Chapter 09
튜플과 딕셔너리

목차

학습목표

- 데이터 수정이 불가한 튜플에 대해서 학습합니다.
- 키(key)와 값(value)의 쌍으로 데이터를 관리하는 딕셔너리에 대해서 학습합니다.

Section 01

이 장에서 만들 프로그램

이 장에서 만들 프로그램

1. 수학시험 프로그램

- 수학 문제에 답을 입력하면 결과를 보여주는 프로그램
- 모든 문제를 풀고 나면 정답 개수, 오답 개수 그리고 점수를 출력함

```
문제 :  3+2
정답을 입력하세요. 5
...생략...
문제 :  4/2
정답을 입력하세요. 2
-----------------------------
정답 개수:  7
오답 개수:  0
Total Score:  27
-----------------------------
```

이 장에서 만들 프로그램

2. 회원가입 프로그램

- 아이디와 비밀번호를 입력 받아 저장하는 회원가입 프로그램
- 회원 입력이 끝나면 모든 회원 리스트를 출력함

```
1. 회원가입, 2. 프로그램 종료  1
아이디를 입력하세요.    superman2020
비밀번호를 입력하세요.  super2020

1. 회원가입, 2. 프로그램 종료  1
아이디를 입력하세요.    betman2021
비밀번호를 입력하세요.  bet2021

1. 회원가입, 2. 프로그램 종료  2
-----------------------------
아이디:비밀번호
-----------------------------
superman2020  :    super2020
batman2021    :    bet2021
-----------------------------
```

Section 02
튜플이란?

튜플 vs 리스트

- 튜플(tuple)
  - 리스트와 비슷하게 데이터를 묶어서 처리하는 컨테이너 자료형임
  - 튜플에 포함된 아이템을 수정할 수 없음
  - 데이터가 수정되면 안 되는 예 : 회사의 급여 명세서

그림 9-1 데이터가 수정되면 안 되는 경우

튜플 선언

튜플 선언

- 소괄호(())를 사용하여 선언함

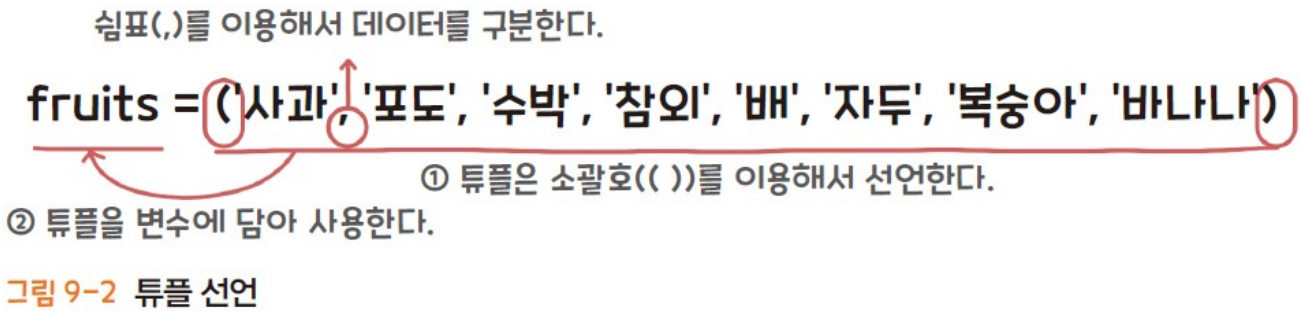


그림 9-2 튜플 선언

- 셸모드에서 과일 집합을 튜플로 선언하고 출력하기

```
>>> fruits = ('사과', '포도', '수박', '참외', '배', '자두', '복숭아', '바나나')  ← 튜플 선언
>>> fruits  ← 아이템 출력
('사과', '포도', '수박', '참외', '배', '자두', '복숭아', '바나나')
>>> type(fruits)  ← 자료형 출력
<class 'tuple'>
```

확인문제

확인문제

1. 튜플에 대한 설명으로 옳은 것은 무엇인가?

   ① 튜플은 리스트와 같은 컨테이너 자료형이다.

   ② 튜플의 아이템은 자유롭게 수정할 수 있다.

   ③ 튜플을 선언할 때는 대괄호([])를 사용한다.

   ④ 파이썬 3.x에서는 튜플을 지원하지 않는다.

2. 다음 데이터 집합을 컨테이너 자료형으로 선언하려고 할 때, 튜플로 선언하는 것이 가장 적합한 것은 무엇인가?

   ① 내가 좋아하는 색상 : 빨간색, 노란색, 흰색, 보라색

   ② 내가 좋아하는 음식 : 김치, 볶음밥, 짜장면

   ③ 학생의 혈액형 : A형, B형, O형, AB형

   ④ 학생 명단 : 홍길동, 홍길순, 홍길자, 홍철수, 홍영희

3. 실행 결과를 보고 빈칸을 채우시오.

```
sports = (  )'태권도', '야구', '농구', '축구', '배구', '권투', '양궁'(  )
print('sports : ', sports)
print('type : ', type(sports))
```

```
sports :  ('태권도', '야구', '농구', '축구', '배구', '권투', '양궁')
type :  <class 'tuple'>
```

정답

1. ①　　2. ③　　3. (,)

Section 03
튜플 조회하기

아이템 조회

튜플의 아이템 조회

- 튜플은 아이템의 수정이 불가능하기 때문에 아이템의 삽입, 삭제, 정렬 등의 기능은 없고 조회 기능만 주로 사용함
- 리스트와 마찬가지로 인덱스를 이용하여 아이템에 접근함

- fruits 튜플에서 특정 아이템을 조회하기

```
>>> fruits = ('사과', '포도', '수박', '참외', '배', '자두', '복숭아', '바나나')
>>> fruits[3]
'참외'
>>> fruits[len(fruits)-1]
'바나나'
```

