02.파이썬 기본 문법-2

6 함수

6.1 함수정의

```
def 함수명(매개변수):
    수행할문장1
    수행할문장2
    ...
    return   결과값
```

- def 키워드를 사용한다.
- 함수 이름과 뒤를 따르는 괄호안에 인수들이 기술된다.
- 함수 선언부는 항상 : 로 끝나야 한다.
- 들여쓰기 규칙이 적용되며 끝날 때는 들여쓰기가 적용 안되는 라인에서 끝나게 된다.
- (자바와 같이 return 자료형이나 매개변수의 자료형은 표기하지 않는다.)

[예제 function_def.py]

보통의 경우

```
def plus(a, b):
    sum= a+b
    return sum

result = plus(3, 7)
print(result)
```

매개변수와 리턴은 없을 경우도 있다.

```
def my_function():
    print("Hello world")


my_function()
print(type(my_function()))
```

내용이 없는 경우도 있다.

```
def nonefunction()
    pass


nonefunction()
```

6.2 return 문

return 문은 없어도 된다.

```
def my_function():
    print("Hello world")


my_function()
print(type(my_function()))
```

1개 이상의 값을 return 할 수 있다.

```
def sum_and_mul(a, b):
    return a+b, a*b


sumNum, mulNum = sum_and_mul(3, 9)
print(sumNum)
print(mulNum)
```

6.3 매개변수(정의할때), 인자(사용할때)

매개변수에 기본값을 지정하여 정의할 수 있다.

```
def plusPrint(a=0, b=0):
    print("a=%d, b=%d 이고 합은 %d 입니다." % (a,b, a+b))


plusPrint()
plusPrint(30)
# plusPrint(, 70)  #오류
plusPrint(b=70)
plusPrint(b=100, a=200)
```

매개변수의 개수를 정의할 수 없을때

```
def sum_many(*args):
    sum = 0
    for num in args:
        sum = sum + num
    return sum

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

다른 매개변수와 함께 사용할때

```
def sum_mul(mode, *args):
    if mode == "sum":
        result = 0
        for i in args:
            result = result + i
    elif mode == "mul":
        result = 1
        for i in args:
            result = result * i
    return result

print(sum_mul("sum", 1,2,3,4,5))
print(sum_mul("mul", 2,3))
```

정의되지 않은 매개변수는 key=value 형태로 전달하면 딕셔너리 형태로 받아 처리한다.)

```
def addPerson(hp, **kwargs):
    print(hp)
    print(kwargs)     # 타입 딕셔너리


addPerson('010-222-3333', name='홍길동', age=28)
```

다른매개변수와 같이 사용할때는 가장 마지막에 와야한다.

```
def addPerson2(*hp, **kwargs):
    print(hp)       # 타입 튜플
    print(kwargs) # 타입 딕셔너리


addPerson2('010-222-3333', name='홍길동', age=28)
addPerson2('010-222-3333','02-333-3333', name='홍길동', age=28)
addPerson2('010-222-3333','02-333-3333', '02-456-6434', name='홍길동', age=28)
```

6.4 스코핑 룰(Scope)

- 이름공간(Namespace) : 프로그램에서 사용하는 이름들이 저장되는 공간
- 이름은 값을 치환할 때 해당 값의 객체와 함께 생겨나고 이름공간에 저장된다.
- 이름공간에 저장된 이름을 통해 객체를 참조하게 된다.
- 이름공간(Namespace)
  - Local Scope : 함수 내부
  - Global Scope : 모듈 내부
  - Built-in Scope : 내장 영역
- 만일 동일한 이름이 이 세 가지 Scope에 존재하면 LGB 순서로 찾는다.

[예제 scope.py]
LGB 규칙에 따라 Local에서 찾을 수 없기 때문에 Global(모듈) 영역에서 x를 찾는다.

```
X = 1
def func(a):
    return a + X

print(func(10))
```

[예제 scope.py]
LGB 규칙에 따라 Local 영역에서 x를 찾는다.

```
X = 1
def func2(a):
    X = 2
    return a + X


print(func2(10))
```

[예제 scope.py]

함수 내부에서 전역 객체(외부변수)를 사용해야 하는 경우 global 선언문을 사용한다.

```
g = 1
def func3(a):
    global g
    g = 3
    return a + g


print(func3(10))
print(g)
```