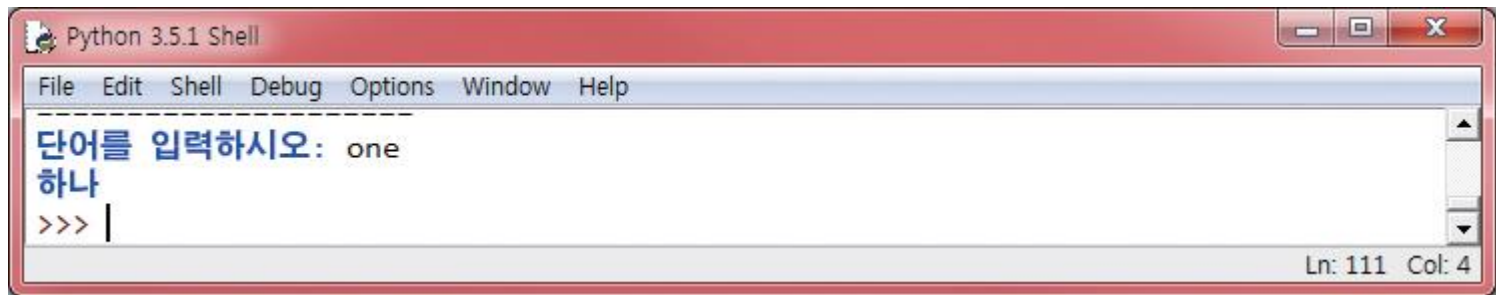




9장 리스트와 딕셔너리

이번 장에서 만들 프로그램

(1) 영한 사전 만들기

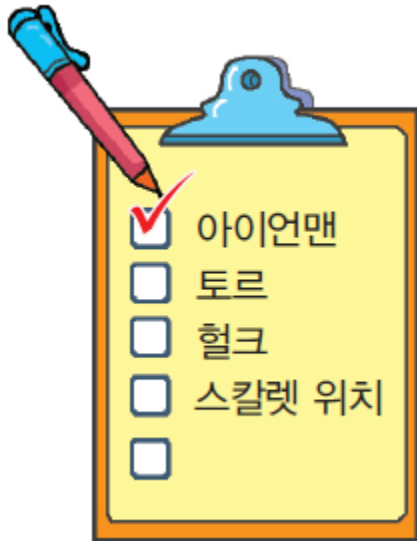


(2) 오류기를 그려보자. 오류기에 대한 정보를 리스트에 저장한다.



리스트

- 어떤 경우에는 여러 개의 데이터를 하나로 묶어서 저장하는 것이 필요하다.



```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]
```

공백 리스트에서 추가하기

```
>>> heroes = [ ]  
>>> heroes.append("아이언맨")  
['아이언맨']  
>>> heroes.append("닥터 스트레인지")  
>>> print(heroes)  
['아이언맨', '닥터 스트레인지']  
>>>
```



점의 의미


- 파이썬에서 모든 것은 객체(object)이다. 객체는 관련되는 변수와 함수를 묶은 것이다.
- 파이썬에서 리스트도 객체이다. 객체 안에 있는 무엇인가를 사용할 때는 객체의 이름을 쓰고 점(.)을 붙인 후에 함수의 이름을 적는다.

```
heroes.append("아이언맨")
```

리스트 항목 접근하기

```
>>> letters = ['A', 'B', 'C', 'D', 'E', 'F']
```

'A'	'B'	'C'	'D'	'E'	'F'
0	1	2	3	4	5

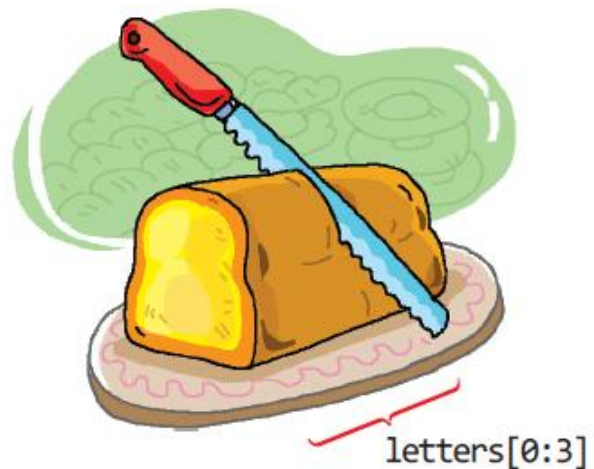


```
>>> print(letters[0])  
A  
>>> print(letters[1])  
B  
>>> print(letters[2])  
C
```