튜플 선언

인덱스가 3인 아이템 출력

마지막 인덱스 아이템 출력

문제 해결

문제 해결 9-1 인덱스가 홀수인 아이템 조회하기 ch09_sol_01.py

sports 튜플에서 인덱스가 홀수인 아이템을 출력하는 프로그램을 만들어봅시다.

```
01  sports = ('태권도', '야구', '농구', '축구', '배구', '권투', '양궁')
02
03  for index, item in enumerate(sports):
04      if index % 2 == 1:
05          print(index, ' : ', item)
```

```
1  :  야구
3  :  축구
5  :  권투
```

특정 아이템의 인덱스 조회

- index() 함수
  - 튜플 내 특정 아이템의 인덱스 조회할 때 사용하는 함수
  - fruits 튜플에서 '포도'와 '바나나'가 몇 번 인덱스에 있는지 확인하기

```
>>> fruits = ('사과', '포도', '수박', '참외', '배', '자두', '복숭아', '바나나')
>>> fruits.index('포도')        '포도'에 해당하는 인덱스 조회
1
>>> fruits.index('바나나')      '바나나'에 해당하는 인덱스 조회
7
```

문제 해결

문제 해결 9-2 아이템 값으로 인덱스 출력하기 ch09_sol_02.py

names 튜플에서 사용자가 선수 이름을 입력하면 이름에 해당하는 인덱스를 출력해봅시다.

```
01   names = ('박찬호', '이승엽', '박세리', '박지성', '이순철', '선동열', '손흥민', '김연아')
02
03   inputData = input('검색하려는 이름을 입력하세요. : ')
04   print('이름 : ', inputData, ', 인덱스 : ', names.index(inputData))
```

```
검색하려는 이름을 입력하세요. : 김연아
이름 : 김연아 , 인덱스 : 7
```

아이템 유/무 확인

- 아이템이 있는지 확인 : **in**

코드 9-1 ch09_01.py

```
01  colors = ('Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Purple')
02  print('Green' in colors)
```

'Green'과 일치하는 아이템이 있는지 확인

```
True
```

- 아이템이 없는지 확인 : **not in**

코드 9-2 ch09_02.py

```
01  colors = ('Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Purple')
02  print('Green' not in colors)
```

'Green'과 일치하는 아이템이 없는지 확인

```
False
```

'Green'과 일치하는 아이템이 존재하므로 결과는 False

문제 해결

문제 해결 9-3 학점 경고 프로그램 만들기 ch09_sol_03.py

scores는 1학기 성적을 튜플로 나타낸 것입니다. F 학점이 있으면 '경고'를 출력하는 프로그램을 만들어봅시다.

```
01   scores = ('A', 'A+', 'B', 'B-', 'F')
02
03   if 'F' in scores:
04       print('경고')
```

```
경고
```

튜플 결합

'+' 연산자

- 서로 다른 튜플을 결합할 때 사용함

```
>>> tuple1 = (1, 2, 3)          ← tuple1 선언
>>> tuple2 = (10, 20, 30)       ← tuple2 선언
>>> sumTuple = tuple1 + tuple2  ← tuple1 + tuple2 결합
>>> sumTuple                    ← 결합된 튜플 출력
(1, 2, 3, 10, 20, 30)
```

하나 더 알기 **extend() 함수는 튜플에서 사용할 수 없나요?**

리스트와 리스트를 연결할 때 사용한 extend() 함수는 튜플에서 사용할 수 없습니다. extend() 함수는 리스트를 연장할 때 사용하는데 튜플은 한 번 선언되면 수정이 불가능하기 때문에 연장하는 기능도 사용할 수 없습니다. 따라서 튜플에서 extend() 함수를 사용하면 에러가 발생합니다.

```
>>> tuple1 = (1, 2, 3)
>>> tuple2 = (10, 20, 30)
>>> tuple1.extend(tuple2)
Traceback (most recent call last):
  File "<pyshell#20>", line 1, in <module>
    tuple1.extend(tuple2)
AttributeError: 'tuple' object has no attribute 'extend'
```

슬라이싱

슬라이싱

- 튜플에서 필요한 부분의 아이템만 뽑아내는 것을 뜻함
- 사용법은 리스트에서 슬라이싱할 때와 같음

[n:m] → n 인덱스부터 (m-1) 인덱스 까지의 아이템을 슬라이싱(추출)한다.

그림 9-3 슬라이싱 표기법

```
>>> animals = ('호랑이', '사자', '곰', '여우', '늑대')
>>> animals
('호랑이', '사자', '곰', '여우', '늑대')
>>> animals[:3]
('호랑이', '사자', '곰')
>>> animals[3:]
('여우', '늑대')
>>> animals[1:4]
('사자', '곰', '여우')
>>> animals[len(animals)-2:]
('여우', '늑대')
```

인덱스 0부터 2(3-1)까지의 아이템 슬라이싱

인덱스 3부터 끝까지의 아이템 슬라이싱

인덱스 1부터 3(4-1)까지의 아이템 슬라이싱

뒤에서 2개의 아이템 슬라이싱

문제 해결

문제 해결 9-5 슬라이싱 연습하기 ch09_sol_05.py

fruits 튜플에서 주어진 요구사항에 맞게 슬라이싱해봅시다.

```
fruits = ('apple', 'banana', 'plum', 'watermelon', 'peach')
```

- 인덱스 2부터 4까지의 아이템을 출력하시오.
- 인덱스 0부터 3까지의 아이템을 출력하시오.
- 인덱스 3부터 끝까지의 아이템을 출력하시오.

```
01   fruits = ('apple', 'banana', 'plum', 'watermelon', 'peach')
02
03   print(fruits[2:5])
04   print(fruits[:4])
05   print(fruits[3:])
```

```
('plum', 'watermelon', 'peach')
('apple', 'banana', 'plum', 'watermelon')
('watermelon', 'peach')
```

리스트와 튜플간 변환

- 튜플 → 리스트로 변환
  - 튜플의 아이템을 수정하기 위해 변환

- 리스트 → 튜플로 변환
  - 리스트로 선언된 데이터를 수정이 안 되게 하기 위해 변환

코드 9-3 ch09_03.py

```
01  colors = ('Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Purple')
02  print('colors의 자료형 :', type(colors))
03
04  colors = list(colors)          튜플을 리스트로 데이터 타입 변환
05  print('colors의 자료형 :', type(colors))
06
07  colors = tuple(colors)         리스트를 튜플로 데이터 타입 변환
08  print('colors의 자료형 :', type(colors))
```

```
colors의 자료형 : <class 'tuple'>
colors의 자료형 : <class 'list'>
colors의 자료형 : <class 'tuple'>
```

