비교과 겨울 방학 특강

파이썬 5일 - 수업자료

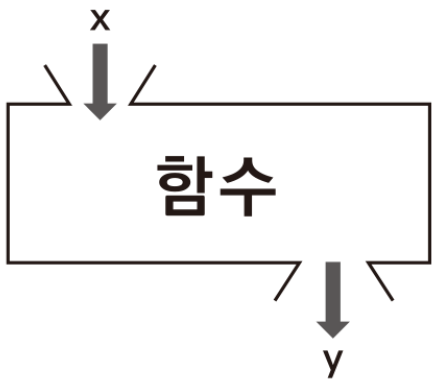
양 자영

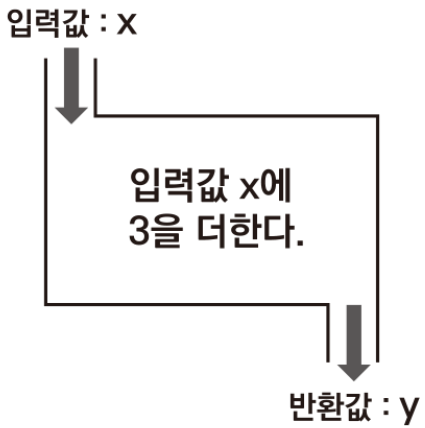
함수

함수란?

- **함수**는 일을 수행하는 코드의 덩어리이며, 더 큰 프로그램을 구축하는 데 사용할 수 있는 작은 조각이다.
- 반복적으로 사용되는 의미 있는 부분을 주로 함수로 구현
- 함수는 작업에 필요한 데이터를 전달받을 수 있으며, 작업이 완료된 후에는 작업의 결과를 호출자에게 반환

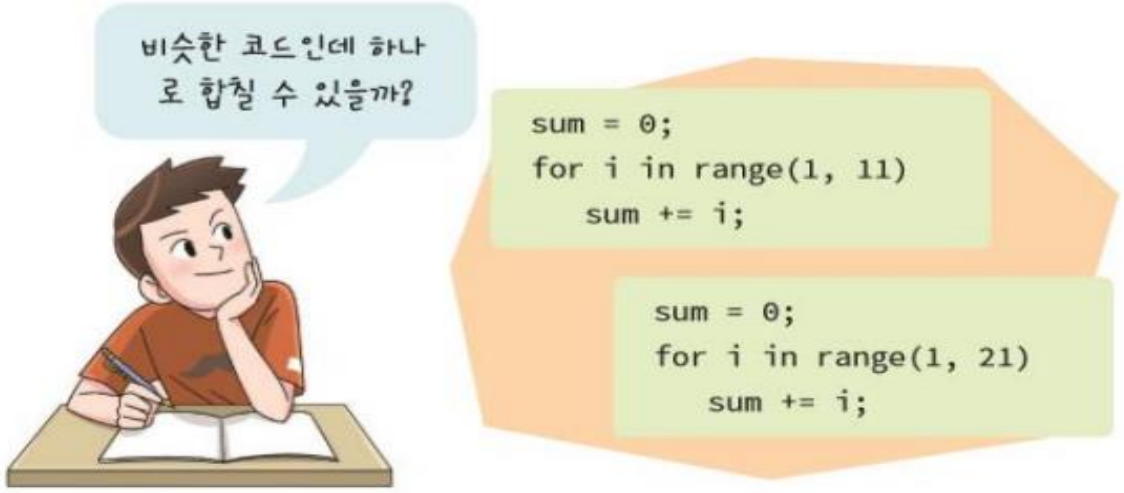
print( )
len( )
input( )
max( )
min( )





함수는 왜 필요한가?

- 프로그램 안에서 중복된 코드를 제거한다.
- 복잡한 프로그래밍 작업을 더 간단한 작업들로 분해할 수 있다.
- 함수는 한번 만들어지면 다른 프로그램에서도 재사용할 수 있다.
- 함수를 사용하면 가독성이 증대되고, 유지 관리도 쉬워진다.



