드

메소드	설명
append()	리스트의 끝에 새 요소를 추가
count()	매개변수로 입력한 데이터와 일치하는 요소가 몇 개 있는지 카운트함
len(리스트)	리스트에 포함된 전체 요소의 개수를 셈
extend()	기존 리스트에 다른 리스트를 이어 붙임
index()	리스트 내에서 매개변수로 입력한 데이터와 일치하는 첫 번째 요소의 첨자를 알려줌
insert()	첨자로 명시한 리스트 내의 위치에 새 요소를 삽입
pop()	리스트의 마지막 요소를 뽑아내어 리스트에서 제거
remove()	매개변수로 입력한 데이터를 리스트에서 찾아 발견한 첫 번째 요소만을 제거
reverse()	리스트 내 요소의 순서를 반대로 뒤집음
sort()	리스트 내의 요소를 정렬함 (매개변수로 reverse = True를 입력하면, 내림차순)
sorted(리스트)	정렬된 리스트를 출력하는 메소드
기타 메소드	https://docs.python.org/3/tutorial/datastructures.html

리스트 실습 소스코드 - 1

```
print("₩n1. 샘플 리스트 출력")
sampleList = [3, 4, 1]
print("    sampleList = ", sampleList)

print("₩n2. 샘플 리스트에 한 요소를 추가")
print("    sampleList = ", sampleList)
sampleList.append(5)
print("    sampleList.append(5) 후의 sampleList = ",sampleList)

print("₩n3. 샘플 리스트의 제일 마지막 요소의 제거(pop)")
print("    sampleList = ", sampleList)
print("    샘플 리스트의 갯수 : len(sampleList) = ", len(sampleList))
print("    제거된 마지막 요소 = ", sampleList.pop())
print("    sampleList.pop() 후의 sampleList = ", sampleList)
print("    샘플 리스트의 갯수 : len(sampleList) = ", len(sampleList))
```

```
1. 샘플 리스트 출력
   sampleList =  [3, 4, 1]

2. 샘플 리스트에 한 요소를 추가
   sampleList =  [3, 4, 1]
   sampleList.append(5) 후의 sampleList =  [3, 4, 1, 5]

3. 샘플 리스트의 제일 마지막 요소의 제거(pop)
   sampleList =  [3, 4, 1, 5]
   샘플 리스트의 갯수 : len(sampleList) =  4
   제거된 마지막 요소 =  5
   sampleList.pop() 후의 sampleList =  [3, 4, 1]
   샘플 리스트의 갯수 : len(sampleList) =  3
```

리스트 실습 소스코드 - 2

```
4. 샘플 리스트의 정렬 3가지 방법
  4-1. 거꾸로 정렬 (reverse)
       sampleList =  [3, 4, 1]
       sampleList.reverse() 후의 sampleList =  [1, 4, 3]
  4-2. 오름차순으로 정렬
       sampleList =  [1, 4, 3]
       sampleList.sort() 후의 sampleList =  [1, 3, 4]
  4-3. 내림차순으로 정렬
       sampleList =  [1, 3, 4]
       sampleList.sort(reverse = True) 후의 sampleList =  [4, 3, 1]
```

```
print("₩n4. 샘플 리스트의 정렬 3가지 방법")
print("  4-1. 거꾸로 정렬 (reverse)")
print("       sampleList = ", sampleList)
sampleList.reverse()
print("       sampleList.reverse() 후의 sampleList = ", sampleList)

print("  4-2. 오름차순으로 정렬")
print("       sampleList = ", sampleList)
sampleList.sort()
print("       sampleList.sort() 후의 sampleList = ", sampleList)

print("  4-3. 내림차순으로 정렬")
print("       sampleList = ", sampleList)
sampleList.sort(reverse = True)
print("       sampleList.sort(reverse = True) 후의 sampleList = ", sampleList)
```