아이템 정렬

- 튜플의 아이템 정렬 : sorted() 함수
  - 튜플은 아이템을 수정할 수 없기 때문에 기본적으로 정렬이 불가능함
  - 하지만 내장 함수 sorted()를 이용하면 튜플도 아이템을 정렬할 수 있음
  - **반환되는 데이터는 리스트**라는 것을 주의해야 함

코드 9-4 ch09_04.py

```
01  colors = ('Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Purple')
02  print(colors)
03  print('colors의 자료형 :', type(colors))
04
05  cs = sorted(colors)                    # 오름차순으로 정렬한 리스트를 반환한다.
06  print(cs)
07  print('cs의 자료형 :', type(cs))
```

```
('Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Purple')
colors의 자료형 : <class 'tuple'>  ●——— 정렬 전
['Blue', 'Green', 'Indigo', 'Orange', 'Purple', 'Red', 'Yellow']
cs의 자료형 : <class 'list'>  ●——————— 정렬 후
```

아이템 정렬

sort() 함수

- 정렬된 리스트는 sort() 함수를 이용하여 오름차순과 내림차순 옵션을 적용할 수 있음

코드 9-5 ch09_05.py

```
01  colors = ('Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Purple')
02  cs = sorted(colors)
03
04  cs.sort(reverse = True)          # 내림차순
05  print(cs)
06
07  cs.sort(reverse = False)         # 오름차순
08  print(cs)
```

sorted()와 sort()는 다른 함수입니다!

```
['Yellow', 'Red', 'Purple', 'Orange', 'Indigo', 'Green', 'Blue']  ← 내림차순
['Blue', 'Green', 'Indigo', 'Orange', 'Purple', 'Red', 'Yellow']  ← 오름차순
```

확인문제

확인문제

1. numbers 튜플에 저장된 아이템을 조회하기 위한 구문으로 옳은 것은 무엇인가?

   ① numbers(3)　　② numbers<3>

   ③ numbers[3]　　④ numbers{3}

2. 다음 튜플에서 처음과 마지막에 있는 아이템의 차이를 구하는 코드로 가장 적합한 것은 무엇인가?

```
numbers = (9, 5, 19, 1, 21, 7, 1)
```

   ① print(numbers[0] - numbers[len(numbers)])

   ② print(numbers[0] - numbers[7])

   ③ print(numbers[1] - numbers[len(numbers)])

   ④ print(numbers[0] - numbers[len(numbers)-1])

3. tools 튜플에서 특정 아이템의 인덱스를 확인하는 구문으로 옳은 것은 무엇인가?

   ① tools.index('망치')　　② ntools.index<'망치'>

   ③ tools.index['망치']　　④ tools.index{'망치'}

확인문제

4. 튜플 결합에 대한 설명으로 옳은 것은 무엇인가?

① 튜플은 수정 불가능한 컨테이너 자료형으로 결합이 불가능하다.

② 튜플은 리스트와 마찬가지로 `extend()`를 이용해서 덧붙일 수 있다.

③ 튜플은 '+' 연산자를 이용하면 두 개의 튜플을 결합할 수 있다. 이때 새로운 튜플이 생성된다.

④ 튜플을 결합할 때 `extend()`를 사용하면 에러는 발생하지만 프로그램은 문제없이 실행된다.

5. 다음 프로그램의 실행 결과를 보고 빈칸을 채우시오.

```
numbers = (10, 20, 30, 40, 50, 60)

print(numbers[    :    ])
print(numbers[:    ])
print(numbers[    :4])
print(numbers[    :    (numbers)-1])
```

```
(20, 30)
(10, 20)
(20, 30, 40)
(40, 50)
```

정답

1. ③ **2.** ④ **3.** ① **4.** ③ **5.** 1, 3, 2, 1, 3, len

Section 04
딕셔너리란?

딕셔너리 vs 리스트

- 딕셔너리(dictionary)
  - 리스트와 함께 파이썬 프로그램에서 많이 사용하는 컨테이너 자료형
  - 리스트가 인덱스를 이용하여 아이템을 참조한다면, 딕셔너리는 인덱스 대신 키(key)를 이용하여 아이템을 참조함

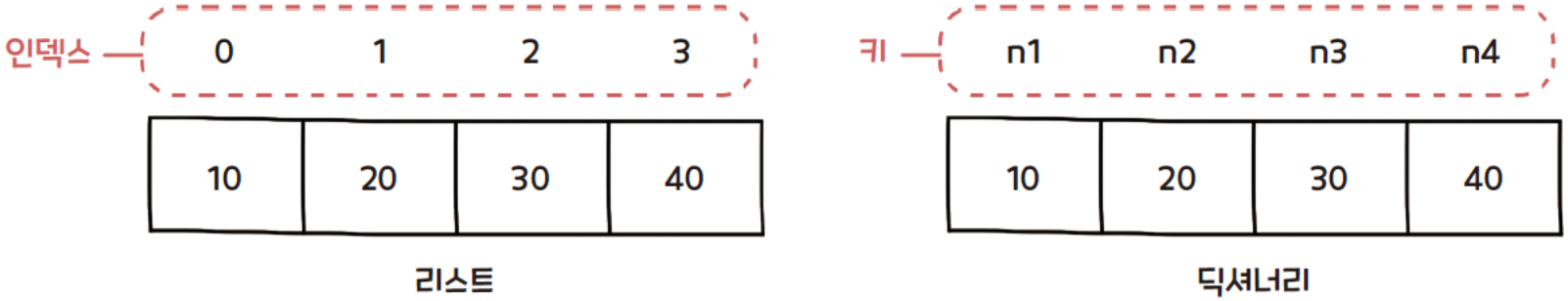


그림 9-4 리스트와 딕셔너리의 차이점

딕셔너리 vs 리스트

- 딕셔너리의 구조
  - 키(key)와 값(value)의 쌍으로 이루어짐
  - 개인 사물함의 구조와 비슷함
  - 개인 사물함의 열쇠가 딕셔너리에서는 키(key)에 해당함
  - 사물함에 넣을 물건이 값(value)에 해당함