함수의 종류

- 내장 함수
- 외장 함수 (외장 모듈의 함수)
  - 모듈 : 자주 쓰는 기능(함수)들을 하나로 묶은 것
- 사용자 정의 함수
  - 프로그래머가 자신의 특수한 문제를 해결하기 위해 필요에 따라 직접 만드는 함수

내장함수

```
print( )
len( )
input( )
max( )
min( )
```

외장함수

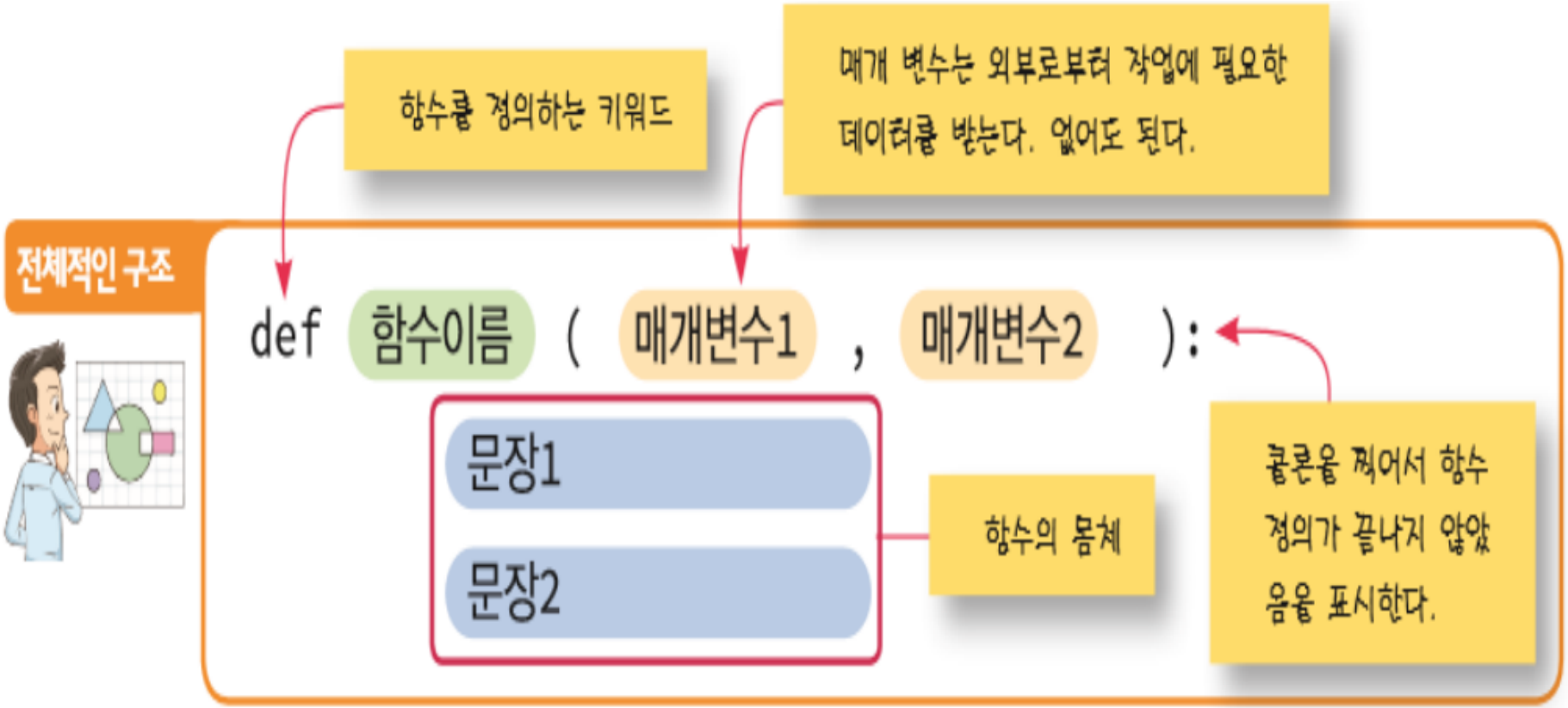
```
import math

print(math.log(4))
print(math.sqrt(9))
```

