04-1 함수

함수: 어떤 일을 수행한 후, 결과물을 내어놓는 것

- 함수를 사용하는 이유: 1. 반복되는 부분이 있을 경우, 2. 프로그램 흐름을 잘 파악할 수 있기 때문
- 함수의 구조

```
def 함수이름(매개변수):
  수행할 문장1
  수행할 문장2
  ...
```

- 매개변수: 함수에 입력으로 전달되는 값을 받는 변수
- 인수: 함수를 호출할 때 매개변수로 전달하는 입력값
- return: 결괏값을 돌려주는 명령어

함수의 형태

1. 일반적인 함수(입력값o, 결괏값o)
2. 입력값x 함수
3. 결괏값x 함수
4. 입력값x, 결괏값x 함수

```
def say(): #입력값x
  print("Hi") #결괏값x
$print는 결괏값을 내보내는 게 아님
결괏값은 오직 return 명령어로만
```

매개변수 지정하여 호출

```
def add(a,b):
  return a+b

result = add(a=3, b=5)
```

```
print(result)
'8'
```

$ 매개변수를 지정하면 순서 상관없이 사용할 수 있다

입력값이 몇 개가 될지 모를 때

방법: *매개변수

```
def add_many(*args):
  result = 0
  for i in args:
  result = result + i
return result
```

- 키워드 파라미터(**)

```
def key(**kwargs):
  print(kwargs)
...
key(a=1)
{'a':1}

매개변수 앞에 ** 을 붙이면 매개변수 kwargs는 딕셔너리가 되고,
key = value의 형태의 결괏값이 그 딕셔너리에 저장됨.
```

함수의 결괏값은 언제나 하나!

```
def add_mul(a,b):
  return a+b, a*b
...
결과는 (a+b,a*b)의 형태로 하나의 값(튜플)으로 출력됨
```

- 값을 따로 출력하고 싶을 때는 result1, result2처럼 함수를 두 번 호출
- return문을 2번 사용하면 2개의 결괏값 돌려주지x → return문이 끝나면 즉시 함수를 빠져나가기 때문에

$ return의 또 다른 쓰임새 → 함수를 즉시 빠져나가고 싶을 때 return 사용

매개변수 초깃값 미리 설정하기

```
def say_myself(name,age,man=True):
man=True처럼 미리 매개변수에 값을 넣어줄 수 있다.
$ 주의) 매개변수에 초깃값을 넣어준 경우, 매개변수의 가장 뒷쪽에 위치해야함
```

함수 안에서 선언한 변수의 효력 범위

- 함수 안에서 선언한 매개변수는 함수 안에서만 사용됨
- 함수 밖에서 선언한 변수는 함수 안에서의 변수 이름과는 전혀 상관X

$ 함수 안에서 함수 밖의 변수 변경하는 방법

→ 0. return 사용

→ 1. global명령어 사용(추천x)

lambda 함수

def와 같은 역할을 함(함수 생성)

```
lambda 매개변수1, 매개변수2, ,..: 표현식
```

차이점) 함수를 한 줄로 간결하게 만들 때 사용하거나, 복잡하지 않은 함수를 생성하거나, def를 사용할 수 없는 곳에 주로 쓰임