



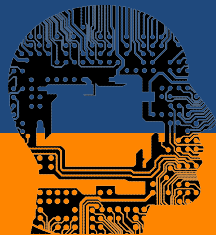
한국IT진흥부설

정보보호교육학원 아이섹

# 파이썬 기초

## 함수(Function)

By SoonGu Hong



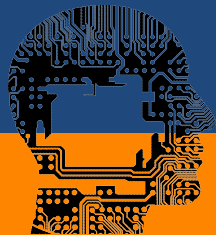


한국IT진흥부설

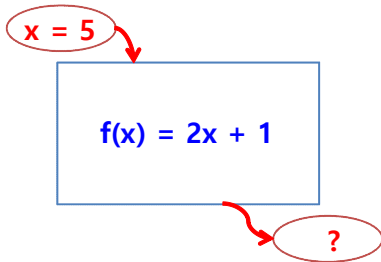
정보보호교육학원 아이섹

# 파이썬 기초

## 1. 함수 기본



## 1-1. 함수란?



- 함수란 **반복사용되는 코드들을** 묶어둔 코드 블록을 말합니다.
- 수학의 함수처럼 어떤 데이터를 함수에 전달하면 특정 값을 반환하는 형태로 만들어져 있습니다.

## 1-2-1. 함수의 선언

함수가 실행할 때 필요한 데이터  
를 받기 위한 변수

**def** 함수이름 ( [매개변수 선언, ...] ) :

실행할 코드를 작성하는 곳

**return** 결과반환 데이터

## 1-2-2. 함수 선언 예시

```
# 함수의 정의
def calc_sum(x):
    sum = 0
    for n in range(1, x+1):
        sum += n
    return sum
```

- 함수는 코드들을 **기능단위**로 사용하기 위한 목적으로 선언합니다. 위 코드는 1부터 x까지의 총합을 구하는 코드를 구성할 목적으로 선언한 예시입니다.
- 함수를 한번 정의해두면 지정해둔 함수 이름을 통해 언제든지 해당 코드 블록을 **재사용**할 수 있습니다.
- 파이썬에서는 함수를 호출하려면 반드시 호출문보다 상단부에 함수를 먼저 정의해야 합니다.

## 1-3. 함수 호출

# 함수의 호출

```
print("1~5까지의 누적합:", calc_sum(5))  
print("1~10까지의 누적합:", calc_sum(10))  
print("1~50까지의 누적합:", calc_sum(50))
```

```
1~5까지의 누적합: 15  
1~10까지의 누적합: 55  
1~100까지의 누적합: 5050
```

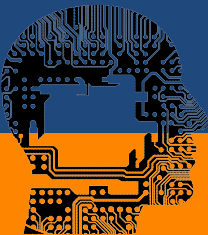
- 함수를 선언했다면 선언한 자체로는 사용이 불가능합니다.
- 함수는 반드시 **호출(call)**을 통해 기능을 동작시켜야 합니다.
- 호출 시에는 함수의 이름과 함께 함수에게 전달할 값을 소괄호 안에 넣어줍니다.



한국IT진흥부설

정보보호교육학원 아이섹

# 파이썬 기초



## 2. 매개변수(parameter)

## 2-1. 매개변수란?

```
def get_items(weapon, armor):  
    w = "{}' 무기를 획득했습니다.".format(weapon)  
    ar = "{}' 방아구를 획득했습니다.".format(armor)  
    detail = w + "\n" + ar + "\n-----"  
    return detail  
  
print(get_items("대검", "철갑옷"))  
print(get_items("누더기옷", "지팡이"))
```

- 매개변수란 함수 실행을 위해 필요한 데이터를 함수 내부로 전달할 목적으로 사용하는 변수입니다.

- 모든 함수가 매개변수가 필요한 것은 아닙니다. 이를테면 사람은 일기를 쓰기 위해 펜이라는 매개체가 필요하지만 일기를 읽기 위해서는 다른 도구가 없어도 되는 것처럼 말이지요



## 2-2. 매개변수가 없는 경우

```
# 인수를 전달받지 않는 함수의 예
def get_board_article():
    return '게시글을 30개 가져왔습니다.'

print(get_board_article())
```

- 함수 안에서 실행하는 값을 사용하지 않을 때 매개변수를 선언하지 않는 함수를 정의하는 것.
- 함수를 정의할 때도 소괄호를 비워놓고 호출할 때도 소괄호를 비워 호출합니다.

