Chapter5: 함수와 파이썬 코드 작성연습

Function Concept 1

함수 : 어떤 일을 수행하는 코드의 덩어리
-반복적인 수행을 1회만 작성 후 호출
-코드를 논리적인 단위로 분리
-캡슐화 : 인터페이스만 알면 타인의 코드 사용

함수 이름, parameter, return value(optional)

```
def 함수 이름 (parmaeter #1 ….):
    수행문 #1(statements)
    수행문 #2(statements)
    return <반환값>
```

Parameter : 함수의 입력 값 인터페이스 ex) x, y
Argument : 실제 parameter에 대입된 값

Parameter 유무, 반환 값(return value)
유무에 따라 함수의 형태가 다름

	Parameter 없음	Parameter 존재
반환 값 없음	함수 내의 수행문만 수행	인자를 사용, 수행문만 수행
반환 값 존재	인자없이, 수행문 수행 후 결과값 반환	인자를 사용하여 수행문 수행 후 결과값 반환

함수 형태 예제

```
def a_rectangle_area():# 인자 x , 리턴 값 x
    print (5 * 7)
def b_rectangle_area(x,y): # 인자 o , 리턴 값 x
    print (x * y)
def c_rectangle_area():    # 인자 x , 리턴 값 o
    return (5 * 7)
def d_rectangle_area(x ,y):# 인자 o , 리턴 값 o
    return (x * y)
```

Cf) sort: 자체를 정렬한다. 리턴 값은 없다. sorted : 메모리를 복사해서 정렬

Function Concept 2

함수 호출 방식 개요

-Call by value : 함수에 인자를 넘길 때 값만 넘김. 함수 내에 인자 값 변경 시, 호출자에게 영향을 주지 않음

-Call by Reference : 함수의 인자를 넘길 때 메모리 주소를 넘김. 함수 내에 인자 값 변경 시, 호출자의 값도 변경됨

파이썬 함수 호출 방식

-객체의 주소가 함수로 전달되는 방식: 새로운 객체를 만들 경우 호출자에게 영향을 주지 않음, 전달된 객체를 참조하여 변경 시 호출자에게 영향을 준다.

변수의 범위(Scoping Rule)

-지역변수(local variable) : 함수내에서만 사용

-전역변수(Global variable) : 프로그램전체에서 사용

```
def f():
    s = "I love London!"
    print(s)

s = "I love Paris!"
f()
print(s)
```

I love London!

I love Paris!

➔ 처음 s는 함수 내에서만 쓰인다

```
def f():
    global s
    s = "I love London!"
    print(s)

s = "I love Paris!"
f()
print(s)
```

I love London!

I love London!

➔ 글로벌 s로 쓰이기 때문에 둘다, I love London! 산출

Swap : 함수를 통해 변수 간의 값을 교환하는 함수

재귀함수(Recursive Function)

-자기자신을 호출하는 함수

-점화식과 같은 재귀적 수학 모형을 표현할 때 사용

Function arguments (function에서 input으로 들어가는 변수를 설정**)**

Passing arguments

-함수에 입력되는 arguments는 다양한 형태를 가짐

1) Keyword arguments

- 함수에 입력되는 parameter의 변수명을 사용, arguments를 넘김

```
def print_somthing(my_name, your_name):
    print("Hello {0}, My name is {1}".format(your_name, my_name))

print_somthing("Sungchul", "TEAMLAB")
print_somthing(your_name="TEAMLAB", my_name="Sungchul")
```

Labeling이 되어있지 않으면 순서대로 인식한다. My_name – Sungchul, your_name - TeamLAB

2) Default arguments

- parameter의 기본 값을 사용, 입력하지 않을 경우 기본값 출력

```
def print_somthing_2(my_name, your_name="TEAMLAB"):
    print("Hello {0}, My name is {1}".format(your_name, my_name))

print_somthing_2("Sungchul", "TEAMLAB")
print_somthing_2("Sungchul")
```

3) Variable-length arguments

-개수가 정해지지 않은 변수를 함수의 parameter로 사용하는 법

-Asterisk(*) 기호를 사용하여 함수의 parameter를 표시함

-입력된 값은 **tuple type**으로 사용할 수 있음

가변인자 (Variable-length)

- **가변인자는 일반적으로 *args를 변수명으로 사용**
- **기존 parameter 이후에 나오는 값을 tuple로 저장함**

```
def asterisk_test(a, b, *args):
    return a+b+sum(args)

print(asterisk_test(1, 2, 3, 4, 5))
```

-tuple 형태로 출력되는 가변인자는 함수 선언 시 맨 뒤쪽에 나타나야 한다.

키워드 가변인자 (Keyword variable-length)

- Parameter 이름을 따로 지정하지 않고 입력하는 방법
- Asterisk(*) 두개를 사용하여 함수의 parameter를 표시함
- 입력된 값은 dict type으로 사용할 수 있음
- 가변인자는 오직 한 개만 기존 가변인자 다음에 사용

```
def kwargs_test_3(one,two, *args, **kwargs):
    print(one+two+sum(args))
    print(kwargs)

kwargs_test_3(3,4,5,6,7,8,9, first=3, second=4, third=5)
```

➔ 결과값 : 42 , {'first' : 3, 'third' : 5, 'second': 4}

코딩 컨벤션과 함수 작성법(참고사항)

코딩 컨벤션 : 사람의 이해를 돕기 위한 규칙, 읽기 좋은 코드가 좋은 코드, 중요한 건 일관성

파이썬 코딩 컨벤션 – PEP8

- 들여쓰기 공백 4칸을 권장
- 한 줄은 최대 79자까지
- 불필요한 공백은 피함
- = 연산자는 1칸 이상 안 띄움
- 소문자I, 대문자 O, 대문자I 금지
- 함수명은 소문자로 구성
- "flake8" 모듈로 체크 (atom전용 flake8모듈도 존재)

함수 작성 가이드 라인

- 함수는 가능하면 짧게 작성할 것
- 함수 이름에 함수의 역할, 의도가 명확히 들어낼 것
- 하나의 함수에는 유사한 역할을 하는 코드만 포함
- 인자로 받은 값 자체를 바꾸진 말 것

```
def count_word(string_variable):
    string_variable = list(string_variable)
    return len(string_variable)
```