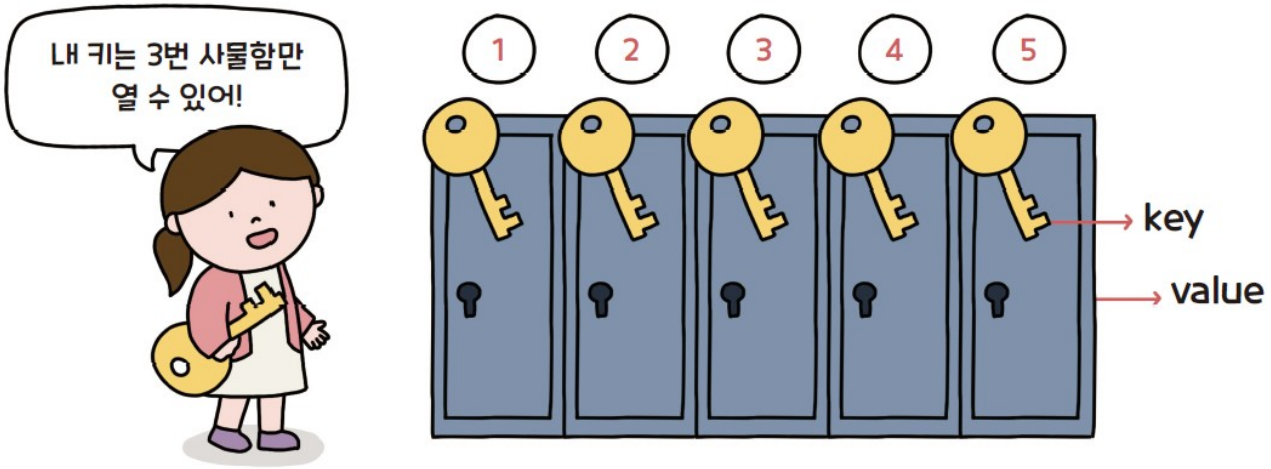


그림 9-5 딕셔너리의 개인 사물함 예시

딕셔너리 선언

딕셔너리 선언

- 중괄호({})를 사용함
- 각 아이템은 '키:값' 형태로 쓰고, 아이템과 아이템은 쉼표(,)로 구분함

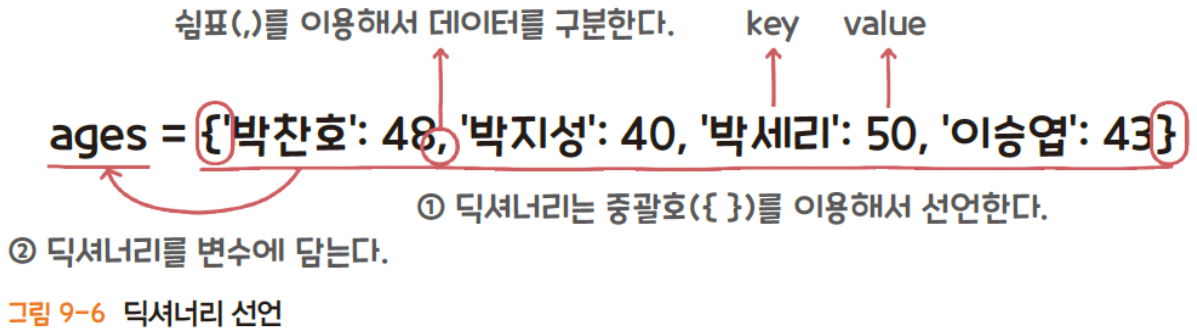


그림 9-6 딕셔너리 선언

- 셸 모드에서 딕셔너리 ages를 선언하고 딕셔너리와 자료형을 출력하기

```
>>> ages = {'박찬호':48, '박지성':40, '박세리':50, '이승엽':43}
>>> ages
{'박찬호': 48, '박지성': 40, '박세리': 50, '이승엽': 43}
>>> type(ages)
<class 'dict'>
```

딕셔너리 선언

아이템 출력

자료형 출력

딕셔너리 선언

딕셔너리의 주의할 점

① 딕셔너리에서 키는 중복할 수 없지만 값은 중복이 가능함

- 사물함의 열쇠를 중복해서 사용할 수 없듯이 딕셔너리에서도 키를 중복할 수 없음

그림 9-7 키와 값의 중복 가능 여부

딕셔너리 선언

- 딕셔너리의 주의할 점

② 키와 값에 사용하는 데이터는 어떤 자료형이든 올 수 있음

- '문자열 : 정수', '정수 : 문자열', '문자열 : 실수', '문자열 : 문자열' 등이 모두 가능함
- '문자열 : 리스트', '문자열 : 튜플', '문자열 : 딕셔너리' 등도 가능함

▪다양한 데이터 타입을 이용해서 딕셔너리를 선언하기

```
>>> dicContainer = {'이름':'홍길동', '나이':25, '주소':'서대문구 연희로2길 62', '취미':
['축구', '수영', '조깅'], '몸무게':85.3}
>>> dicContainer
{'이름': '홍길동', '나이': 25, '주소': '서대문구 연희로2길 62', '취미': ['축구', '수영',
'조깅'], '몸무게': 85.3}
```

확인문제

확인문제

1. 딕셔너리에 대한 설명으로 옳은 것은 무엇인가?

   ① 딕셔너리를 선언할 때는 소괄호(())를 이용한다.

   ② 딕셔너리의 아이템은 '키:값' 쌍으로 이루어져 있다.

   ③ 딕셔너리는 다른 컨테이너 자료형에 비해서 조회 속도가 느려 자주 사용하지 않는다.