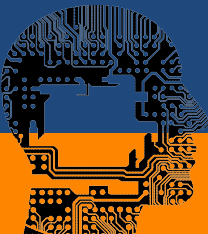


한국IT진흥부설

정보보호교육학원 아이섹

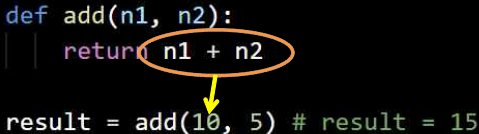
# 파이썬 기초

## 3. 리턴(return)



### 3-1. return

```
def add(n1, n2):  
    return n1 + n2  
  
result = add(10, 5) # result = 15
```



- 반환값이란 함수를 호출한 곳으로 함수의 최종실행 결과를 전달하는 값입니다.
- 매개변수는 호출부에서 함수에게 전달되는 값이고 반환값은 호출부에게 실행결과를 보고하는 값입니다.
- 매개변수는 여러 개 존재할 수 있지만 반환값은 언제나 하나만 존재해야 합니다.

## 3-2. return은 함수의 탈출문!

```
def sum_sub(n1, n2):  
    return n1 + n2  
    return n1 - n2
```

- 함수는 return을 만나는 순간 즉시 종료됩니다.
- 따라서 위와 같이 return이후에 코드를 적게되면 실행되지 않습니다.

### 3-3. return이 없는 함수

```
def multi(n1, n2):  
    result = n1 * n2  
    print("%d x %d = %d" % (n1, n2, result))
```

- 함수에 항상 return문이 필요한 건 아닙니다.
- 위와 같은 코드는 2개의 정수의 곱셈을 출력하고 싶을 뿐이지 곱한 값을 가져오고 프로그램에서 가져오고 싶지는 않기 때문에 리턴을 사용하지 않습니다.

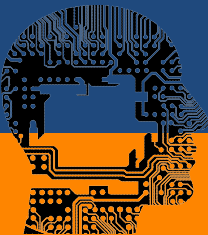


한국IT진흥부설

정보보호교육학원 아이섹

# 파이썬 기초

## 4. 함수 응용



## 4-1. docstring

```
def hello(message):  
    """  
    * 이 함수는 메시지를 전달하면  
    hello와 함께 메시지를 출력해 줍니다.  
    """  
    print("hello!! {}~!".format(message))
```

```
help(hello)
```

```
hello(message)
```

```
* 이 함수는 메시지를 전달하면  
hello와 함께 메시지를 출력해 줍니다.
```

- docstring이란 함수의 설명서를 뜻합니다.
- 함수 정의부 안에 따옴표 3개를 통해 함수의 설명을 적어두면 docstring으로 자동인식합니다.
- help()함수로 정의된 함수의 설명을 볼 수 있습니다.

## 4-2. 여러개 값 반환하기

```
def operate_all(n1, n2):  
    return n1 + n2, n1 - n2, n1 * n2, n1 / n2  
  
result = operate_all(10, 5)  
print(type(result))  
  
print("덧셈 결과:", result[0])  
print("뺄셈 결과:", result[1])  
print("곱셈 결과:", result[2])  
print("나눗셈 결과:", result[3])
```

```
<class 'tuple'>  
덧셈 결과: 15  
뺄셈 결과: 5  
곱셈 결과: 50  
나눗셈 결과: 2.0
```

- return문 뒤에 반환값을 여러 개 콤마로 나열하여 적으면 자동으로 튜플(tuple)에 묶어서 반환합니다.
- 위 방법 이외에도 리스트나 딕셔너리 등으로도 묶어서 리턴한다면 여러 개의 값을 반환할 수 있습니다.



## 4-3. 기본값을 갖는 매개변수

```
def make_url(ip, port=80):  
    return "http://{}:{ {}".format(ip, port)  
  
print(make_url("www.kokono.com"))  
print(make_url("www.kokono.com", 8080))
```

```
http://www.kokono.com:80  
http://www.kokono.com:8080
```

- 기본값으로 자주 사용되는 값은 매개변수로 기본값을 사전에 설정해둘 수 있습니다.
- 위 예제에서는 2번째 매개변수에 port값을 주지 않으면 자동으로 80으로 처리합니다.

