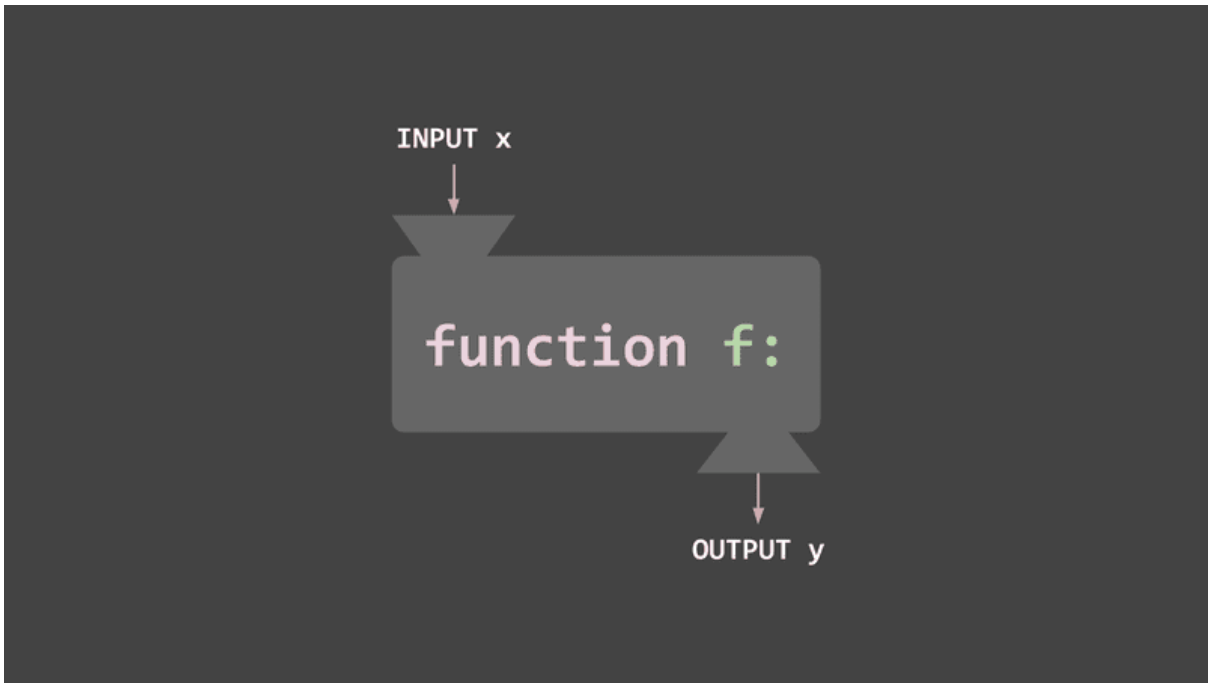
14. 파이썬 - 함수

Made By. 김규일

함수란 무엇일까

- 함수는 수학의 함수와 비슷하다.
- $y = ax + b$ 를 생각해보자. 좌측의 식은 흔한 직선그래프 함수이다.
- 위의 식은 특정 입력(x) 값에 따라 출력(y) 값이 나온다.
- 프로그래밍에서의 함수 또한 비슷하다.
- 프로그래밍에서의 함수는 특정 입력에 대하여 특정한 수행을 한 후에 결과물을 내어놓는 것이다.



- 함수를 위의 사진의 형태를 이해하면 편하다.
- 실제 함수는 4가지의 형태가 있다.

- 1) 위의 사진처럼 입력과 출력이 있는 형태
- 2) 입력은 있고 출력이 없는 형태
- 3) 입력은 없고 출력만 있는 형태
- 4) 입력도 출력도 없는 형태

함수를 왜 사용할까

- 함수를 사용하는 이유는 간단하다. 반복되는 수행들을 함수로 처리하여 쉽고 간편하게 사용하기 위해서이다.
- 또한 함수를 이용해서 코드를 작성함으로써 코드의 가독성을 높이는 것이다.
  - 이를 보통 유지보수성이 좋다고 표현한다.
- 예를 들어 이중 for문으로 이루어진 구구단 코드가 있다고 생각해보자. 구구단을 코드에서 한번만 사용한다면 상관이 없지만 만약 10번 사용한다면 이중 for문의 코드를 10번 반복해서 붙여넣어야 한다. 이를 만약 함수로 만든다면 단지 구구단 함수만 호출하면 쉽게 해결 가능하다.

