함수

함수의 기본 형태와 활용

함수 (Function)

- 입력 값을 받아서 어떤 일을 수행한 뒤 그 결과 값을 되돌려 주는 구문들의 모음

입력 값

함수

결과 값

함수를 사용하는 이유

- 동일한 일을 수행하는 구문이 반복적으로 존재할 때
  - 이 구문들을 매번 작성할 필요없이, 하나의 함수로 구성해 두고 언제 어디서나 필요할 때마다 그 함수를 실행시키면 된다.

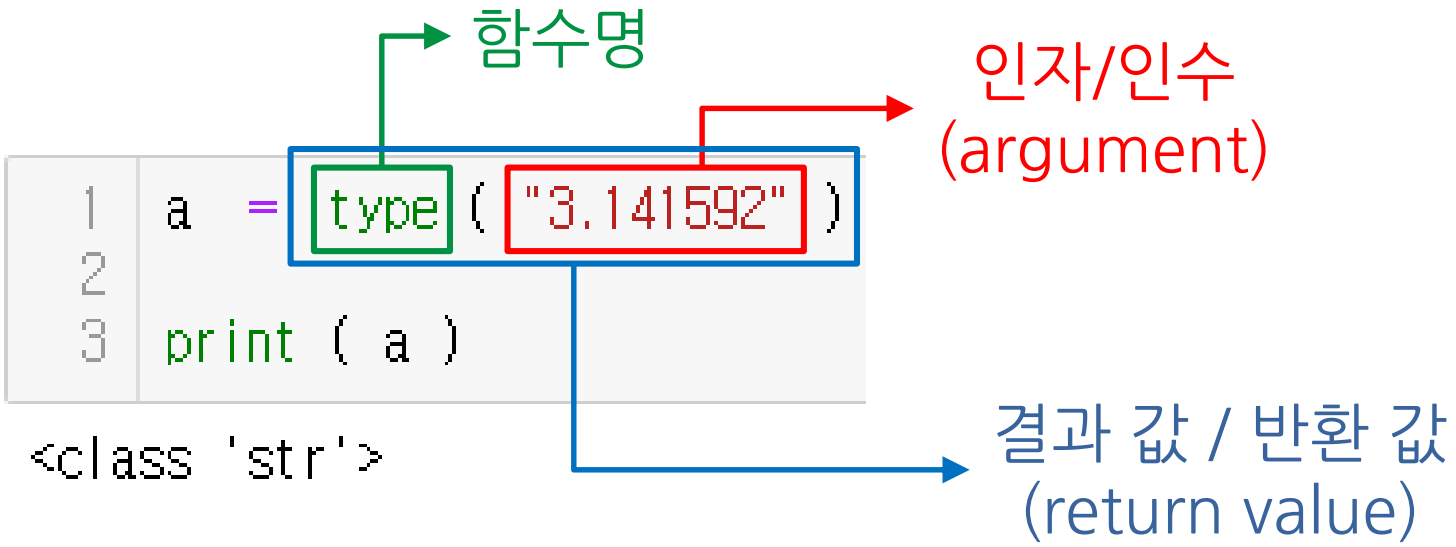
- 프로그램의 흐름을 보다 구조적으로 만들기 위해
  - 각각 다른 역할을 하는 구문들을 각각 다른 함수로 구성하여, 프로그램의 가독성을 더욱 향상시킨다.

함수 (Function)

- 함수 호출 (Function Call)

  결과 값을 할당할 변수 = 함수의 이름(필요한 입력 값(들))

  – 이미 존재하는 어떤 함수의 이름을 불러서 그 함수에게 담당한 일을 시키는 것을 함수 호출이라고 한다.



함수 (Function)

- 함수 호출 (Function Call)

  결과 값을 할당할 변수 = 함수의 이름(필요한 입력 값(들))

  – 함수들은 각자 역할에 따라 필요한 입력 값의 종류와 개수가 다르고, 도출되는 결과의 종류가 다르다.

```
1  a = 0
2  b = [1, 2, 3]
3  c = "Hello"
4
5  print(a, b, c)
```

```
0 [1, 2, 3] Hello
```

```
1  a = pow(3, 4)
2
3  print(a)
```

```
81
```

함수 (Function)

- 함수 호출 (Function Call)

  결과 값을 할당할 변수 = 함수의 이름(필요한 입력 값(들))

  – 어떤 함수들은 반드시 특정 대상을 지정해야 하고, 어떤 함수들은 범용적으로 사용할 수 있다.