슬라이싱

- 슬라이싱(slicing)은 리스트에서 한 번에 여러 개의 항목을 추출하는 기법이다.

```
>>> letters = ['A', 'B', 'C', 'D', 'E', 'F']  
>>> print(letters[0:3])  
['A', 'B', 'C']
```



인덱스 생략

```
>>> print(letters[:3])  
['A', 'B', 'C']
```

```
>>> print(letters[3:])  
['D', 'E', 'F']
```

```
>>> print(letters[:])  
['A', 'B', 'C', 'D', 'E', 'F']
```

리스트를 복사할 때 사용한다.

리스트 항목 변경하기

```
>>> heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
>>> heroes[1] = "닥터 스트레인지"  
>>> print(heroes)  
['아이언맨', '닥터 스트레인지', '헐크', '스칼렛 위치']  
>>>
```

인덱스를 이용한다.

함수를 이용하여 추가하기

```
>>> heroes.append("스파이더맨")
>>> print(heroes)
['아이언맨', '닥터 스트레인지', '헐크', '스칼렛 위치', '스파이더맨']

>>> heroes.insert(1, "배트맨")
>>> print(heroes)
['아이언맨', '배트맨', '닥터 스트레인지', '헐크', '스칼렛 위치', '스파이더맨']
>>>
```

항목 삭제하기

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치"]  
heroes.remove("스칼렛 위치")  
print(heroes)
```

```
['아이언맨', '토르', '헐크']
```

항목이 리스트 안에 있는지 체크

```
if "슈퍼맨" in heroes:  
    heroes.remove("슈퍼맨")
```

del

- del는 인덱스를 사용하여 항목을 삭제한다.

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
del heroes[0]  
print(heroes)
```

```
['토르', '헐크', '스칼렛 위치']
```

pop()

- pop()은 리스트에서 마지막 항목을 삭제

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
last_hero = heroes.pop()  
print(last_hero)
```

스칼렛 위치

리스트 탐색하기

□ index() 사용

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
print(heroes.index("헐크"))
```

2

리스트 방문하기

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
for hero in heroes:  
    print(hero)
```

아이언맨
토르
헐크
스칼렛 위치

리스트 정렬하기

```
heroes = [ "아이언맨", "토르", "헐크", "스칼렛 위치" ]  
heroes.sort()  
print(heroes)
```

['스칼렛 위치', '아이언맨', '토르', '헐크']



Lab: 오늘의 속담



- 리스트에 여러 개의 속담을 저장한 후에 속담 중에서 하나를 랜덤하게 골라서 오늘의 속담으로 제공한다

#####

오늘의 속담

#####

고생 없이 얻을 수 있는 진실로 귀중한 것은 하나도 없다.



사람은 사랑할 때
누구나 시인이 된다

시작이 반이다

꿈을 지녀라. 그러면
어려운 현실을 이길 수 있다

고생 없이 얻을 수 있는
진실로 귀중한 것은 하나도 없다

Solution

```
import random

quotes = []
quotes.append("꿈을 지녀라. 그러면 어려운 현실을 이길 수 있다.")
quotes.append("분노는 바보들의 가슴속에서만 살아간다..")
quotes.append("고생 없이 얻을 수 있는 진실로 귀중한 것은 하나도 없다.")
quotes.append("사람은 사랑할 때 누구나 시인이 된다.")
quotes.append("시작이 반이다.")