사용자 정의함수

```
def mysquare(x, y, len) :
    t.up()
    t.goto(x, y)
    t.down()
    for i in range(4) :
        t.forward(len)
        t.left(90)
```

```
x=0
y=0
length = 50
for i in range(4) :
    for j in range(4) :
        mysquare(x, y, length)
        x = x + length
    y = y + length
    x = 0
```

함수 작성의 예


함수를 정의하는 키워드
매개 변수는 외부로부터 작업에 필요한 데이터를 받는다. 없어도 된다.
전체적인 구조
def 함수이름 (매개변수1 , 매개변수2):
문장1
문장2
함수의 몸체
콜론을 찍어서 함수 정의가 끝나지 않았음을 표시한다.

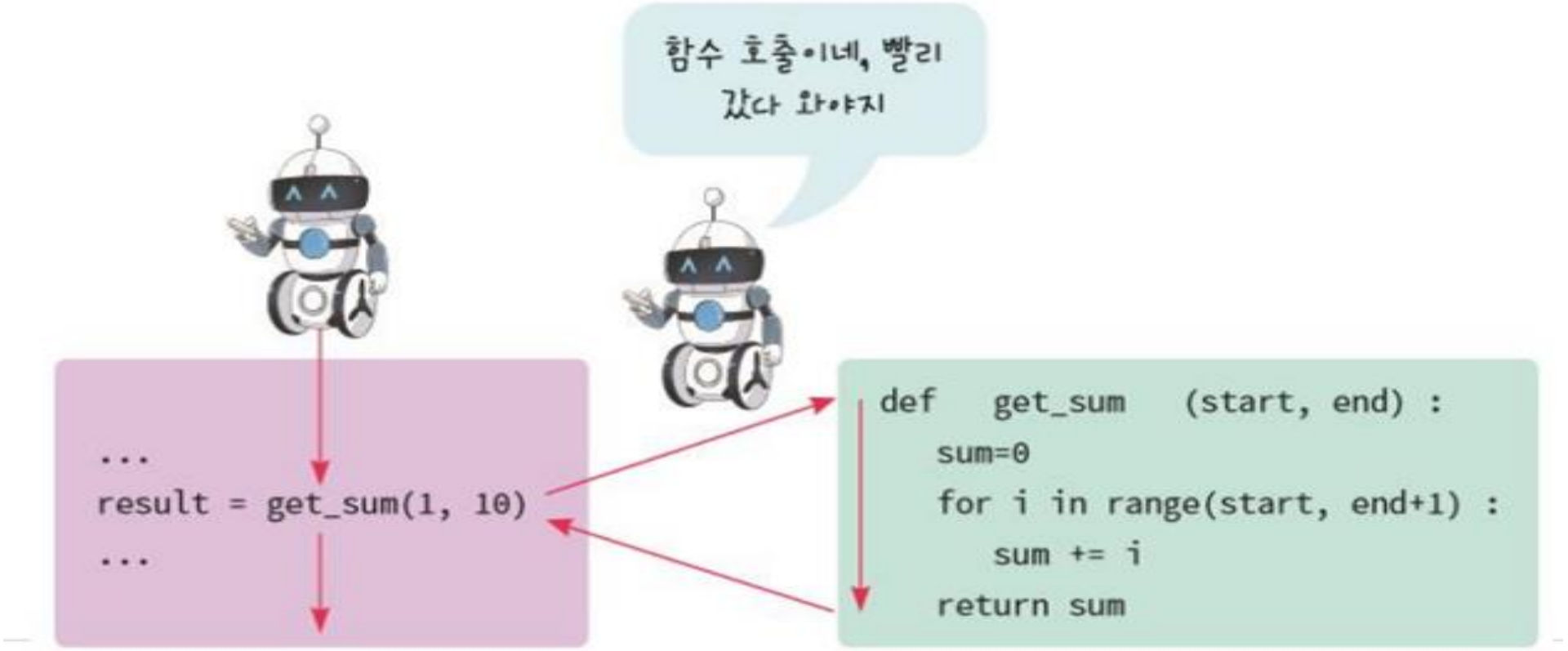
함수의 이름

- 함수의 목적을 설명하는 **동사** 또는 **동사+명사**를 사용

```
square(side)                    // 정수를 제곱하는 함수
compute_average(list)           // 평균을 구하는 함수
set_cursor_type(c)              // 커서의 타입을 설정하는 함수
```