  – 이 때, 범용으로 사용되는 함수를 내장 (built-in) 함수 라고 한다.

```
1  a = len("Hello")
2
3  print(a)
```

```
5
```

```
1  a = [1, 2, 3].pop()
2
3  print(a)
```

```
3
```

함수 (Function)

- 기본적인 내장 함수들

		Built-in Functions		
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

※ https://docs.python.org/3/library/functions.html

함수 (Function)

- 기본적인 내장 함수의 예 (1)

함수명	호출할 때의 인자	결과 값	수행하는 일
print	여러 개의 자료(들)	없음	인자로 받은 자료들을 순서대로 화면에 출력한다.
input	1개의 문자열 s	문자열	인자로 받은 문자열 s를 화면에 출력한 뒤, 사용자가 키보드로 타이핑하여 입력한 문자열을 반환한다.
len	1개의 자료 t	정수	인자로 받은 자료 구조 t에 들어 있는 항목의 개수를 반환한다.
pow	2개의 수치 x, y	수치	인자로 받은 수치 x의 y제곱을 구하여 반환한다.

함수 (Function)

- 기본적인 내장 함수의 예 (2)

함수명	호출할 때의 인자	결과 값	수행하는 일
abs	1개의 수치 x	수치	인자로 받은 수치 x의 절대값을 구하여 반환한다.
round	수치 x, 정수 y	수치	인자로 받은 수치 x에 대하여 정수 y자리까지 반올림한 수치를 반환한다.

```
1 abs(1234)
```

```
1234
```

```
1 abs(-5.678)
```

```
5.678
```

```
1 round(3.9)
```

```
4
```

```
1 round(123.456, 1)
```

```
123.5
```

```
1 round(123.456, -2)
```

```
100.0
```

함수 (Function)

- 기본적인 내장 함수의 예 (3)

함수명	호출할 때의 인자	결과 값	수행하는 일
eval	1개의 문자열 s	처리된 자료	인자로 받은 문자열 s에 대하여 연산 또는 처리된 결과를 반환한다.
sorted	1개의 자료 t	리스트	인자로 받은 자료 구조 t의 항목들이 정렬된 리스트를 반환한다.

```
1  a = eval("123 + 456")
2  print(a)
```

```
579
```

```
1  b = eval(input("내용을 입력하세요: "))
2
3  t = sorted(b)
4  print(t)
```

```
내용을 입력하세요: 3,9,-777, 3.1425, 123, 1000
[-777, 3, 3.1425, 9, 123, 1000]
```

함수 (Function)

- 기본적인 내장 함수의 예 (4)

함수명	호출할 때의 인자	결과 값	수행하는 일
range	3개의 정수 x, y, z	정수 목록	정수 x부터 정수 y 전까지 간격 z 단위로 나열된 정수의 목록을 반환한다.
enumerate	자료 t, 정수 x	튜플 목록	자료 구조 t의 항목에 순서대로 정수 x부터 시작하는 번호를 부여한 튜플의 목록을 반환한다.

```
1  a = (1, 4, 9, 16)
2  b = list(enumerate(a))
3
4  print(b)
```

```
[(0, 1), (1, 4), (2, 9), (3, 16)]
```

```
1  seasons = ["Spring", "Summer", "Fall", "Winter"]
2
3  for i, s in enumerate(seasons, 1):
4      print(s, i)
```

```
Spring 1
Summer 2
Fall 3
Winter 4
```

함수 (Function)

- 기본적인 내장 함수의 예 (5)

함수명	호출할 때의 인자	결과 값	수행하는 일
zip	2개 이상의 자료 t1, t2, ...	튜플 목록	각 자료 구조 t1, t2, ...에서 동일한 위치의 항목들을 순서대로 묶은 튜플의 목록을 반환한다.

```
1  a = [1, 2, 3]
2  b = "abcd"
3  c = list(zip(a, b))
4
5  print(c)
```

```
[(1, 'a'), (2, 'b'), (3, 'c')]
```

```
1  seasons = ["Spring", "Summer", "Fall", "Winter"]
2
3  for s, i in zip(seasons, range(1, 5)):
4      print(s, i)
```

```
Spring 1
Summer 2
Fall 3
Winter 4
```

함수 (Function)

- 함수 정의 (Function Definition)

  def 함수명(매개변수(들)):
  **   이 함수가 수행할 구문1**
  **   이 함수가 수행할 구문2**
  **   ...**

  – 어떠한 일을 수행할 함수를 사용자가 직접 작성하는 것을 함수 정의라고 한다.

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

```
1  def make_string(a, b, c):
2      d = a + b
3      e = a + c
4      print(d, e)
```

함수 (Function)

- 함수 정의 (Function Definition)

  def 함수명(매개변수(들)):
  이 함수가 수행할 구문1
  이 함수가 수행할 구문2
  ...

