5. 함수(function)

5-1. 함수의 정의 및 호출

함수는 프로그램의 기능을 작은 task 단위로 나누어 재사용 가능한 독립적 형태로 나눈 것

함수는 결과값(return value)을 반환하는 함수와 결과값을 반환하지 않는 함수로 구분된다.

*** 결과값을 반환하지 않는 파이썬 함수의 예 : print( )**

*** 결과값을 반환하는 파이썬 함수의 예 : type( )**

함수의 작성 방법:

1. def keyword 로 시작한다.

2. 함수의 이름은 변수의 이름과 작성 규칙이 동일하다.

문자와 숫자, _ (underscore) 로 구성되며, 문자 (a~z, A~Z) 혹은 _ (underscore) 로만 시작 가능.

3. 함수 이름 다음에 ( ) 로 둘러싸여진 인수 (parameter, argument) 전달 가능

4. 들여쓰기 (indentation) 에 의해 함수의 body 임을 표시한다.

5. return keyword 를 이용하여 결과값을 반환한다. 반환할 값이 없으면 return 은 생략 가능하다.

```
def function_name (< parameters >):
    function_body
    return (value)
```

6. 함수의 반환값은 변수에 저장할 수 있다.

7. 여러개의 값을 한번에 반환할 수 있다.

8. Python 이 기본적으로 제공하는 함수를 내장함수 (built-in function) 이라고 한다.

print(), sum(), abs(), help(), str(), round(), type() 등 약 70 가지

*args, **kwargs

- *args 로 여러개의 parameter 를 한번에 받아들임
- **kwargs 로 여러개의 keyword argument 를 한번에 받아들임

In [1]:

```
def hello_world():
    print("hello world")

hello_world()
```

hello world

In [2]:

```
def return_result():
    a = 10
    b = 20
    result = a + b
    return result

x = return_result()
print(x)
```

30

In [3]:

```
def param_func(x, y):
    result = x / y
    return result

a = 10
b = 20
x = param_func(a, b)
print(x)
```

0.5

In [4]:

```
def circle(pi, radius):
    print("pi = ", pi)
    print("반지름 = ", radius)
    return pi * radius ** 2
```

In [5]:

```
circle(3.14, 10)
```

pi = 3.14
반지름 = 10

Out[5]:

314.0

In [6]:

```
r1, r2 = param_func(a, b) * 20, circle(3.14, a) / 10
print(r1, r2)
```

```
pi =  3.14
반지름 =  10
10.0 31.4
```

In [7]:

```
def plus_and_multi(m, n):
    return m + n, m * n

x, y = plus_and_multi(5, 10)
print(x, y)
```

```
15 50
```

In [8]:

```
def calculate_volume(length, width, depth):
    return length * width * depth

calculate_volume(10, 20, 30)
```

Out[8]:

```
6000
```

In [9]:

```
def sum_values(*args):
    total = 0
    for i in args:
        total += i
    return total
```

In [10]:

```
sum_values(1,2,3)
```

Out[10]:

```
6
```

In [11]:

```
sum_values(1,2,3,4,5)
```

Out[11]:

```
15
```

In [12]:

```
1  x = print("Hello World")
```

Hello World

In [13]:

```
1  print(x)
```

None

In [14]:

```
1  a = type(10)
```

In [15]:

```
1  print(a)
```

<class 'int'>

5-2. Scope of Variable (변수 영역)

variable 에는 global variable (전역변수) 과 local variable (지역변수) 의 두가지 종류가 있음

global variable - 함수의 바깥에 정의하고 프로그램 전체에서 접근 가능

local variable - 함수의 안쪽에 정의하고 함수 내에서만 접근 가능

gobal variable 과 local variable 이 같은 이름을 가질 경우는 local variable 이 우선함

함수의 영역(Scope of Function) 도 변수의 영역과 동일함.