   ④ 리스트와 마찬가지로 딕셔너리에서도 인덱스를 사용한다.

2. 다음은 홍길동 학생이 중간고사에서 받은 성적을 나타낸 것이다. 이 데이터 집합을 딕셔너리를 이용하여 선언하시오.

과목명	C/C++	Java	네트워킹	보안	해킹	시스템
점수	A	B+	C	A+	F	C+

정답

1. ②

2.
```
scores = {'C/C++':'A', 'Java':'B+', '네트워킹':'C', '보안':'A+', '해킹':'F',
          '시스템':'C+'}
```

Section 05
딕셔너리 조회/삽입/수정/삭제하기

딕셔너리 조회

- 딕셔너리의 조회
  - 특정 아이템을 조회할 때 키를 이용함
  - 리스트에서 인덱스를 사용하는 것과 같음

  - 홍길동의 개인 정보를 저장하는 dicContainer 딕셔너리를 선언하고 나이와 취미를 조회하기

```
>>> dicContainer = {'이름':'홍길동', '나이':25, '주소':'서대문구 연희로', '취미':['축구',
'수영', '조깅'], '몸무게':85.3}
>>> dicContainer['나이']
25
>>> dicContainer['취미']
['축구', '수영', '조깅']
```

딕셔너리 선언

'나이'에 해당하는 아이템 조회

'취미'에 해당하는 아이템 조회

아이템 삽입

- 딕셔너리의 아이템 삽입
  - '딕셔너리이름[키]=값' 형태로 작성
  - 이때 새로 추가하는 키는 기존에 있던 키와 중복되어서는 안 됨
  - 만약 키가 중복되면 기존 키에 저장되어 있는 값이 바뀌므로 주의해야 함

  - 홍길동의 혈액형이 'O'형일 때 dicContainer 딕셔너리에 혈액형 정보를 삽입하기

```
>>> dicContainer['혈액형'] = 'O'
>>> dicContainer
{'이름': '홍길동', '나이': 25, '주소': '서대문구 연희로', '취미': ['축구', '수영', '조깅'], '몸무게': 85.3, '혈액형': 'O'}
```

아이템 삽입

아이템 수정

- 딕셔너리의 아이템 수정
  - 특정 아이템을 수정할 때는 삽입할 때와 같이 수정하려는 아이템의 키와 값을 씀
  - 홍길동의 몸무게를 90으로, 나이를 한 살 더 많게 수정하기

```
>>> dicContainer['몸무게'] = 90
>>> dicContainer['나이'] = dicContainer['나이'] + 1
>>> dicContainer
{'이름': '홍길동', '나이': 26, '주소': '서대문구 연희로', '취미': ['축구', '수영', '조깅'], '몸무게': 90, '혈액형': 'O'}
```

'몸무게'에 해당하는 아이템 수정

'나이'에 해당하는 아이템 수정

아이템 삭제

딕셔너리의 아이템 삭제

- del 키워드를 이용하면 딕셔너리의 특정 아이템을 삭제할 수 있음

- 홍길동의 '몸무게' 정보를 삭제하기

```
>>> del dicContainer['몸무게']
>>> dicContainer
{'이름': '홍길동', '나이': 26, '주소': '서대문구 연희로', '취미': ['축구', '수영', '조깅'], '혈액형': 'O'}
```

'몸무게'에 해당하는 아이템 삭제

딕셔너리의 아이템 개수 조회

- len() 함수를 이용하면 전체 아이템의 개수를 조회할 수 있음

```
>>> len(dicContainer)
5
```

딕셔너리 길이(아이템 개수) 조회

전체 키와 값 조회

- 딕셔너리에서 전체 키와 값을 조회
  - keys()와 values() 함수를 이용

```
>>> dicContainer
{'이름': '홍길동', '나이': 26, '주소': '서대문구 연희로', '취미': ['축구', '수영', '조
깅'], '혈액형': 'O'}
>>> dicContainer.keys()
dict_keys(['이름', '나이', '주소', '취미', '혈액형'])
>>> dicContainer.values()
dict_values(['홍길동', 26, '서대문구 연희로', ['축구', '수영', '조깅'], 'O'])
```

전체 키 조회

전체 값 조회

- ke
  출

```
>>> dicContainer
{'이름': '홍길동', '나이': 26, '주소': '서대문구 연희로', '취미': ['축구', '수영', '조
깅'], '혈액형': 'O'}
>>> for key in dicContainer.keys():
        print(key, '\t: ', dicContainer[key])

이름 :  홍길동
나이 :  26
주소 :  서대문구 연희로2길 62
취미 :  ['축구', '수영', '조깅']
혈액형 :  O
```

키에 해당하는 값 출력

for문을 끝내고 출력 결과를 보려면
한 번 더 Enter를 눌러주세요!

전체 키와 값 조회

딕셔너리에서 전체 키와 값을 조회

- keys() 함수로 조회한 키와 for문을 이용하여 각 키에 해당하는 값을 출력하기

```
>>> dicContainer
{'이름': '홍길동', '나이': 26, '주소': '서대문구 연희로', '취미': ['축구', '수영', '조깅'], '혈액형': 'O'}
>>> for key in dicContainer.keys():
        print(key, '\t: ', dicContainer[key])

이름 :  홍길동
나이 :  26
주소 :  서대문구 연희로2길 62
취미 :  ['축구', '수영', '조깅']
혈액형 :  O
```

키에 해당하는 값 출력

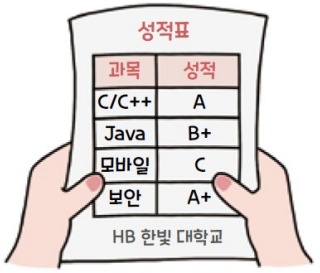
for문을 끝내고 출력 결과를 보려면
한 번 더 Enter를 눌러주세요!

문제 해결

문제 해결 9-6 중간고사 성적 관리 프로그램 만들기 ch09_sol_06.py

아래 시나리오를 기반으로 딕셔너리를 이용해서 중간고사 성적 관리 프로그램을 만들어봅시다.