  – 함수 정의는 말 그대로 함수를 “만들어 두는 것”일 뿐이며, 자동으로 실행되는 것이 아니라 그 함수를 호출할 때에 실행된다.

```
1  add(1, 2)
```

```
3
```

```
1  make_string("hello", "Python", 2)
```

```
helloPython hellohello
```

함수 (Function)

- 함수 정의 (Function Definition)

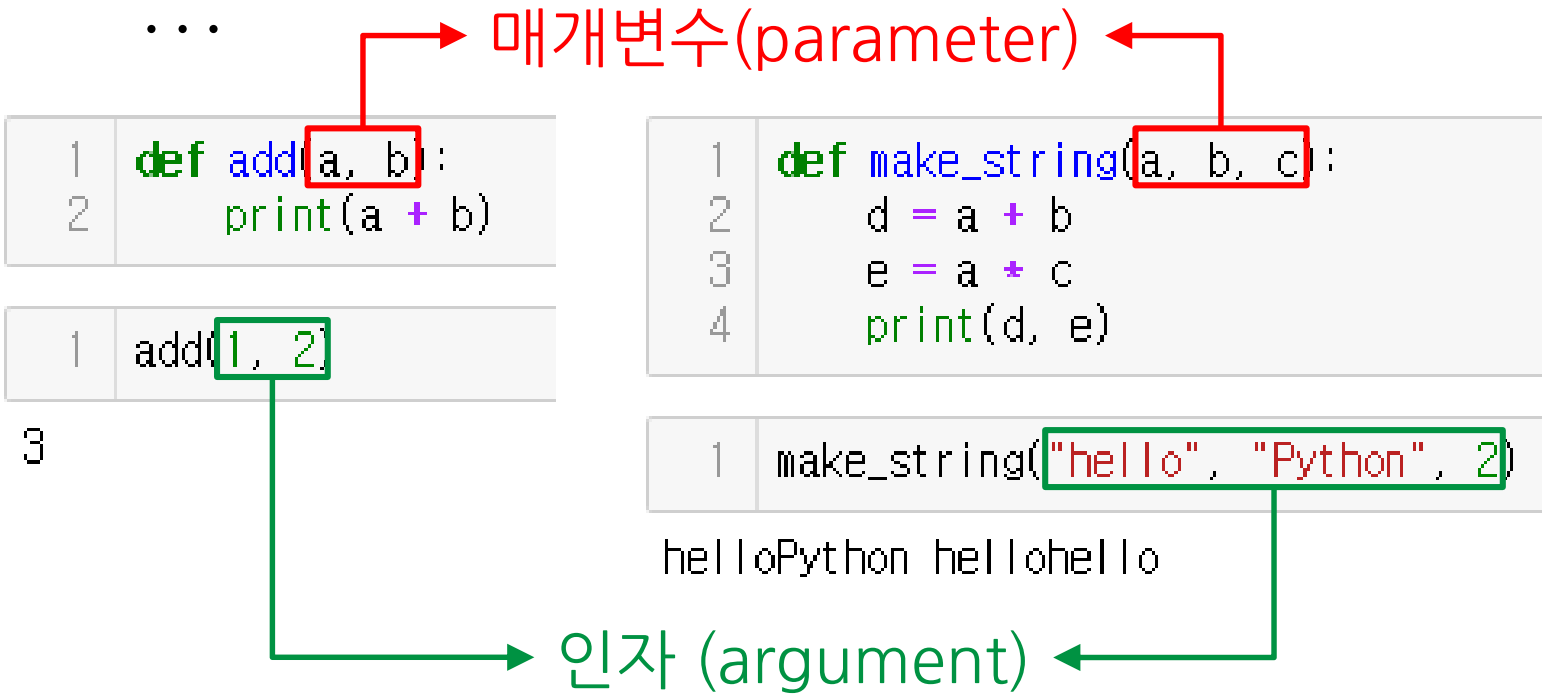
```
def 함수명(매개변수(들)):
    이 함수가 수행할 구문1
    이 함수가 수행할 구문2
    ...
```

  - 매개변수 (parameter)
    - 함수 정의에서 입력으로 전달된 값을 할당 받는 변수
  - 인자 (argument)
    - 함수 호출에서 입력으로 전달되는 값

함수 (Function)

- 함수 정의 (Function Definition)

  def 함수명(매개변수(들)):
  이 함수가 수행할 구문1
  이 함수가 수행할 구문2
  ...