함수 호출

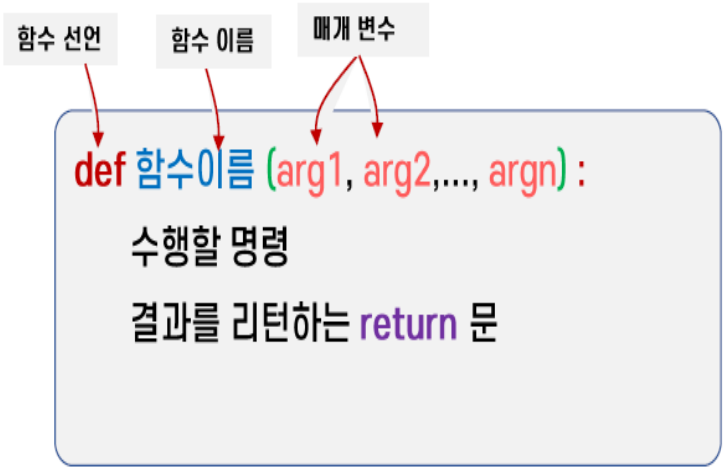
- 함수를 사용하려면 함수를 **호출(call)** 하여야 한다.



함수 작성하고 호출하기

- def 키워드를 이용하여 정의

```
def print_Hello() :          # 함수 정의
    print( " 반갑습니다")

print_ Hello()               # 함수 호출
print_ Hello()               # 함수 호출
print_ Hello()               # 함수 호출
```

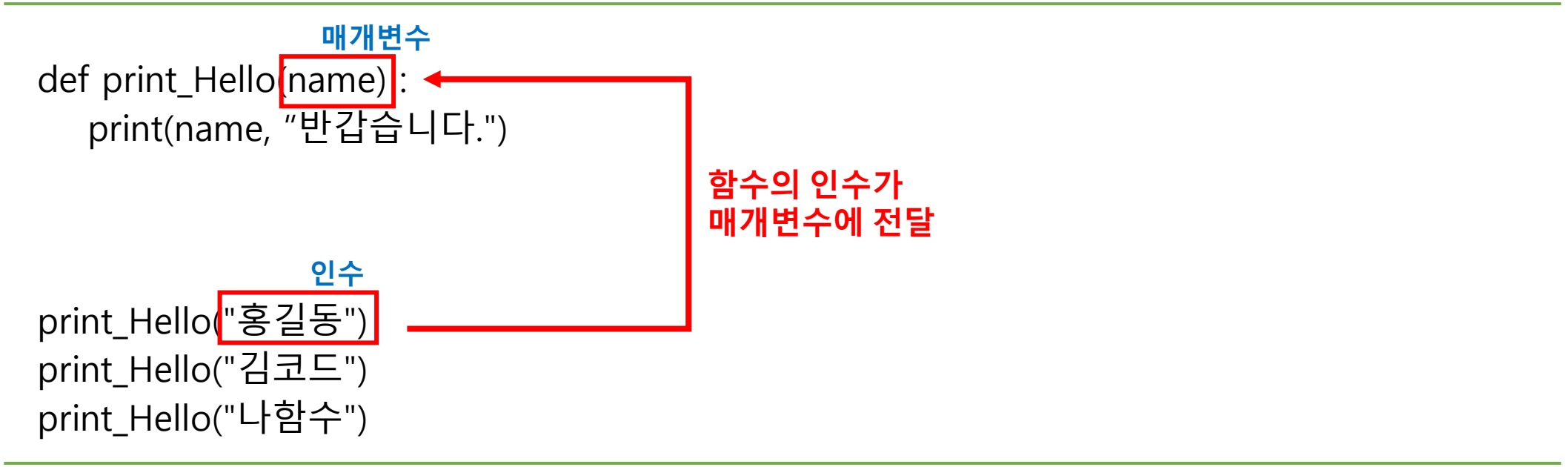


※ 함수를 호출하여 사용되려면 반드시 먼저 정의가 되어 있어야 함.