dailyQuote = random.choice(quotes)
print("#####")
print("# 오늘의 속담 #")
print("#####")
print("")
print(dailyQuote)
```

Lab: 오류기 그리기



- 반복 구조를 사용하여 화면에 오류기를 그려 보자. 오류기의 색상과 위치를 리스트에 저장한다.



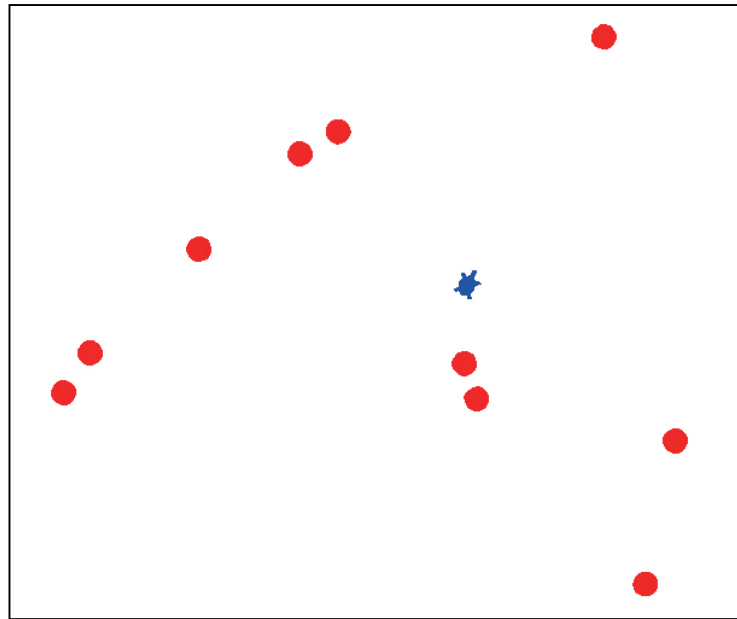
Solution

```
import turtle

def draw_olympic_symbol():
    positions = [[0, 0, "blue"], [-120, 0, "purple"], [60,60, "red"],
                 [-60, 60, "yellow"], [-180, 60, "green"]]
    for x, y, c in positions:
        t.penup()
        t.goto(x, y)
        t.pendown()
        t.color(c, c)
        t.begin_fill()
        t.circle(30)
        t.end_fill()
    t = turtle.Turtle()
    draw_olympic_symbol()
```

Lab: 애스터로이드 게임 업그레이드

- 8장에서 작성하였던 애스터로이드 게임 기억나는지 모르겠다. 8장에서는 소행성이 2개만 생성되었다. 만약 소행성이 10개 정도 있어야 한다면 리스트에 저장하여야 한다.



```
import turtle
import random
import math
```

```
player = turtle.Turtle()
player.color("blue")
player.shape("turtle")
player.penup()
player.speed(0)
screen = player.getscreen()
```

```
asteroids = []
for i in range(10):
    a1 = turtle.Turtle()
    a1.color("red")
    a1.shape("circle")
    a1.penup()
    a1.speed(0)
    a1.goto(random.randint(-300, 300), random.randint(-300, 300))
    asteroids.append(a1)    # 생성된 터틀을 리스트에 추가한다.
```

공백 리스트를 생성한다.
10개의 터틀을 생성한다.

Solution

```
def turnleft():
    player.left(30) # 왼쪽으로 30도 회전한다.

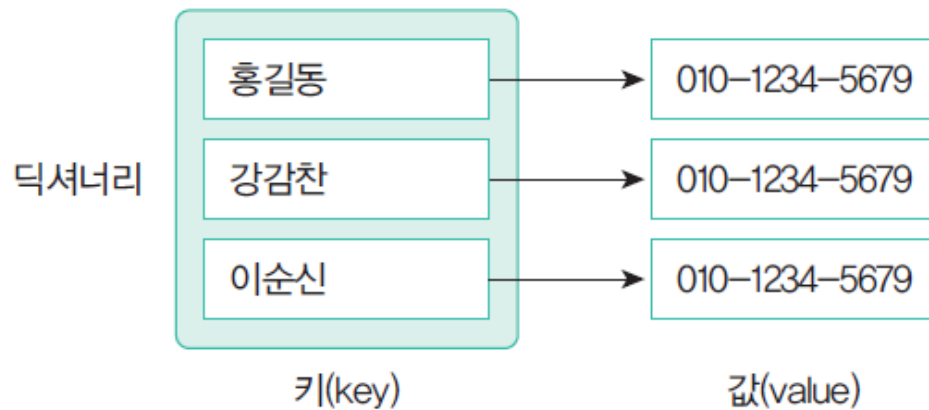
def turnright():
    player.right(30) # 오른쪽으로 30도 회전한다.