In [16]:

```
1  a = 1
2  b = 2
3
4  def func():
5      c = 3
6      d = 4
7      print("global variable :", a, b)
8      print("local  variable :", c, d)
```

In [17]:

```
1 func()
```

```
global variable : 1 2
local  variable : 3 4
```

In [18]:

```
1 print(c)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-18-1dd5973cae19> in <module>
----> 1 print(c)

NameError: name 'c' is not defined
```

In [19]:

```
1 price = 100
2
3 def nego(price):
4     price = price - 10
5     print("negotiated price =", price)
6
7 nego(price)
8 print(price)
```

```
negotiated price = 90
100
```

In [20]:

```
1  def outer(x):
2
3      def inner(y):
4          return y ** 2 + 2
5
6      z = inner(x) + 2
7
8      return z
9
10 print(outer(4))
```

```
20
```

In [21]:

```
1 inner(4)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-21-1bb6dd009076> in <module>
----> 1 inner(4)

NameError: name 'inner' is not defined
```

In [22]:

```
1 def outer(x):
2     def inner(y):
3         return y ** 2 + 2
4     return inner(x) + 2
5
6 outer(4)
```

Out[22]:

```
20
```

In [23]:

```
1  def square(length):
2      print("정사각형의 한변 = ", length)
3      return length ** 2
4
5  def outer(x):
6      def inner(y):
7          return y ** 2
8      return inner(x) + square(x)
9
10 outer(4)
```

```
정사각형의 한변 =  4
```

Out[23]:

```
32
```

In [24]:

```
1 def rectangle(L1, L2):
2     return L1 * L2
3
4 print(rectangle(10, 20))
5 print(rectangle(L1=10, L2=20))
```

```
200
200
```

5-3. *args, **kwarg

In [25]:

```
def names(*args):
    print(args)

names('오', '영제', '남성')
```

('오', '영제', '남성')

In [26]:

```
def names(**kwargs):
    print(kwargs)

names(first='오', second='영제', third='남성')
```

{'first': '오', 'second': '영제', 'third': '남성'}

In [27]:

```
def names(*args, **kwargs):
    print(args)
    print(kwargs)

names('오', '영제', gender='남성')
```

('오', '영제')
{'gender': '남성'}

5-4. Python 의 내장함수 (Built-In Function)

Python 에 내장되어서 명령어처럼 사용할 수 있는 함수(function)들을 내장함수 (built-in function) 이라고 한다.

In [28]:

```
print(abs(-3))
```

3

In [29]:

```
print(bool(0))
print(bool(1))
```

False
True

In [30]:

```
year = input('태어난해를 입력하시오')
```

태어난해를 입력하시오

In [31]:

```
1 year
```

Out[31]:

''

In [32]:

```
1 eval('10 * 5')
```

Out[32]:

50

In [33]:

```
1 float("12")
```

Out[33]:

12.0

In [34]:

```
1 int('123')
```

Out[34]:

123

In [35]:

```
1 lst = list((1,2,3))
```

In [36]:

```
1 len(lst)
```

Out[36]:

3

In [37]:

```
1 max(lst)
```

Out[37]:

3

In [38]:

```
1 min(lst)
```

Out[38]:

1

In [39]:

```
1 test_file = open("./test_text.txt")
```

In [40]:

```
1 test_file.read()
```

Out[40]:

'This is a python open built-in function test file'

5-5. 연습문제

1) 선형방정식 (linear equation) y = mx + b 를 함수로 작성한다.

이때, 기본값(default value)은 m = 1, b = 0 로 한다.

2) 다음 함수가 수행된 후 print 되는 값은 ?

```
def f(x):
    return x + 1, x * x

x, y = f(3)
print(x, y)
```

3) 다음 code 가 수행된 이후 z 의 값은 ?

```
def f1(x, y):
    return (x + 1) / (y - 1)

z = f1(2, 2)
print(z)

def f1(x, y=2):
    return (x + 1) / (y - 1)

z = f1(1)
print(z)
```

4) 섭씨 온도를 화씨 온도로 변환하는 함수를 작성한다. 변환 공식은 다음과 같다.

$$T_f = \frac{9}{5}T_c + 32$$