작업이 한곳에서만 일어난다면 반복문과 함수는 비슷하다.
하지만 함수는 프로그램의 여러 곳에서 동일한 작업을 시킬 수 있다. 또한 입력을 전달해서 호출할 때마다 다르게 실행할 수 있다.

함수의 인수와 매개변수

- 사용자는 함수를 사용할 때 작업에 필요한 정보(값)을 전달할 수 있다.
- 인수(argument) : 호출 프로그램에 의하여 함수에 실제로 전달되는 값
- 매개변수(parameter) : 인수를 담아 함수 내에서 쓰이는 변수
- 함수의 인수는 전달하지 않을 수도 있고 1개 이상 전달할 수도 있다.



함수의 인수와 매개변수

- 여러 개의 인수 전달 - start, end까지 더하는 함수

```
s = 0
for i in range(1,11):
    s += i
print("합계는 ", s)
```

```
s = 0
for i in range(start,end+1):
    s += i
print("합계는 ", s)
```

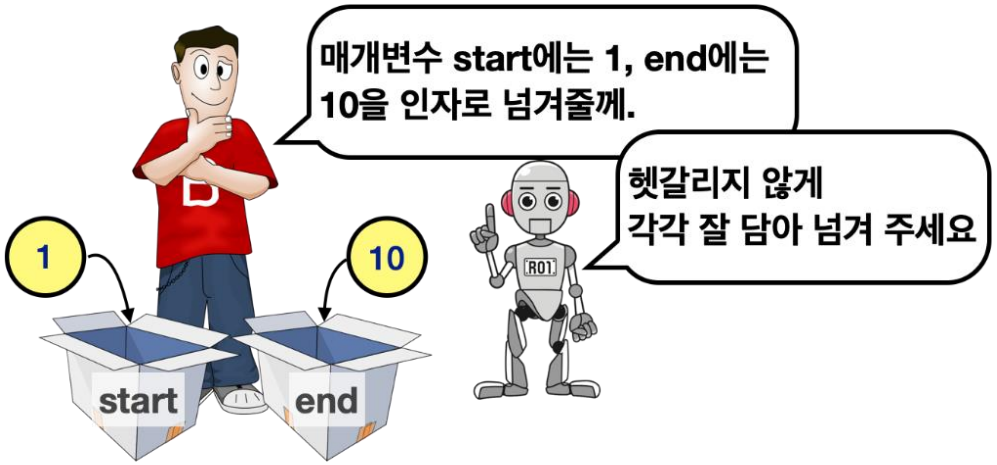
```
# 시작값(start)부터 끝값(end)까지의 합을 계산하여 출력하는 함수
def get_sum(start, end) :
    sum = 0
    for i in range(start, end+1) :
        sum += i
    print("sum = ", sum)

get_sum(1, 10)
get_sum(1, 20)
```

```
sum =  55
sum =  210
```

함수에 여러 개의 값을 넘겨주는 고급 기능

```
x = get_sum(1, 10) # 1과 10이 인자가 된다.
print('x =', x)

y = get_sum(1, 20) # 1과 20이 인자가 된다.
print('y =', y)
```

```
x = 55
y = 210
```

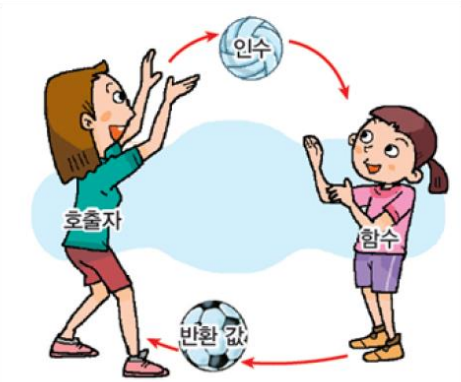


- 여기서 주의할 점은 매개변수의 개수와 넘어가는 인자의 개수가 정확히 일치하여야 한다는 점이다.
- 즉 매개 변수가 두 개이면 인자도 두 개를 전달하여야 한다. 매개변수의 개수와 인자의 개수가 일치하지 않으면 오류가 발생하게 된다.