함수 (Function)

- 함수의 실행 결과 값 반환

```
def 함수명(매개변수(들)):
    이 함수가 수행할 구문1
    이 함수가 수행할 구문2
    ...
    return 결과 값
```

  - 함수의 실행 결과를 되돌려 주려면, return 구문을 사용한다.

```
1  def add(a, b):
2      print("입력받은 2개의 숫자는", a, b)
3      return a + b
```

```
1  c = add(1, 2)
```

```
입력받은 2개의 숫자는 1 2
```

```
1  print(c)
```

```
3
```

함수 (Function)

- 함수의 실행 결과 값 반환
  - 함수의 실행 결과를 되돌려 주려면, return 구문을 사용한다.

```
1  def check_maximum(a, b):
2      if a > b:
3          max_value = a
4      elif a < b:
5          max_value = b
6      else:
7          max_value = "값이 같습니다."
8
9      return max_value
```

```
1  print(check_maximum(1, 2))
```

```
2
```

```
1  print(check_maximum(123, 0))
```

```
123
```

```
1  print(check_maximum(-9, -9))
```

```
값이 같습니다.
```

함수 (Function)

- 함수의 실행 결과 값 반환
  - 함수의 실행 결과를 되돌려 주려면, return 구문을 사용한다.

```
1  def check_maximum(a, b):
2      if a > b:
3          max_value = a
4      elif a < b:
5          max_value = b
6      else:
7          max_value = "값이 같습니다."
8
9      return max_value
```

```
1  def check_maximum(a, b):
2      if a > b:
3          return a
4      elif a < b:
5          return b
6      else:
7          return "값이 같습니다."
```

함수 (Function)

- 함수의 형태
  - ① 입력(인자) 값이 없고 결과(반환) 값도 없는 경우

```
1  def hello1():
2      print("Hello", 123, "Python")
```

```
1  hello1()
```

```
Hello 123 Python
```

  - ② 입력(인자) 값이 없고 결과(반환) 값이 있는 경우

```
1  def hello2():
2      print("Hello", 123, "Python")
3      return "안녕하세요?"
```

```
1  a = hello2()
```

```
Hello 123 Python
```

함수 (Function)

- 함수의 형태

③ 입력(인자) 값이 있고 결과(반환) 값이 없는 경우

```
1  def hello3(x):
2      print(x + 123)
```

```
1  hello3(7)
```

```
130
```

④ 입력(인자) 값이 있고 결과(반환) 값도 있는 경우

```
1  def hello4(x):
2      print(x + 123)
3      return x / 2
```

```
1  a = hello4(7)
```

```
130
```

함수 (Function)

- 함수의 결과 값이 없는 경우
  - 함수가 반환하는 실행 결과가 없다는 것은 None을 되돌려 주는 것과 마찬가지이다.

```
1  def hello1():
2      print("Hello", 123, "Python")
```

```
1  a = hello1()
2
3  print(a)
```

```
Hello 123 Python
None
```

```
1  def hello1():
2      print("Hello", 123, "Python")
3      return
```

```
1  def hello1():
2      print("Hello", 123, "Python")
3      return None
```

함수 (Function)

- 매개변수와 인자의 매치
  - 기본적으로, 인자는 순서대로 매개변수에 할당된다.
  - 인자의 값이 복사되어 매개변수로 전달되는 것이며, 인자가 되는 변수와 매개변수 간에는 서로 아무런 관계가 없다.

```
1  def check_maximum(a, b):
2      if a > b:
3          max_value = a
4      elif a < b:
5          max_value = b
6      else:
7          max_value = "값이 같습니다."
8
9      return max_value
```

```
1  print(check_maximum(1, 2))
```

```
2
```

```
1  x = 123
2  y = 0
3  print(check_maximum(x, y))
```

```
123
```

```
1  a = -9
2  b = -9
3
4  print(check_maximum(a, b))
```

```
값이 같습니다.
```

함수 (Function)

- 변수의 지역성 (Local Variable)
  - 함수 내부와 함수 외부는 서로 접근할 수 없는 영역으로서, 함수 안에 있는 변수와 밖에 있는 변수는 서로 관계가 없다.

```
1  a = 1
2
3  def test(a):
4      a += 2
5      print("함수 test 안에 있는 a의 값은", a)
6
7  print("함수 test를 호출하기 전에 외부에 있는 a의 값은", a)
8  test(7)
9  print("함수 test를 호출한 후에 외부에 있는 a의 값은", a)
```

```
함수 test를 호출하기 전에 외부에 있는 a의 값은 1
함수 test 안에 있는 a의 값은 9
함수 test를 호출한 후에 외부에 있는 a의 값은 1
```

