Python Basic

함수(Function)

함수(Function)

- 함수란?
  - 반복해서 사용하기 위한 코드의 묶음.
  - 미리 정의한 후 사용할 때 호출.

```
def 함수명(인수):
    실행할 코드
    [return 값]

변수 = 함수명(인수)
```

pass 문장

- pass 문장
  - 실행 시 아무런 기능도 하지 않는 명령어.
  - 빈 함수의 본문에 사용하여 에러 발생을 방지할 수 있음.

```
def calc():
   # 에러!

def calc():
   pass
   # 에러가 발생하지 않음
```

인수(Argument)

- 인수의 기본값
  - 함수를 호출할 때 인수를 전달하지 않은 경우 설정되는 기본값.
  - 함수를 정의할 때 해당 인수에 직접 값을 대입하여 설정함.
  - 인수 목록의 가장 오른쪽에 위치해야 함.

```
def calc(num, div = 1):
    if div == 0:
        print("Cannot divide by 0.")
        return (0, 0)
    else:
        return (num // div, num % div)

print(calc(11))
print(calc(11, 2))
print(calc(11, 0))
```

인수(Argument)

- 키워드 인수
  - 인수를 전달할 때 인수명을 직접 호출하여 지정하는 방식.
  - 정의한 인수의 순서와 호출한 인수의 순서가 달라도 상관없음.
  - 키워드 인수를 호출하면 이후의 인수도 키워드 인수로 호출해야 함.

```
def calc(begin, end, step):
    sum = 0
    for num in range(begin, end + 1, step):
        sum += num
    return sum

print(calc(3, 5, 1))
print(calc(3, 5, step = 1))
print(calc(begin = 3, end = 5, step = 1))
print(calc(step = 1, end = 5, begin = 3))
print(calc(3, step = 1, end = 5))
# print(calc(begin = 3, 5, 1))
# print(calc(end = 5, 3, 1))
```

인수(Argument)

- 가변 인수
  - 전달받을 인수의 개수를 미리 정하지 않는 것.
  - 함수 선언 시 인수명 앞에 * 기호로 표시.
  - 함수 선언 시 일반 인수 이후에 위치함.
  - 두 개 이상의 가변 인수를 선언할 수 없음.

```
def calc(*nums):
    result = 0
    for num in nums:
        result += num
    return result

print(calc(1, 2, 3))
print(calc(1, 3, 5, 7, 9))
print(calc(10, 20, 30, 40, 50, 60, 70, 80, 90))
```

인수(Argument)

- 키워드 가변 인수
  - 키워드 인수를 가변 개수 전달할 때 사용.
  - 함수 선언 시 인수명 앞에 ** 기호로 표시.
  - 함수 내에서 인수명과 값을 사전처럼 사용.

```
def calc(*nums, **options):
    result = 0
    for num in nums:
        if "sum" in options and options["sum"] == True:
            result += num
        elif "sub" in options and options["sub"] == True:
            result -= num
    return result

print(calc(1, 2, 3))
print(calc(1, 2, 3, sum = True))
print(calc(1, 2, 3, sum = False, sub = True))
```

지역 변수와 전역 변수

- 지역 변수(Local Variable)
  - 함수 내부에서 사용되는 변수.
  - 함수 외부에서는 접근할 수 없음.
  - 함수가 리턴되면 사라짐.

- 전역 변수(Global Variable)
  - 함수 외부에서 사용되는 변수.
  - 함수 내부에서도 접근할 수 있음.
  - 직접 삭제하기 전까지 사라지지 않음.

지역 변수와 전역 변수

- 파이썬의 지역 변수와 전역 변수
  - 함수 안에서 처음 대입하는 변수는 항상 지역 변수로 간주함.
  - 함수 안에서 초기화하는 변수는 전역 변수와 이름이 같더라도 항상 지역 변수로 간주함.
  - 전역 변수를 단순히 읽기만 하는 것은 상관없음.
  - global 키워드를 통해 전역 변수 사용 가능.

```
a, b, c = 10, 20, 30

def func():
    b = 200
    global c
    c = 300
    print("a:", a, "b:", b, "c:", c)

func()
print("a:", a, "b:", b, "c:", c)
```

지역 변수와 전역 변수

- global vs nonlocal
  - global은 무조건 최상위 scope 전역 변수를 가리킨다.
  - nonlocal은 바로 한 단계 위 scope 변수를 가리킨다(전역 제외).

```
num = 0

def first():
    num = 10

    def second():
        global num # 전역 변수 num
        # nonlocal num # first 함수의 지역 변수 num
        num = 20

    second()
    print("지역 변수 num:", num)

first()
print("전역 변수 num:", num)
```

docstring

- docstring이란?
  - 함수에 대한 설명.
  - 함수 선언문과 본체 사이에 문자열로 작성한다.
  - help() 함수와 __doc__ 특수 변수를 이용해 호출할 수 있다.

```
def echo(anything):
    'echo returns its input argument'
    return anything

print(echo('I am'))
help(echo)
print(echo.__doc__)
```

docstring

```
def print_if_true(thing, check) :
    '''
    Prints the first argument if a second argument is true,
    The operation is :
        1. Check whether the *second* argument is true.
        2. If it is, print the *first* argument.
    '''
    if check :
        print(thing)

help(print_if_true('abc', True))
print(print_if_true.__doc__)
```