리스트 실습 소스코드 - 3

```
print("₩n5. 샘플 리스트의 한 요소의 값의 위치")
print("    sampleList = ", sampleList)
print("    sampleList.index(4) = ", sampleList.index(4))
print("    sampleList.index(3) = ", sampleList.index(3))
print("    sampleList.index(1) = ", sampleList.index(1))

print("₩n6. 샘플 리스트의 특정위치에 한 값의 추가")
print("    sampleList = ", sampleList)
sampleList.insert(2, 9)
print("    sampleList의 2번 요소(즉, 3번째 요소)에 9의 값을 추가 : ")
print("    sampleList.insert(2, 9) 후의 sampleList = ", sampleList)
```

```
5. 샘플 리스트의 한 요소의 값의 위치
   sampleList =  [4, 3, 1]
   sampleList.index(4) =  0
   sampleList.index(3) =  1
   sampleList.index(1) =  2

6. 샘플 리스트의 특정위치에 한 값의 추가
   sampleList =  [4, 3, 1]
   sampleList의 2번 요소(즉, 3번째 요소)에 9의 값을 추가 :
   sampleList.insert(2, 9) 후의 sampleList =  [4, 3, 9, 1]
```

리스트 실습 소스코드 - 4

```
print("₩n7. 샘플 리스트에 또 다른 리스트를 붙이기(extend)")
print("    sampleList = ", sampleList)
sampleList.extend([3, 6])
print("    sampleList.extend([3, 6]) 후의 sampleList = ", sampleList)
addList = [3,3]
print("    addList = [3,3]")
sampleList.extend(addList)
print("    sampleList.extend(addList) 후의 sampleList = ", sampleList)

print("₩n8. 샘플 리스트에서 특정 값의 갯수 카운트 (count)")
print("    sampleList = ", sampleList)
print("    sampleList.count(3) = ", sampleList.count(3))
print("    sampleList.count(9) = ", sampleList.count(9))
print("    sampleList.count(2) = ", sampleList.count(2))
```

```
7. 샘플 리스트에 또 다른 리스트를 붙이기(extend)
   sampleList =  [4, 3, 9, 1]
   sampleList.extend([3, 6]) 후의 sampleList =  [4, 3, 9, 1, 3, 6]
   addList = [3,3]
   sampleList.extend(addList) 후의 sampleList =  [4, 3, 9, 1, 3, 6, 3, 3]

8. 샘플 리스트에서 특정 값의 갯수 카운트 (count)
   sampleList =  [4, 3, 9, 1, 3, 6, 3, 3]
   sampleList.count(3) =  4
   sampleList.count(9) =  1
   sampleList.count(2) =  0
```

리스트 실습 소스코드 - 5

```
print("₩n9. 샘플 리스트에 한 값의 제거(remove)")
print("    9-1. 샘플 리스트에 한 값의 제거")
print("        sampleList = ", sampleList)
print("        sampleList.index(9) = ", sampleList.index(9))
sampleList.remove(9)
print("        sampleList.remove(9) 후의 sampleList = ", sampleList)
print("    9-2. 샘플 리스트에 한 값의 제거(같은 값이 여러 개가 있을 경우)")
print("        sampleList = ", sampleList)
print("        sampleList.count(3) = ", sampleList.count(3))
print("        sampleList.index(3) = ", sampleList.index(3))
sampleList.remove(3)
print("        sampleList = ", sampleList)
print("₩n10. 샘플 리스트의 갯수 카운트 ( len(샘플리스트) )")
print("    sampleList = ", sampleList)
print("    len(sampleList) = ", len(sampleList))
```

```
9. 샘플 리스트에 한 값의 제거(remove)
   9-1. 샘플 리스트에 한 값의 제거
      sampleList =  [4, 3, 9, 1, 3, 6, 3, 3]
      sampleList.index(9) =  2
      sampleList.remove(9) 후의 sampleList =  [4, 3, 1, 3, 6, 3, 3]
   9-2. 샘플 리스트에 한 값의 제거(같은 값이 여러 개가 있을 경우)
      sampleList =  [4, 3, 1, 3, 6, 3, 3]
      sampleList.count(3) =  4
      sampleList.index(3) =  1
      sampleList =  [4, 1, 3, 6, 3, 3]

10. 샘플 리스트의 갯수 카운트 ( len(샘플리스트) )
    sampleList =  [4, 1, 3, 6, 3, 3]
    len(sampleList) =  6
```

리스트 실습 소스코드 - 6

```
print("₩n11. 샘플 리스트를 이용한 Loop")
print("    11-1. Python스러운 Loop")
print("        sampleList = ", sampleList)
total = 0
for i in sampleList:
    total += i
print("        total = 0₩n        for i in sampleList:")
print("            total += i₩n        print(total) = %s" %total)
print("    11-2. range(len()) 함수 이용한 Loop")
print("        sampleList = ", sampleList)
total = 0
for i in range(len(sampleList)):
    total += sampleList[i]
print("        total = 0₩n        for i in range(len(sampleList)):")
print("            total += sampleList[i]₩n        print(total) = %s" %total)
```

```
11. 샘플 리스트를 이용한 Loop
   11-1. Python스러운 Loop
       sampleList =  [4, 1, 3, 6, 3, 3]
       total = 0
       for i in sampleList:
         total += i
       print(total) = 20
   11-2. range(len()) 함수 이용한 Loop
       sampleList =  [4, 1, 3, 6, 3, 3]
       total = 0
       for i in range(len(sampleList)):
         total += sampleList[i]
       print(total) = 20
```