- #1 : 중간고사의 성적(C/C++은 A, Java는 B+, 모바일은 C, 보안은 A+, 해킹은 F, 시스템은 C+)을 저장하는 딕셔너리를 만든다.
- #2 : 'Java'와 '시스템' 과목의 성적을 조회한다.
- #3 : 추가로 2과목의 성적(파이썬은 A, OS는 A+)을 삽입한다.
- #4 : 'Java'와 '시스템'의 성적을 각각 'F'와 'A'로 수정한다.
- #5 : 전체 과목과 성적을 조회하여 최종 성적표를 출력한다.



문제 해결

```
01  scores = {'C/C++':'A', 'Java':'B+', '모바일':'C', '보안':'A+', '해킹':'F',
02            '시스템':'C+'}
03
04  print('#시나리오1 ')
05  print(scores)
06
07  print('#시나리오2')
08  print('Java : ', scores['Java'])
09  print('시스템 : ', scores['시스템'])
10
11  print('#시나리오3 ')
12  scores['파이썬'] = 'A'
13  scores['OS'] = 'A+'
14  print(scores)
15
16  print('#시나리오4')
17  scores['Java'] = 'F'
18  scores['시스템'] = 'A'
19  print(scores)
20
21  print('#시나리오5')
22  for key in scores.keys():
23      print(key, '\t: ', scores[key])
```

문제 해결

```
#시나리오1
{'C/C++': 'A', 'Java': 'B+', '모바일': 'C', '보안': 'A+', '해킹': 'F', '시스템': 'C+'}
#시나리오2
Java :  B+
시스템 :  C+
#시나리오3
{'C/C++': 'A', 'Java': 'B+', '모바일': 'C', '보안': 'A+', '해킹': 'F', '시스템': 'C+', '파
이썬': 'A', 'OS': 'A+'}
#시나리오4
{'C/C++': 'A', 'Java': 'F', '모바일': 'C', '보안': 'A+', '해킹': 'F', '시스템': 'A', '파
이썬': 'A', 'OS': 'A+'}
#시나리오5
C/C++  :  A
Java   :  F
모바일 :  C
보안   :  A+
해킹   :  F
시스템 :  A
파이썬 :  A
OS     :  A+
```

확인문제

확인문제

1. dics 딕셔너리에서 아이템을 조회하는 구문을 옳은 것은 무엇인가?

① dics('major')　　② dics['major']

③ dics[]='major'　　④ dics{'major'}

2. 다음은 학용품 재고를 관리하는 프로그램이다. 새로운 학용품으로 줄자 20개가 입고되었을 때 올바른 코딩은 무엇인가?

```
goods = {'연필':10, '지우개':20, '색연필':15}
```

① goods['줄자'] = 20　　② goods['줄자':20]

③ goods['줄자' = 20]　　④ goods['줄자',20]

확인문제

3. 다음은 강아지 정보를 나타내는 프로그램이다. 해가 바뀌어서 나이가 한 살 증가할 때의 코드로 옳은 것을 모두 고르시오.

```
dogInfos = {'이름':10, '성별':20, '나이':15}
```

① dogInfos['나이'] = 1

② dogInfos['나이'] = 15 + 1

③ dogInfos['나이'] += 1

④ dogInfos['나이'] = dogInfos['나이'] + 1

4. 다음은 쇼핑몰 회원 정보를 나타내는 프로그램이다. 개인정보 중 휴대폰 번호를 삭제하는 코드로 옳은 것은 무엇인가?

```
memberInfos = {'이름':'홍길동', '메일':'abc@gmail.com', '휴대폰':'010-1234-5678'}
```

① del memberInfos('휴대폰')

② del memberInfos['휴대폰']

③ def memberInfos['휴대폰']

④ del memberInfos{'휴대폰'}

정답

1. ② 2. ① 3. ③, ④ 4. ②

Section 06
터틀 프로그래밍

정삼각형부터 정육각형까지 그리기

- 1. 먼저 각 도형의 내각의 합을 구함
- n각형 내각의 합은 180×(n-2)으로 구할 수 있음
  - 삼각형은 180°, 사각형은 360°, 오각형은 540°, 육각형은 720°라는 점을 참고

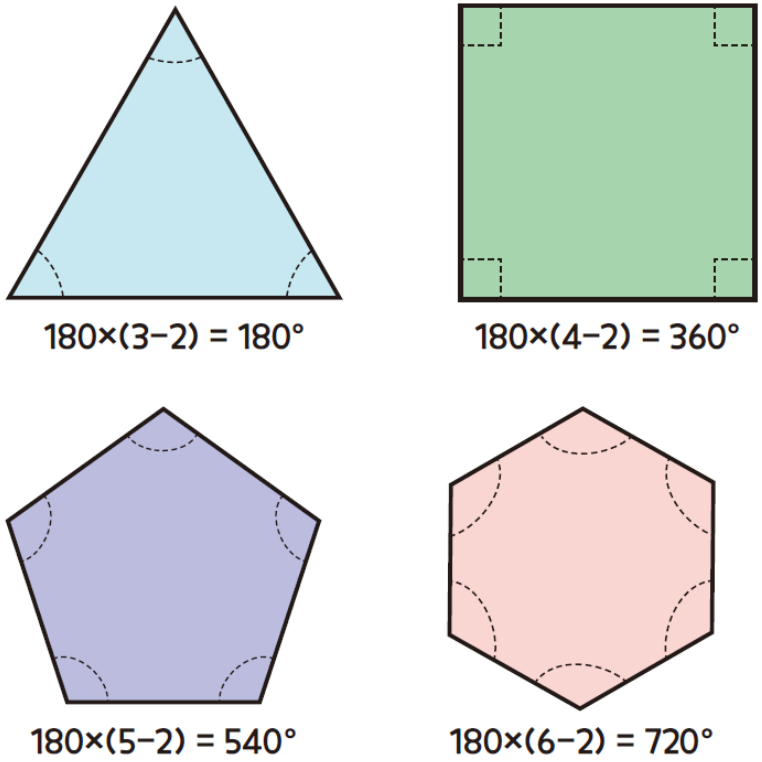


그림 9-8 n각형 내각의 합

- 정삼각형부터 정육각형까지 그리기
  - 2. 내각을 구함
  - n각형의 내각은 (내각의 합÷n)으로 구할 수 있음
    - 삼각형은 60°, 사각형은 90°, 오각형은 108°, 육각형은 120°