파이썬 함수 형태

- 함수는 기본적으로 입력과 출력, 수행해야 할 것들로 이루어져 있다.
- 함수는 다음과 같은 형태로 만들 수 있다.

```
def 함수명(매개변수):
    <수행할 문장1>
    <수행할 문장2>
    ...
```

- 여기서 def는 함수를 만들겠다는 파이썬 예약어이다. 예약어는 if, for, while 처럼 이미 등록되어 문법에 사용되는 하나의 약속된 단어라고 생각하면 된다.
- 함수명은 본인 스스로 아무렇게 지으면 된다. 물론 다양한 표기법들이 있고 이러한 표기법들을 따르는 것을 추천한다.
  - 다양한 표기법 중에서 카멜 표기법을 추천한다. 카멜 표기법은 변수 안의 단어를 따로 구분없이 붙여서 작성하고 맨 앞의 글자를 대문자로 작성하는 방식이다. 대신, 첫 단어의 맨 앞 글자는 소문자로 작성한다.

- 또한 변수 안의 단어들의 조합은 동사가 아닌 명사로 작성하는 것이 보통이다.
- 예시)

```
int totalNumber;
int phoneNumber;
int count;
char backgroundColor
...
```

- 함수명, 변수 이름, 클래스 이름 등 대부분 카멜 표기법으로 작성하는 것을 추천한다.

- 다음의 예시를 보자

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

- def는 예약어, add는 함수 이름이다.
- 매개변수 안의 a, b가 입력 값이 된다. return(예약어) 옆의 a + b가 출력이 된다.
- 즉 위의 함수는 a, b라는 변수를 입력으로 받아 a+b를 출력으로 내놓는 함수가 된다.
- 위의 형태로 알 수 있듯이 함수의 입력은 매개변수를 통해 조절하고 출력은 return을 통해 조절한다.
- 함수의 4가지 형태로 생각했을 때
  - 1) 입력과 출력이 둘 다 존재할 때 → 매개변수, return 둘 다 존재

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

  - 2) 입력이 없고 출력만 존재할 때 → 매개변수는 공백, return 존재

```
def add():
    return 10
```

  - 3) 입력만 있고 출력은 없을 때 → 매개변수는 존재, return은 X

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

- 4) 입력도 없고 출력도 없을 때 → 매개변수는 공백, return은 X

```
def add():
    print("Empty")
```

- 여기서 중요한 점은 입력이 없어도 매개변수는 공백으로 () 존재하지만 출력이 없다면 return 자체를 안 써도 된다는 점이다.
- 이는 함수를 사용할 때 괄호의 유무를 보고 변수와 함수를 판단하기 때문이다.
- 즉 파이썬에서 괄호가 붙은 대부분은 함수라고 생각해도 무방하다.
- 함수를 사용할 때는 예약어 def는 제외한 add()의 형태로 사용한다.

```
>>> a = 3
>>> b = 4
>>> c = add(a, b)
>>> print(c)
7
```

- 중요한 점은 입력(함수 안에 매개변수)이 있다면 꼭 괄호를 채워줘야 한다.
- 또한 return이 있다면 return의 결과를 받아주는 것 또한 필요하다.
  - 결과를 받아준다는 것은 c = add(a, b) 처럼 결과를 다른 변수에 넣거나 print(add(a, b)) 처럼 결과를 또 다른 함수의 매개변수로 전달하는 것이다.