리스트 실습 소스코드 - 7

```
print("₩n12. 샘플 리스트의 정렬")
print("     sampleList = ", sampleList)
print("     print(sorted(sampleList)) = ", sorted(sampleList))
print("     sortedSampleList = sorted(sampleList) ")
sortedSampleList = sorted(sampleList)
print("     sortedSampleList = ",sortedSampleList)
print("     sampleList = ", sampleList)
print("     print(sampleList.sort()) = ", sampleList.sort())
print("     sampleList = ", sampleList)
```

```
12. 샘플 리스트의 정렬
    sampleList =  [4, 1, 3, 6, 3, 3]
    print(sorted(sampleList)) =  [1, 3, 3, 3, 4, 6]
    sortedSampleList = sorted(sampleList)
    sortedSampleList =  [1, 3, 3, 3, 4, 6]
    sampleList =  [4, 1, 3, 6, 3, 3]
    print(sampleList.sort()) =  None
    sampleList =  [1, 3, 3, 3, 4, 6]
```

2. 튜플

튜플의 메소드

- 리스트와 다른 점
  - 리스트는 [과]으로 둘러싸지만 튜플은 (과)으로 둘러쌈
  - 리스트는 그 값의 생성/삭제/수정이 가능, 튜플은 그 값을 바꿀 수 없음
- 튜플에서 사용할 수 있는 인덱스는 제한적임
  - 이유 : 변형이 불가능한 자료형이므로

메소드	내용
index()	튜플 내에서 매개변수로 입력한 데이터와 일치하는 첫 번째 요소의 첨자를 알려줌 일치하는 요소가 없으면, 오류를 일으킴 튜플.index()
len(튜플)	튜플에 포함된 전체 요소의 개수를 출력하는 메소드 len(튜플)
count()	매개변수로 입력한 데이터와 일치하는 요소가 몇 개 있는지 카운트함 튜플.count()
sorted(튜플)	튜플을 sorting한 결과를 출력하는 메소드 sorted(튜플)

튜플 실습 소스코드 - 1

```
print("₩n1. 샘플튜플  출력")
sampleTuple = (4, 3, 9, 1, 3, 6, 3, 3)
print("    sampleTuple = ", sampleTuple)

print("₩n2. 샘플튜플의 한 요소의 값의 위치")
print("    sampleTuple = ", sampleTuple)
print("    sampleTuple.index(4) = ", sampleTuple.index(4))
print("    sampleTuple.index(3) = ", sampleTuple.index(3))
print("    sampleTuple.index(1) = ", sampleTuple.index(1))
# 없는 요소에 값의 위치를 찾고자할 때는 오류(valueError)를 일으킴. 즉,
#print("    sampleTuple.index(2) = ", sampleTuple.index(2))

print("₩n3. 샘플튜플의 갯수 카운트 ( len(샘플튜플) )")
print("    sampleTuple = ", sampleTuple)
print("    len(sampleTuple) = ", len(sampleTuple))
```

```
1. 샘플튜플  출력
   sampleTuple =  (4, 3, 9, 1, 3, 6, 3, 3)

2. 샘플튜플의 한 요소의 값의 위치
   sampleTuple =  (4, 3, 9, 1, 3, 6, 3, 3)
   sampleTuple.index(4) =  0
   sampleTuple.index(3) =  1
   sampleTuple.index(1) =  3

3. 샘플튜플의 갯수 카운트 ( len(샘플튜플) )
   sampleTuple =  (4, 3, 9, 1, 3, 6, 3, 3)
   len(sampleTuple) =  8
```