함수의 값 반환

- 인수(argument) : 호출 프로그램에 의하여 함수에 실제로 전달되는 값
- 반환값(return value) : 함수를 수행한 후 함수 안에서 저장된 값을 호출한 쪽으로 반환하는 값

```
# 원의 반지름이 주어졌을 때 원의 면적을 구하는 함수
def calculate_area(radius):                # 함수 정의
    area = 3.14 * (radius ** 2)
    return area                            # 함수의 반환값

c_area = calculate_area(5.0)               # 함수 호출
print(c_area)
```



함수의 값 반환

- 여러 개의 값을 반환할 수도 있음

```
# 원의 반지름이 주어졌을 때 원의 면적과 둘레를 구하는 함수
def calc_circle(radius) :
    area = 3.14 * (radius ** 2)
    circum = 2 * 3.14 * radius
    return area, circum

c_area, c_circum = calc_circle(10)
print("원의 면적 =", c_area, "원의 둘레 =", c_circum)
```

함수의 구조

매개변수와 반환값이 모두 있는 함수	반환값만 있는 함수
def 함수명(매개변수): 수행할 명령문 … return 반환값	def 함수명( ): 수행할 명령문 … return 반환값
매개변수만 있는 함수	모두 없는 함수
def 함수명(매개변수): 수행할 명령문 …	def 함수명( ): 수행할 명령문 …

실습 - 함수 작성해 보기

- 정수를 입력받아서 제곱한 값을 반환하는 twice 함수를 만들고, 호출해보자.
  - 함수 이름 :
  - 입력 :
  - 출력 :

- 정수를 입력받아서 제곱한 값과 세제곱한 값을 반환하는 twice_triple 함수를 만들고, 호출해보자.
  - 함수 이름 :
  - 입력 :
  - 출력 :

- 점수(score)를 전달받아 70점이상이면 Pass, 60점이상이면 Retry, 60점 미만이면 Fail 을 출력하는 함수 **PassFail**을 작성하고, 호출해보자.

함수의 값 반환

- return으로 함수 중간에서 빠져나오기

```
def PassFail(score) :
    if score < 0  or score > 100 :
        print("잘못 입력했습니다")
        return                          # 이 부분에서 함수 종료. 만약 이 부분이 없다면 음수나 100
                                        # 보다 큰 값이 들어왔을 때의 출력 결과는 ?
    if score >= 80 :
        print("Pass")
    elif score >= 60 :
        print("재시험")                          잘못 입력했습니다
    else :
        print("Fail")

PassFail(-30)
```

실습 – BMI (함수 이용)

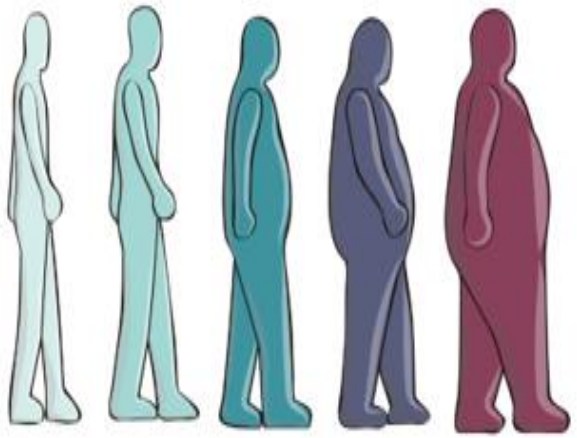
BMI 계산기

사용자로부터 ①키(m단위)와 몸무게(kg단위)를 입력받아, ②BMI를 계산한 후 ③결과를 출력하는 프로그램을 작성하시오.(단, 키와 몸무게는 메인 프로그램 부분에서 입력받고, BMI 계산과 결과 출력은 각각의 함수에서 작성한다.)