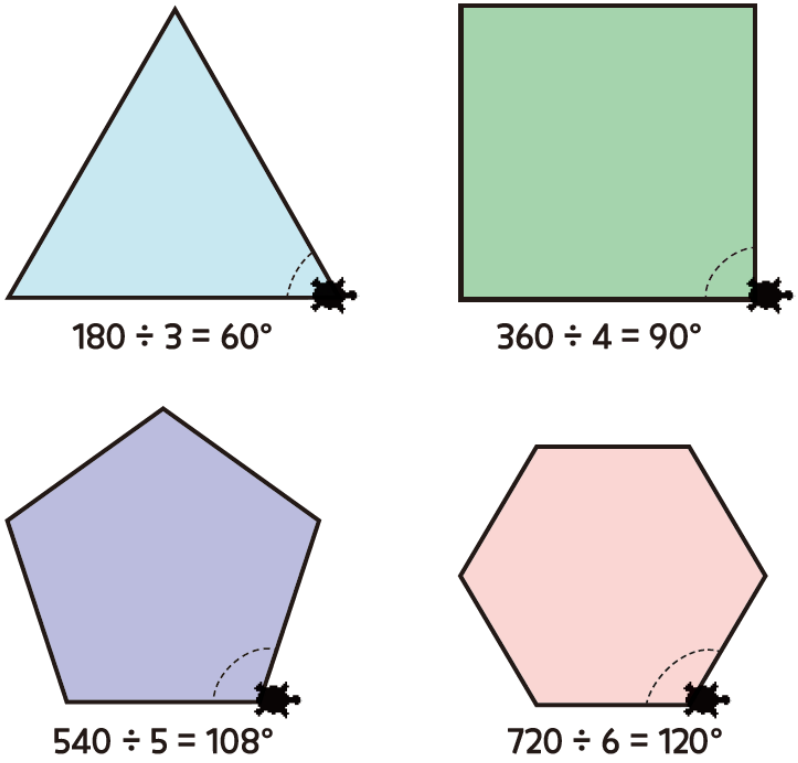


그림 9-9 n각형 내각

- 정삼각형부터 정육각형까지 그리기
  - 3. 터틀이 도화지에서 n각형을 그리기 위해 회전해야 할 각도를 구함
  - 회전 각도는 (180-내각)으로 구할 수 있음
    - 삼각형은 120°, 사각형은 90°, 오각형은 72°, 육각형은 60°

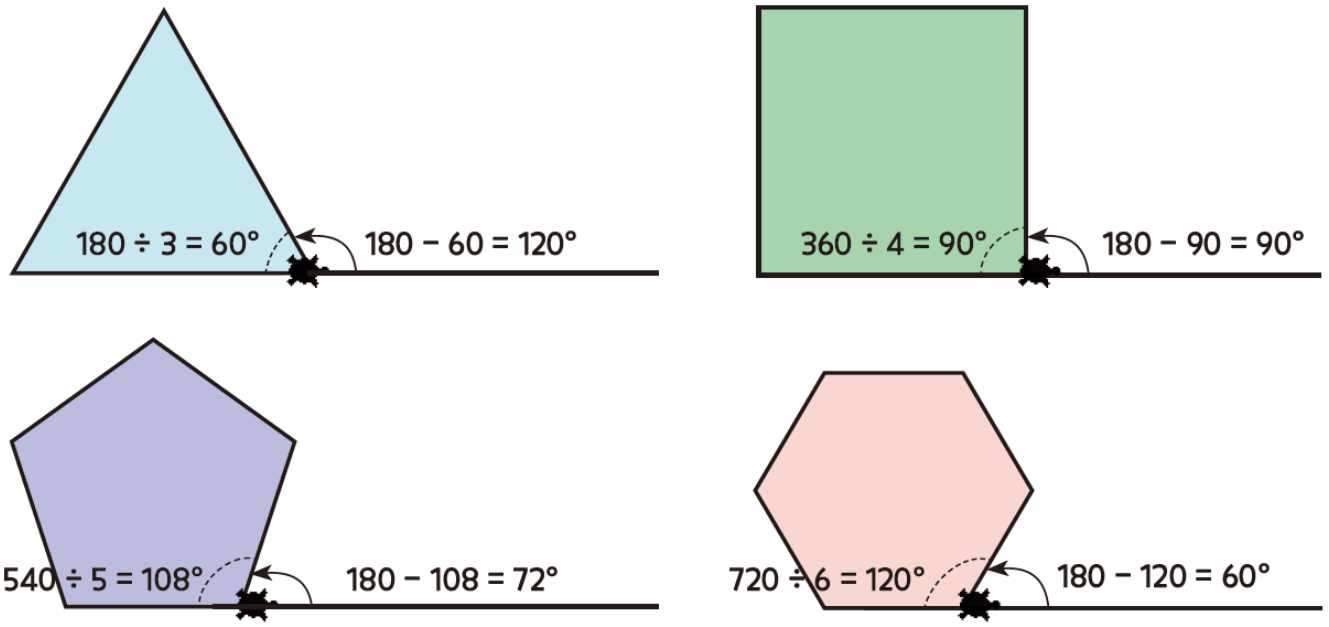


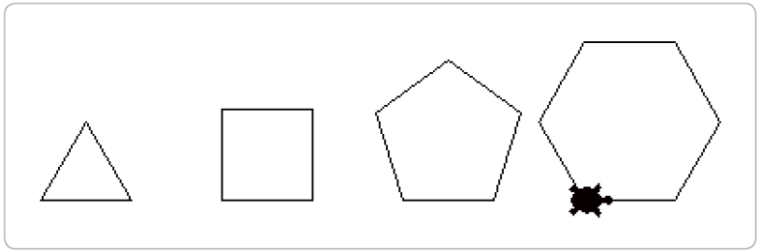
그림 9-10 터틀의 회전 각도

터틀 프로그래밍

- 정삼각형부터 정육각형까지 그리기
  - [그림 9-10]의 터틀 회전 각도를 참고하여 컨테이너 자료형에 좌표와 회전 각도를 설정한 다음 차례대로 도형을 그리기