튜플 실습 소스코드 - 2

```
print("₩n4. 샘플튜플에서 특정 값의 갯수 카운트 (count)")
print("    sampleTuple = ", sampleTuple)
print("    sampleTuple.count(3) = ", sampleTuple.count(3))
print("    sampleTuple.count(9) = ", sampleTuple.count(9))
print("    sampleTuple.count(2) = ", sampleTuple.count(2))

print("₩n5. 샘플튜플의 정렬")
print("    sampleTuple = ", sampleTuple)
print("    print(sorted(sampleTuple)) = ", sorted(sampleTuple))
print("    sortedSampleTuple = sorted(sampleTuple) ")
sortedSampleTuple = sorted(sampleTuple)
print("    sortedSampleTuple = ",sortedSampleTuple)
print("    sampleTuple = ", sampleTuple)
# sort()는 변경불가능한 자료형에서 사용하지 못함. 다음은 AttributeError가 남
#print("    print(sampleTuple.sort()) = ", sampleTuple.sort())
#print("    sampleTuple = ", sampleTuple)
```

```
4. 샘플튜플에서 특정 값의 갯수 카운트 (count)
   sampleTuple =  (4, 3, 9, 1, 3, 6, 3, 3)
   sampleTuple.count(3) =  4
   sampleTuple.count(9) =  1
   sampleTuple.count(2) =  0

5. 샘플튜플의 정렬
    sampleTuple =  (4, 3, 9, 1, 3, 6, 3, 3)
    print(sorted(sampleTuple)) =  [1, 3, 3, 3, 3, 4, 6, 9]
    sortedSampleTuple = sorted(sampleTuple)
    sortedSampleTuple =  [1, 3, 3, 3, 3, 4, 6, 9]
    sampleTuple =  (4, 3, 9, 1, 3, 6, 3, 3)
```

3. 세트

세트

- set 자료형
  - 값을 순서없이 저장하면서, 중복을 불허하는 자료형
    - 다음에 언급될 dictionary 자료형에서 key로만 이루어진 자료형
  - 생성할 때, 중괄호 { }를 이용하여 생성
    - 예) aSet1 = {1,2,3}
  - 공백 세트를 생성할 때는 aSet1 = {}로 안됨. 공백 딕셔너리가 만들어짐
    - aSet1 = set() 로 생성하여야 함.
  - 시퀀스 자료형이 아님
    - 리스트나 튜플과 달리, 인덱스가 정수여야 할 필요가 없음
- 사용 예)
  - 문장에 있는 단어의 갯수 count할 때 유용
    - 예)
      - aList = ['홍길동', '최판서', '홍길동', '성춘향', '이성계', '최판서']
      - aSet2 = set(aList)
      - aSet2

```
>>> aList = ['홍길동', '최판서', '홍길동', '성춘향', '이성계', '최판서']
>>> aSet2 = set(aList)
>>> aSet2
{'최판서', '이성계', '홍길동', '성춘향'}
>>>
```

세트의 메소드

- 예)
  - a = {1, 2, 3, 4}, b = {2, 3, 4, 5, 6}, c = {1, 2}, d = {3, 4}

메소드	연산	기호	연산결과
a.union(b)	합집합	a | b	{1, 2, 3, 4, 5, 6}
a.intersection(b)	교집합	a & b	{2, 3, 4}
a.difference(b)	차집합	a – b	{1}
a.symmetric_difference(b)	대칭차집합	a ^ b	{1, 5, 6}
c.issubset(a)	부분집합	c <= a	True
c.issuperset(a)	상위집합	c >= a	False
c.isdisjoint(d)	서로소		True

세트 실습 소스코드 - 1

```
print("₩n1. 샘플 세트  출력")
sampleSet1 = {4, 3, 1, 2}
sampleSet2 = {2, 3, 4, 5, 6}
sampleSet3 = {1, 2}
sampleSet4 = {3, 4}
sampleSet5 = set()
print("    sampleSet1 = ", sampleSet1)
print("    sampleSet2 = ", sampleSet2)
print("    sampleSet3 = ", sampleSet3)
print("    sampleSet4 = ", sampleSet4)
print("    sampleSet5 = ", sampleSet5, ": 공백 세트를 생성")
```

```
1. 샘플 세트  출력
   sampleSet1 =  {1, 2, 3, 4}
   sampleSet2 =  {2, 3, 4, 5, 6}
   sampleSet3 =  {1, 2}
   sampleSet4 =  {3, 4}
   sampleSet5 =  set() : 공백 세트를 생성

2. 샘플세트의 갯수 카운트 ( len(샘플세트) )
   sampleSet1 =  {1, 2, 3, 4}
   len(sampleSet1) =  4
```

```
print("₩n2. 샘플세트의 갯수 카운트 ( len(샘플세트) )")
print("    sampleSet1 = ", sampleSet1)
print("    len(sampleSet1) = ", len(sampleSet1))
```