screen.onkeypress(turnleft, "Left")
screen.onkeypress(turnright, "Right")
screen.listen()

def play():
    player.forward(2) # 2픽셀 전진
    for a in asteroids: # 리스트에 저장된 모든 터틀에 대하여
        a.right(random.randint(-180, 180))
        a.forward(2)
    screen.ontimer(play, 10) # 10ms가 지나면 play()를 다시 호출
screen.ontimer(play, 10)
```


딕셔너리

- 딕셔너리(dictionary)도 리스트와 같이 값을 저장하는 방법이다. 하지만 딕셔너리에는 값(value)과 관련된 키(key)가 있다.



딕셔너리

```
>>> phone_book = { }  
>>> phone_book["홍길동"] = "010-1234-5678"  
  
>>> print(phone_book)  
{'홍길동': '010-1234-5678'}
```

```
>>> phone_book = {"홍길동": "010-1234-5678"}  
>>> phone_book["강감찬"] = "010-1234-5679"  
>>> phone_book["이순신"] = "010-1234-5680"  
  
>>> print(phone_book)  
{'이순신': '010-1234-5680', '홍길동': '010-1234-5678', '강감찬': '010-1234-5679'}
```

딕셔너리에서 탐색

- 키를 가지고 값을 찾는다.

```
>>> print(phone_book["강감찬"])  
010-1234-5679
```

딕셔너리의 모든 키 출력하기

```
>>> phone_book.keys()  
dict_keys(['이순신', '홍길동', '강감찬'])
```

```
>>> phone_book.values()  
dict_values(['010-1234-5680', '010-1234-5678', '010-1234-5679'])
```

예제

- 한 학생에 대한 정보를 딕셔너리로 저장하기

```
dict = {'Name': '홍길동', 'Age': 7, 'Class': '초급'}  
print (dict['Name'])  
print (dict['Age'])
```

홍길동
7

딕셔너리 항목 방문

```
>>> for key in sorted(phone_book.keys()):  
    print(key, phone_book[key])
```

강감찬 010-1234-5679

이순신 010-1234-5680

홍길동 010-1234-5678

Lab: 편의점 재고 관리



- 편의점에서 재고 관리를 수행하는 프로그램을 작성해보자. 편의점에서 판매하는 물건의 재고를 딕셔너리에 저장한다.

물건의 이름을 입력하시오: 콜라

4



Solution

```
items = { "커피음료": 7, "펜": 3, "종이컵": 2, "우유": 1,  
          "콜라": 4, "책": 5 }
```

```
item = input("물건의 이름을 입력하시오: ");  
print (items[item])
```



도전문제

위의 프로그램을 편의점의 재고를 관리하는 프로그램으로 업그레이드해보자. 즉 재고를 증가, 또는 감소시킬 수도 있도록 코드를 추가해보자. 간단한 메뉴도 만들어보자.

Lab: 영한사전



- 영한사전을 구현해보자. 영어 단어를 키로 하고 설명을 값으로 하여 저장하면 될 것이다

단어를 입력하시오: one

하나

단어를 입력하시오: two

둘



Solution

```
english_dict = dict()

english_dict['one'] = '하나'
english_dict['two'] = '둘'
english_dict['three'] = '셋'

word = input("단어를 입력하시오: ");
print (english_dict[word])
```



도전문제

영한사전이 아닌 한영사전을 만들려면 어떻게 해야 하는가?

이번 장에서 배운 것

- 리스트는 항목들을 모아둔 곳이다.
- 리스트의 항목은 어떤 것이든 가능하다.
- 공백 리스트를 만들고 `append()`를 호출하여서 코드로 항목을 추가할 수 있다.
- 딕셔너리는 키와 값으로 이루어진다.
- 딕셔너리에 키를 제시하면 값을 반환한다.



Q & A