함수 (Function)

- 변수의 지역성 (Local Variable)
  - 단, 키워드 **global**을 사용하면 함수 내부에서 함수 외부에 있는 변수에 직접 접근할 수 있다.

```
1  a = 1
2
3  def test(x):
4      global a
5      a += 2
6      print("함수 test가 매개변수로 받은 x의 값은", x)
7      print("함수 test 안에 있는 a의 값은", a)
8
9  print("함수 test를 호출하기 전에 외부에 있는 a의 값은", a)
10 test(7)
11 print("함수 test를 호출한 후에 외부에 있는 a의 값은", a)
```

```
함수 test를 호출하기 전에 외부에 있는 a의 값은 1
함수 test가 매개변수로 받은 x의 값은 7
함수 test 안에 있는 a의 값은 3
함수 test를 호출한 후에 외부에 있는 a의 값은 3
```

함수 (Function)

- 변수의 지역성 (Local Variable)
  - 그러나 global은 지양하고, 함수 결과의 반환을 통해서 함수 외부의 변수 값을 변경하는 것이 바람직하다.

```
1  a = 1
2
3  def test(a):
4      a += 2
5      print("함수 test 안에 있는 a의 값은", a)
6      return a
7
8  print("함수 test를 호출하기 전에 외부에 있는 a의 값은", a)
9  a = test(7)
10 print("함수 test를 호출한 후에 외부에 있는 a의 값은", a)
```

```
함수 test를 호출하기 전에 외부에 있는 a의 값은 1
함수 test 안에 있는 a의 값은 9
함수 test를 호출한 후에 외부에 있는 a의 값은 9
```

함수 (Function)

- 기본적인 내장 함수들 다시보기

		Built-in Functions		
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

※ https://docs.python.org/3/library/functions.html

함수 (Function)

- 기본적인 내장 함수의 예 (6)

함수명	호출할 때의 인자	결과 값	수행하는 일
map	함수 f, 자료 t	자료 목록	자료 구조 t의 항목들을 순서대로 함수 f의 인자로 전달하여 실행한 결과 목록을 반환한다.

```
1  def test(x):
2      return 2 * x
3
4  a = [1, 2, 3, "A"]
5  b = list(map(test, a))
6
7  print(b)
```

```
[2, 4, 6, 'AA']
```

```
1  def make_upper(s):
2      return s.upper()
3
4  seasons = ["Spring", "Summer", "Fall", "Winter"]
5
6  for s in map(make_upper, seasons):
7      print(s)
```

```
SPRING
SUMMER
FALL
WINTER
```

함수 (Function)

- 기본적인 내장 함수의 예 (7)

함수명	호출할 때의 인자	결과 값	수행하는 일
filter	함수 f, 자료 t	자료 목록	자료 구조 t의 항목들을 순서대로 함수 f의 인자로 전달하여 그 실행 결과가 참(True)인 목록을 반환한다.

```
1  def is_pos(x):
2      return x > 0
3
4  a = [3, -10, 999, 54, -267]
5  b = list(filter(is_pos, a))
6
7  print(b)
```

```
[3, 999, 54]
```

```
1  def start_with_s(s):
2      return s[0] == "S"
3
4  seasons = ["Spring", "Summer", "Fall", "Winter"]
5
6  for s in filter(start_with_s, seasons):
7      print(s)
```

```
Spring
Summer
```

함수 (Function)

- 함수 설명 추가하기 (DocString)
  - 함수 정의 시작 직후, 그 함수에 대한 설명을 여러 줄로 된 문자열로 직접 기재할 수 있다.

```
1  def check_maximum(a, b):
2      """2개의 수를 비교하여 큰 값을 반환합니다.
3
4      매개변수 a, b는 둘 다 수치이거나 동일한 자료형이어야 합니다.
5
6      작성일 : 2019.01.01.
7      작성자 : 홍길동"""
8
9      if a > b:
10         max_value = a
11     elif a < b:
12         max_value = b
13     else:
14         max_value = "값이 같습니다."
15
16     return max_value
```

함수 (Function)

- 함수 설명 추가하기 (DocString)
  - 이것을 DocString(Documentation String)이라고 하며, 이 내용을 **help** 함수로 열람할 수 있다.

```
1  help(check_maximum)
```

```
Help on function check_maximum in module __main__:

check_maximum(a, b)
    2개의 수를 비교하여 큰 값을 반환합니다.

    매개변수 a, b는 둘 다 수치이거나 동일한 자료형이어야 합니다.

    작성일 : 2019.01.01.
    작성자 : 홍길동
```