세트 실습 소스코드 - 2

```
print("₩n3. 샘플세트의 정렬 : 세트 자료형은 정렬되어 유지되므로 의미없음")
print("     sampleSet1 = ", sampleSet1)
print("     print(sorted(sampleSet1)) = ", sorted(sampleSet1))
print("     sortedSampleSet1 = sorted(sampleSet1) ")
sortedSampleSet1 = sorted(sampleSet1)
print("     sortedSampleSet1 = ",sortedSampleSet1)
print("     sampleSet1 = ", sampleSet1)

print("₩n4. 샘플세트를 이용한 Python스러운 Loop")
print("     sampleSet1 = ", sampleSet1)
total = 0
for i in sampleSet1:
    total += i
print("     total = 0₩n     for i in sampleSet1:")
print("         total += i₩n     print(total) = %s" %total)
```

```
3. 샘플세트의 정렬 : 세트 자료형은 정렬되어 유지되므로 의미없음
     sampleSet1 =  {1, 2, 3, 4}
     print(sorted(sampleSet1)) =  [1, 2, 3, 4]
     sortedSampleSet1 = sorted(sampleSet1)
     sortedSampleSet1 =  [1, 2, 3, 4]
     sampleSet1 =  {1, 2, 3, 4}

4. 샘플세트를 이용한 Python스러운 Loop
     sampleSet1 =  {1, 2, 3, 4}
     total = 0
     for i in sampleSet1:
        total += i
     print(total) = 10
```

세트 실습 소스코드 - 3

```
print("₩n5. 샘플세트의 연산 ")
print("    sampleSet1 합집합 sampleSet2  ", sampleSet1.union(sampleSet2))
print("    sampleSet1 교집합 sampleSet2  ", sampleSet1.intersection(sampleSet2))
print("    sampleSet1 차집합 sampleSet2  ", sampleSet1.difference(sampleSet2))
print("    sampleSet1 대칭차집합 sampleSet2  ", sampleSet1.symmetric_difference(sampleSet2))
print("    sampleSet3이 sampleSet1의 부분집합 ?  ", sampleSet3.issubset(sampleSet1))
print("    sampleSet3이 sampleSet1의 상위집합 ?  ", sampleSet3.issuperset(sampleSet1))
print("    sampleSet3과 sampleSet4가 서로  소 ?  ", sampleSet3.isdisjoint(sampleSet4))
```

```
5. 샘플세트의 연산
    sampleSet1 합집합 sampleSet2   {1, 2, 3, 4, 5, 6}
    sampleSet1 교집합 sampleSet2   {2, 3, 4}
    sampleSet1 차집합 sampleSet2   {1}
    sampleSet1 대칭차집합 sampleSet2   {1, 5, 6}
    sampleSet30이 sampleSet1의 부분집합 ?   True
    sampleSet30이 sampleSet1의 상위집합 ?   False
    sampleSet3과 sampleSet4가 서로  소 ?   True
```

4. 딕셔너리

딕셔너리

- dict 자료형
  - 순서대로 정렬되지 않기 때문에 인덱스가 아닌 키(key)를 사용해야 함
    - (key, value) 쌍
  - 시퀀스 자료형이 아님
    - 리스트나 튜플과 달리, 인덱스가 정수여야 할 필요가 없음
    - 인덱스가 변형 불가능한 어떤 값이라도 가능함
  - 중괄호 { } 를 이용하여 표현
- 예) 학생 정보의 보관
  - 리스트를 이용
    - Names = [' Ana ' , ' John ' , ' Denise ' , ' Katy ']
    - Grade = [' B ' , ' A+ ' , ' A ' , ' A ']
  - 딕셔너리를 이용
    - grades = {'Ana':'B', 'John':'A+', 'Denise':'A', 'Katy':'A'}