코드 9-6 ch09_06.py

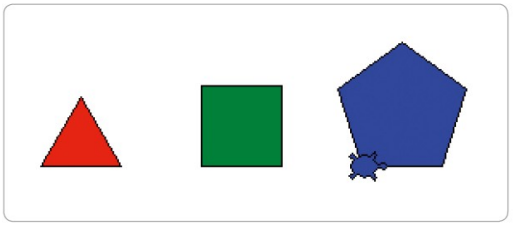
```
01  import turtle
02  t= turtle.Turtle()
03  t.shape('turtle')
04
05  # x, y좌표 및 회전각도 데이터
06  datas = [(0, 0, 120, 120, 120), (100, 0, 90, 90, 90, 90), (200, 0, 72, 72,
07            72, 72, 72), (300, 0, 60, 60, 60, 60, 60, 60)]
08
09  for points in datas:
10      t.up()
11      t.goto(points[0], points[1])        # x, y좌표 설정
12      t.down()
13
14      for num in range(2, len(points)):   # 회전하면서 그리기
15          t.forward(50)
16          t.left(points[num])
```



확인문제

확인문제

컨테이너 자료형과 터틀을 이용하여 삼각형부터 오각형까지 연속해서 그리고 색칠하시오.



확인문제

정답

```
import turtle
t= turtle.Turtle()
t.shape('turtle')

datas = [(0, 0, 'red', 120, 120, 120), (100, 0, 'green', 90, 90, 90, 90),
(200, 0, 'blue', 72, 72, 72, 72, 72)]

for points in datas:
    t.up()
    t.goto(points[0], points[1])
    t.down()
    t.fillcolor(points[2])      # 색상 설정
    t.begin_fill()              # 채색 시작

    for num in range(3, len(points)):
        t.forward(50)
        t.left(points[num])

    t.end_fill()                # 채색 끝
```

실전 예제

실전 예제1 – 수학시험 프로그램

문제

다음은 수학시험 문제 및 정답입니다. 튜플에 문제를 저장하고 사용자가 답을 입력하면 채점하는 프로그램을 만들어 봅시다.

문제	정답	점수
3+2	5	3점
5÷2의 몫	2	5점
10-2	8	3점
10^2×2	200	5점
1-(10÷4의 나머지)	-1	5점
2^4	16	3점
4÷2	2	3점

```
문제 :  3+2
정답을 입력하세요. 5
문제 :  5/2의 몫
정답을 입력하세요. 2
...생략...
문제 :  4/2
정답을 입력하세요. 2
-----------------------------
정답 개수:  7
오답 개수:  0
Total Score:  27
 -----------------------------
```

실전 예제1 – 수학시험 프로그램

해결

ch09_appEx_01.py

```
01  quiz = (['3+2', 5, 3], ['5/2의 몫', 2, 5], ['10-2', 8, 3], ['10^2*2', 200, 5],
            ['1-(10/4의 나머지)', -1, 5], ['2^4', 16, 3], ['4/2', 2, 3])
02
03  answerCount = 0
04  wrongAnswerCount = 0
05  totalScore = 0
06
07  for item in quiz:
08      print('문제 : ', item[0])
09      answer = int(input('정답을 입력하세요. '))
10
11      if answer == item[1]:
12          answerCount += 1
13          totalScore += item[2]
14      else:
15          wrongAnswerCount += 1
16
17  print('-'*30)
18  print('정답 개수\t: ', answerCount)
19  print('오답 개수\t: ', wrongAnswerCount)
20  print('Total Score\t: ', totalScore)
21  print('-'*30)
```

실전 예제2 – 회원가입 프로그램

문제

다음 4단계를 따라 회원가입 프로그램을 만들어봅시다.

1. 프로그램을 실행하면 '1. 회원가입', '2. 프로그램 종료'를 출력하고, 사용자에게 선택을 유도한다.
2. 사용자가 '1'을 입력하면 '회원가입'을 진행하고, '2'를 입력하면 프로그램을 종료한다. 프로그램이 종료되면 전체 회원의 아이디와 비밀번호를 출력한다.
3. '회원가입'은 아이디와 비밀번호를 입력받아 딕셔너리에 저장한다. 이때 아이디는 키(key), 비밀번호는 값(values)으로 한다.
4. 회원가입이 끝나면 다시 1단계로 돌아가 '1. 회원가입', '2. 프로그램 종료'를 출력하고, 사용자에게 선택을 유도한다.


```
1. 회원가입, 2. 프로그램 종료   1
아이디를 입력하세요.    superman2020
비밀번호를 입력하세요.  super2020

1. 회원가입, 2. 프로그램 종료  1
아이디를 입력하세요.    betman2021
비밀번호를 입력하세요.  bet2021

1. 회원가입, 2. 프로그램 종료  2
------------------------------
아이디 : 비밀번호
------------------------------
superman2020 :   super2020
batman2021   :   bet2021
------------------------------
```

실전 예제2 – 회원가입 프로그램

해결

ch09_appEx_02.py

```
01  members = {}
02  flag = True
03
04  while flag:
05      selectNum = int(input('\n1. 회원가입, 2. 프로그램 종료\t'))
06
07      if selectNum == 1:
08          id = input('아이디를 입력하세요.\t')
09          pw = input('비밀번호를 입력하세요.\t')
10          members[id] = pw
11
12      elif selectNum == 2:
13          print('-' * 30)
14          print('아이디 : 비밀번호')
15          print('-' * 30)
16
17          for key in members.keys():
18              print(key, '\t:\t', members[key])
19          print('-' * 30)
20
21          flag = False
```

Thank you!

한빛아카데미 Hanbit Academy, Inc.