체질량지수(BMI) 공식

$$BMI = \frac{\text{몸무게(kg)}}{\text{키(m)} \times \text{키(m)}}$$

구분	BMI
저체중	<18.5
정상	18.5~22.9
과체중	23~24.9
경도비만	25~29.9
고도비만	30 이상



저체중, 정상, 과체중, 경도비만, 고도비만

```
def calc_BMI(height, weight) :
    #함수 완성하기


def result_print(bmi):
    if bmi <18.5 :
        print('저체중')
    #함수 완성하기




# 메인 프로그램 부분
h = float(input("키(m단위) : "))
w = float(input("몸무게(kg단위) : "))
bmi_result = calc_BMI(h, w)
print(bmi_result)
result_print(bmi_result)
```

실습 – 성적 판별(함수 이용)

- 다음 우리 반 5명 학생의 성적 리스트(slist)를 이용하여 다음 기능의 프로그램을 작성하시오.
  - 새로운 리스트(pflist)를 생성한 후 각 학생들의 성적에 대해 80점 이상이면 "PASS", 80점 미만이면 "FAIL"을 추가하는 함수를 작성하고 메인 프로그램에서 호출하여 결과를 출력한다.

```
def pass_fail(slist) :
    #1. pflist 초기화
    #2. slist의 원소에 대해서 반복
    #       만약에 원소가 80이상이면
    #            pflist에 "Pass" 추가
    #        아니면
    #             pflist에 "Fail" 추가
    # 3. pflist 반환




slist = [80, 70, 90, 60, 50]
pf = pass_fail(slist)
print(pf)
```

실습 – 주급 계산

주단위로 봉급을 받는 알바생이 있다. 현재 시급과 일한 시간을 입력하면 주급을 계산해 주는 함수 weeklyPay(rate, hour)을 만들고 함수를 호출해보자. 30시간이 넘는 근무시간에 대해서는 시급의 1.5배를 지급한다고 하자.

weeklyPay(10000, 38)

420000.0

모듈을 이용해서 함수를 두고두고 재사용하자

- **모듈**module이란, 파이썬 함수나 변수 또는 클래스들을 모아놓은 스크립트 파일을 말한다. 파이썬은 여러 기관과 개발자들에 의해서 만들어진 많은 모듈이 있으며 우리는 이 모듈을 이용하여 효과적으로 소프트웨어를 개발할 수 있다.

- 다음은 현재 스크립트나 대화창에서 모듈을 불러오는 명령이다.

```
import 모듈이름1 [, 모듈이름2, ... ]
```

모듈을 이용해서 함수를 두고두고 재사용하자

- 파이썬 설치와 함께 제공되는 모듈을 **파이썬 표준 라이브러리**python standard library라고도 하며 문자열과 텍스트 처리, 이진 데이터 처리, 날짜와 시간 처리, 배열 등의 자료형 처리, 수치 연산과 수학 함수, 파일과 디렉토리 접근, 유닉스 시스템의 데이터베이스 접근, 데이터 압축, 그래픽 등을 위한 100여 가지 이상의 모듈이 표준 라이브러리들로 제공된다.

자주 사용되는 모듈 - math

```
1  import math
2
3  print('절댓값:', math.fabs(-3.1415))
4  print('올림:', math.ceil(-3.1415))
5  print('내림:', math.floor(-3.1415))
6  print('버림:', math.trunc(-3.1415))
7  print('최대공약수:', math.gcd(9, 21))
8  print('팩토리얼:', math.factorial(10))
9  print('제곱근:', math.sqrt(4))
```

```
절댓값: 3.1415
올림: -3
내림: -4
버림: -3
최대공약수: 3
팩토리얼: 3628800
제곱근: 2.0
```

자주 사용되는 모듈 - random

```
1  import random
2
3  print('0 이상 1미만 임의 실수:', random.random())
4  print('0 이상 9이하 임의 정수:', random.randint(0, 9)) # 9 포함
5  print('0 이상 9미만 임의 정수:', random.randrange(9)) # 9 미포함
6  print('0 이상 9미만 난수 3개:', random.sample(range(0, 10), 3))
```

```
0 이상 1미만 임의 실수: 0.6446935559019615
0 이상 9이하 임의 정수: 9
0 이상 9미만 임의 정수: 5
0 이상 9미만 난수 3개: [0, 4, 5]
```