딕셔너리 메소드

메소드	설명
len(d)	딕셔너리 d의 길이 리턴
d[key]	딕셔너리의 특정키로 값(value) 조회
d[key] =value	딕셔너리의 특정 항목의 값을 삽입하거나, 바꿀 때 사용
del d[key]	특정 key로 딕셔너리 아이템을 삭제하기
key in d	딕셔너리에 특정 key가 있는지 여부의 불리언 값을 리턴
iter(d)	또는 iter(d.keys()) 를 이용하여 딕셔너리의 모든 아이템을 순회하는 루프 만들기
d.get(key, default)	딕셔너리 d에 key가 있으면 d[key] 값을 리턴하고, 없으면 default를 리턴
d.items()	딕셔너리 d의 아이템들을 출력
d.keys()	딕셔너리 d의 key들을 출력
d.values()	딕셔너리 d의 value들을 출력
d.pop(key)	딕셔너리 d의 특정 key를 갖는 아이템 삭제
d.popitem()	딕셔너리의 임의의 아이템 하나의 삭제
Sorted(d.items())	딕셔너리를 key의 오름차순으로 정렬

딕셔너리 실습 소스코드 - 1

딕셔너리 초기화 방법 5가지는 동일함.

```
asampleDic = {"홍길동":97, "김판서":75, "성춘향":83, "이율곡":61}
bsampleDic = dict(홍길동=97, 김판서=75, 성춘향=83, 이율곡=61)
csampleDic = dict(zip(["홍길동", "김판서", "성춘향", "이율곡"],[97, 75, 83, 61]))
dsampleDic = dict([("김판서", 75), ("홍길동", 97), ("성춘향", 83), ("이율곡", 61)])
esampleDic = dict({"이율곡":61, "성춘향":83, "김판서":75, "홍길동":97 })
```

```
1. 다음의 5가지 딕셔너리 초기화 방법은 동일
asampleDic = {'홍길동': 97, '김판서': 75, '성춘향': 83, '이율곡': 61}
bsampleDic = {'홍길동': 97, '김판서': 75, '성춘향': 83, '이율곡': 61}
csampleDic = {'홍길동': 97, '김판서': 75, '성춘향': 83, '이율곡': 61}
dsampleDic = {'김판서': 75, '홍길동': 97, '성춘향': 83, '이율곡': 61}
esampleDic = {'이율곡': 61, '성춘향': 83, '김판서': 75, '홍길동': 97}
If (asampleDic==bsampleDic) =  True
If (bsampleDic==csampleDic) =  True
If (csampleDic==dsampleDic) =  True
If (dsampleDic==esampleDic) =  True
If (esampleDic==asampleDic) =  True
```

딕셔너리 실습 소스코드 - 2

```
print("₩n2. 딕셔너리의 길이 : len(sampleDic)")
sampleDic = {"홍길동":97,"김판서":75,"성춘향":83,"이율곡":61}
print("    len(sampleDic) = ", len(sampleDic))


print("₩n3. 딕셔너리의 키로 값 조회 : sampleDic[key]")
print("    3-1. 특정 키로 값찾기 : sampleDic[key]")
print("        sampleDic['홍길동'] = ", sampleDic["홍길동"])
print("        sampleDic['김판서'] = ", sampleDic["김판서"])
...
print("₩n    3-2. 루프를 이용한 모든 키의 값 검색")
for k in sampleDic:
    print("        sampleDic['"+k+"'] = ", sampleDic[k])


print("₩n4. 특정 아이템의 값 바꾸기 : sampleDic[key] = value")
sampleDic['홍길동'] = 10
sampleDic['김판서'] = 20
```

```
2. 딕셔너리의 길이 : len(sampleDic)
   sampleDic =  {'홍길동': 97, '김판서': 75, '성춘향': 83, '이율곡': 61}
   len(sampleDic) =  4

3. 딕셔너리의 키로 값 조회 : sampleDic[key]
   3-1. 특정 키로 값찾기 : sampleDic[key]
      sampleDic['홍길동'] =  97
      sampleDic['김판서'] =  75
      sampleDic['성춘향'] =  83
      sampleDic['이율곡'] =  61

   3-2. 루프를 이용한 모든 키의 값 검색
      sampleDic['홍길동'] =  97
      sampleDic['김판서'] =  75
      sampleDic['성춘향'] =  83
      sampleDic['이율곡'] =  61

4. 특정 아이템의 값 바꾸기 : sampleDic[key] = value
      sampleDic['홍길동'] =  97
      sampleDic['김판서'] =  75
      sampleDic['성춘향'] =  83
      sampleDic['이율곡'] =  61

      sampleDic['홍길동'] = 10
      sampleDic['김판서'] = 20
      sampleDic['성춘향'] = 30
      sampleDic['이율곡'] = 40

      sampleDic['홍길동'] =  10
      sampleDic['김판서'] =  20
      sampleDic['성춘향'] =  30
      sampleDic['이율곡'] =  40
```