- 파이썬의 함수는 매개변수의 개수를 미리 지정하지 않아도 된다.
- 즉 매개변수의 갯수를 미리 모를 때 아래와 같은 방법을 사용할 수 있다.

```
def 함수이름(*매개변수):
    <수행할 문장>
    ...
```

```
>>> def add_many(*args):
...     result = 0
```

```
...     for i in args:
...         result = result + i
...     return result
...
>>>

>>> result = add_many(1,2,3)
>>> print(result)
6
>>> result = add_many(1,2,3,4,5,6,7,8,9,10)
>>> print(result)
55
```

- * 기호를 통해 위와 같이 코드를 작성하면 여러 개의 매개변수를 다 처리할 수 있다.
- 아래처럼 기존의 매개변수와 혼합하여 사용 가능하다.

```
>>> def add_mul(choice, *args):
...     if choice == "add":
...         result = 0
...         for i in args:
...             result = result + i
...     elif choice == "mul":
...         result = 1
...         for i in args:
...             result = result * i
...     return result
...
>>>

>>> result = add_mul('add', 1,2,3,4,5)
>>> print(result)
15
>>> result = add_mul('mul', 1,2,3,4,5)
>>> print(result)
120
```

- 비슷한 예로 *를 2개 붙여서 사용하는 것도 존재한다.
- 이는 kwargs 라고 부르며 모든 매개변수로 들어오는 값들을 딕셔너리로 만들어준다.
- 물론 매개변수에 전달할 때 key=value의 형태로 전달해야 한다.

```
>>> def print_kwargs(**kwargs):
...     print(kwargs)
...

>>> print_kwargs(a=1)
{'a': 1}
```

```
>>> print_kwargs(name='foo', age=3)
{'age': 3, 'name': 'foo'}
```

- 매개변수는 여러 개가 가능하지만 return은 하나만 가능하다. 물론 return a+b, a*b의 형태로 만들 수 있지만 이때는 결과가 튜플로 리턴된다.
- 튜플을 각각 따로 받고 싶다면

```
>>> def add_and_mul(a,b):
...     return a+b, a*b

>>> result1, result2 = add_and_mul(3, 4)
```

- 위와 같이 사용하면 된다.

- 파이썬의 함수는 매개변수를 미리 정할 수 있다.

```
def say_myself(name, age, man=True):
    print("나의 이름은 %s 입니다." % name)
    print("나이는 %d살입니다." % age)
    if man:
        print("남자입니다.")
    else:
        print("여자입니다.")
```

- 당연히 함수를 사용하는 과정에서 매개변수를 다르게 전달해서 바꿔도 된다.
- 함수 생성 시 적는 값은 단지 초기값을 나타낼 뿐이다.

- 함수 안에서 선언된 변수는 함수 밖에서 사용할 수 없다.
  - 지역 변수라는 표현을 사용하며 변수는 선언된 구역안에서만 사용될 수 있다.
  - 다른 언어는 선언된 위치의 중괄호 {} 안에서만 사용 가능하고 파이썬은 선언된 위치의 들여쓰기 된 부분에서만 사용 가능하다.
- 함수 안에서 선언되거나 변경된 변수를 밖에서 사용하는 것은 보통 2가지 방법을 사용한다,

- 첫 번째는 return을 통해 해당 변수를 결과로 보내주는 것이다.
- 두 번째는 global을 통해 사용하는 방식이다. 아래의 예시 코드를 참조하자

```
a = 1
def vartest():
    global a
    a = a+1

vartest()
print(a)
```

- 파이썬은 또한 람다를 지원한다. 람다는 함수를 한줄로 만들어서 사용하는 것이다.
- 프로그래밍 초보자에게 람다는 어려울 수 있으니 참고하고 넘어가자

```
# 함수명 = lambda 매개변수1, 매개변수2, ... : 매개변수를 이용한 표현식

>>> add = lambda a, b: a+b
>>> result = add(3, 4)
>>> print(result)
7

# 위 아래가 동일한 코드

>>> def add(a, b):
...     return a+b
...
>>> result = add(3, 4)
>>> print(result)
7
```