## 4-4. 키워드 인수

```
def calc_stepsum(begin, end, step):  
    sum = 0  
    for n in range(begin, end+1, step):  
        sum += n  
    return sum  
  
# 위치 인수값을 사용  
print(calc_stepsum(3, 7, 1))  
  
# 키워드 인수값을 사용  
print(calc_stepsum(begin=3, end=7, step=1))  
print(calc_stepsum(end=7, step=1, begin=3))
```

- 함수에 데이터를 전달할 때 매개변수의 이름을 사용하여 전달하면 순서에 상관없이 데이터를 전달할 수 있습니다.

## 4-5. 튜플 매개변수를 이용한 가변인수 설정

```
def calc_total(*nums):  
    sum = 0  
    for n in nums:  
        sum += n  
    return sum  
  
print("-" * 40)  
print(calc_total(5, 7, 9))  
print(calc_total(5, 7, 9, 11, 100))
```

- 매개변수 앞에 \*을 붙이면 함수 호출부에서 콤마로 나열되어 전달된 인수들을 모아 튜플로 받아서 처리합니다.

## 4-6. 딕셔너리 매개변수

```
def calcstep(**args):
    print(args)
    start = args["start"]
    end = args["end"]
    step = args["step"]

    sum = 0
    for num in range(start, end + 1, step):
        sum += num
    return sum

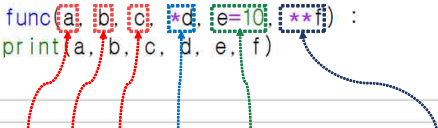
# call
print(calcstep(start=4, end=40, step=4))
```

```
{'start': 4, 'end': 40, 'step': 4}
220
```

- 매개변수 앞에 \*\*을 붙이고 키워드 인수형태로 데이터를 전달하면 함수 내부에서 딕셔너리로 묶어서 처리합니다.

## 4-7. 매개변수의 순서

```
1 def func(a, b, c, *d, e=10, **f) :  
2     print(a, b, c, d, e, f)  
3
```



```
1 func(10, 20, 30, 4, 5, e=20, name="JinKyoung", age=30)
```

10 20 30 (4, 5) 20 {'name': 'JinKyoung', 'age': 30}

- 여러 종류의 매개변수를 설정할 때에는 반드시 위의 순서를 지켜야 합니다.

- 위치기반 매개변수, 튜플 매개변수, 키워드기반 매개변수, 딕셔너리 매개변수 순서

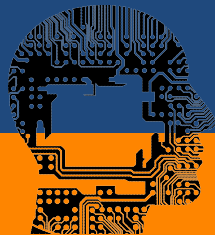


한국IT진흥부설



정보보호교육학원 아이섹

# 파이썬 기초

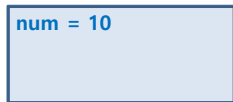
## 5. 지역변수와 전역변수



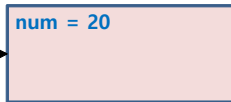
## 5-1. 지역변수와 전역변수

지역 변수	전역 변수
<pre>def func_a():     num = 10     print(num) # 10 }  def func_b():     print(num) # 에러 }</pre>  <p>생명주기</p>	<pre>num = 20 def func_a():     print(num) # 20 }  def func_b():     print(num) # 20 }</pre>  <p>생명주기</p>

지역 심볼 테이블



전역 심볼 테이블



## 5-2. 지역변수와 전역변수

### 1. 함수 안에 정의된 변수들을 지역변수(Local Variable)

### 2. 로컬 심볼 테이블(Local Symbol Table)

- 함수가 실행될 때에 지역변수들은 함수 실행을 위한 특별한 영역에 저장
- 함수가 실행될 때 함수내의 모든 지역변수들은 해당 함수의 로컬 심볼 테이블에 값을 저장

### 3. 전역 테이블(Global Symbol Table)

- 함수 밖에 정의된 전역변수들을 저장하는 공간

### 4. 변수의 값 조회 순서

- 로컬 심볼 테이블 > 전역 심볼 테이블 > 내장 된 이름 테이블



**감사합니다**  
**THANK YOU**