딕셔너리 실습 소스코드 - 3

```
print("₩n5. 특정 키로 딕셔너리 아이템 삭제하기 : del sampleDic[key]")
del sampleDic['김판서']

print("₩n6. 딕셔너리에 특정 키가 있는지 여부의 불리언 결과 출력 : key in sampleDic")
if '홍길동' in sampleDic:
    print('    홍길동 있음')
else:
    print('    홍길동 없음')
if '김판서' in sampleDic:
    print('    김판서 있음')
else:
    print('    김판서 없음')
for k in sampleDic:
    print('    '+k +'가 sampleDic에 있음 : ', k in sampleDic)
```

```
5. 특정 키로 딕셔너리 아이템 삭제하기 : del sampleDic[key]
sampleDic =  {'홍길동': 10, '김판서': 20, '성춘향': 30, '이율곡': 40}
len(sampleDic) =  4

del sampleDic['김판서']

sampleDic =  {'홍길동': 10, '성춘향': 30, '이율곡': 40}
len(sampleDic) =  3

6. 딕셔너리에 특정 키가 있는지 여부의 불리언 결과 출력 : key in sampleDic
sampleDic =  {'홍길동': 10, '성춘향': 30, '이율곡': 40}
홍길동 있음
김판서 없음
홍길동가 sampleDic에 있음 :  True
성춘향가 sampleDic에 있음 :  True
이율곡가 sampleDic에 있음 :  True
```

딕셔너리 실습 소스코드 - 4

```
print("₩n7. 딕셔너리 아이템을 순회하는 루프만들기 : iter(sampleDic)")
print("₩n    7-1. 파이썬스러운 반복 : iter(sampleDic)")
for k in iter(sampleDic):
    print('       '+k)


print("₩n    7-2. 7-1과 같지만 읽기좋은 반복 표현 : iter(sampleDic).keys()")
for k in iter(sampleDic.keys()):
    print('       '+k)


print("₩n8. sampleDic.get(k, default)")
print("    sampleDic에 k가 있으면 sampleDic[k]를 반환하고, 아니면 default를 반환")
print("₩n    홍길동은 ",sampleDic.get('홍길동','결시'))
print("    성춘향은 ",sampleDic.get('성춘향','결시'))
print("    이율곡은 ",sampleDic.get('이율곡','결시'))
print("    김판서은 ",sampleDic.get('김판서','결시'))
print("    이순신은 ",sampleDic.get('이순신','결시'))
```

```
7. 딕셔너리 아이템을 순회하는 루프만들기 : iter(sampleDic)
(김판서, 50) 추가
sampleDic =  {'홍길동': 10, '성춘향': 30, '이율곡': 40, '김판서': 50}

7-1. 파이썬스러운 반복 : iter(sampleDic)
   홍길동
   성춘향
   이율곡
   김판서

7-2. 7-1과 같지만 읽기좋은 반복 표현 : iter(sampleDic).keys()
   홍길동
   성춘향
   이율곡
   김판서

8. sampleDic.get(k, default)
   sampleDic에 k가 있으면 sampleDic[k]를 반환하고, 아니면 default를 반환
   sampleDic =  {'홍길동': 10, '성춘향': 30, '이율곡': 40, '김판서': 50}

   홍길동은  10
   성춘향은  30
   이율곡은  40
   김판서은  50
   이순신은  결시
```

딕셔너리 실습 소스코드 - 5

```
print("₩n9. 샘플 딕셔너리의 아이템, 키, 값 출력")
print("₩n    9-1. 샘플딕셔너리의 요소(item) 출력")
print("        sampleDic.items() : ", sampleDic.items())
print("₩n    9-2. 샘플딕셔너리의 키 출력")
print("        sampleDic.keys() : ", sampleDic.keys())
print("₩n    9-3. 샘플딕셔너리의 값  출력")
print("        sampleDic.values() : ", sampleDic.values())

print("₩n10. 샘플딕셔너리의 제일 마지막 아이템 삭제")
print("₩n    10-1. 샘플딕셔너리의 특정 키를 갖는 아이템 삭제 : pop('키')")
sampleDic.pop('홍길동')
print("₩n    10-2. 샘플딕셔너리의 임의의 아이템 삭제 : popitem()")
sampleDic.popitem()

print("₩n11. 딕셔너리 아이템 지우기: sampleDic.clear()")
sampleDic.clear()
```

9. 샘플 딕셔너리의 아이템, 키, 값 출력

9-1. 샘플딕셔너리의 요소(item) 출력
sampleDic.items() : dict_items([('홍길동', 10), ('성춘향', 30), ('이율곡', 40), ('김판서', 50)])

9-2. 샘플딕셔너리의 키 출력
sampleDic.keys() : dict_keys(['홍길동', '성춘향', '이율곡', '김판서'])

9-3. 샘플딕셔너리의 값 출력
sampleDic.values() : dict_values([10, 30, 40, 50])

10. 샘플딕셔너리의 제일 마지막 아이템 삭제

10-1. 샘플딕셔너리의 특정 키를 갖는 아이템 삭제 : pop('키')
sampleDic = {'홍길동': 10, '성춘향': 30, '이율곡': 40, '김판서': 50}
sampleDic.pop('홍길동')
sampleDic = {'성춘향': 30, '이율곡': 40, '김판서': 50}

10-2. 샘플딕셔너리의 임의의 아이템 삭제 : popitem()
sampleDic = {'성춘향': 30, '이율곡': 40, '김판서': 50}
sampleDic.popitem()
sampleDic = {'성춘향': 30, '이율곡': 40}

11. 딕셔너리 아이템 지우기: sampleDic.clear()
sampleDic = {'성춘향': 30, '이율곡': 40}
sampleDic.clear()
sampleDic = {}
다시 예제 자료 입력
sampleDic = {'홍길동': 10, '김판서': 20, '성춘향': 30, '이율곡': 40}

딕셔너리 실습 소스코드 - 6

```
print("₩n    12-1. 딕셔너리를 이용한 값의 평균 구하기")
keys = sampleDic.keys()
values = sampleDic.values()
n = 0
for val in values:
    n += val
print("       값 평균 = ", n/len(values))

print("₩n    12-2. 딕셔너리의 키나 값을 리스트로 만들기")
print("₩n       키(keys)를 리스트로 출력 : ", list(keys))
print("       값(values)를 리스트로 출력 : ", list(values))

print("₩n    12-3. 키값을 기준으로 정렬 ")
print("       키(이름의 오름차순)을 기준으로 정렬한 결과")
print("         (1) 아이템 출력 : ", sorted(sampleDic.items()))

print("₩n    12-4. 딕셔너리를 이용한 집합 연산 ")
print("       A : ", keys)
print("       B : ", '{"홍길동","이성계","김판서"}')
print("       A & B : ", keys & {"홍길동","이성계","김판서"})
print("       A ^ B : ", keys ^ {"홍길동","이성계","김판서"})
```

12. 딕셔너리를 이용한 다양한 작업

12-1. 딕셔너리를 이용한 값의 평균 구하기

```
sampleDic =  {'홍길동': 10, '김판서': 20, '성춘향': 30, '이율곡': 40}
값 평균 =  25.0
```

12-2. 딕셔너리의 키나 값을 리스트로 만들기

```
type(keys) :  <class 'dict_keys'>
type(values) :  <class 'dict_keys'>
type(list(keys)) :  <class 'list'>
type(list(values)) :  <class 'list'>

키(keys)를 리스트로 출력 :  ['홍길동', '김판서', '성춘향', '이율곡']
값(values)를 리스트로 출력 :  [10, 20, 30, 40]
```

12-3. 키값을 기준으로 정렬

```
sampleDic =  {'홍길동': 10, '김판서': 20, '성춘향': 30, '이율곡': 40}
키(이름의 오름차순)을 기준으로 정렬한 결과
  (1) 아이템 출력 :  [('김판서', 20), ('성춘향', 30), ('이율곡', 40), ('홍길동', 10)]
  (2) 아이템의 리스트 출력 :  [('김판서', 20), ('성춘향', 30), ('이율곡', 40), ('홍길동', 10)]
  (3) 키 출력 ['김판서', '성춘향', '이율곡', '홍길동']
  (4) 키의 리스트 출력 ['김판서', '성춘향', '이율곡', '홍길동']
  (5) 값 출력 [10, 20, 30, 40]
  (6) 값의 리스트  [10, 20, 30, 40]
```

12-4. 딕셔너리를 이용한 집합 연산

```
A :  dict_keys(['홍길동', '김판서', '성춘향', '이율곡'])
B :  {"홍길동","이성계","김판서"}
A & B :  {'김판서', '홍길동'}
A ^ B :  {'성춘향', '이율곡', '이성계'}
```