함수의 이해 및 파일의 모듈화

송기태 (kitae040522@gmail.com)
Soongsil Univ. (Computer Science and Engineering)

Content

함수의 개념

- 코딩을 하다 보면 특정 기능을 반복해서 수행해야 할 경우가 있다.
  이때 사용할 수 있는 것이 '함수 (function)'이다.

- 특정 기능을 수행하는 코드의 묶음으로 무엇을 넣으면 처리 후 다시 어떤 것을 돌려주는 기능이다.
  함수를 이용하면 같은 기능을 수행하는 코드를 반복해서 작성할 필요가 없어진다.
  또한 한번 만든 코드는 재사용할 수 있어 코드를 작성이 용이하다.

함수는 왜 쓰는 것일까?

1. 반복적인 프로그래밍을 피할 수 있기 때문
2. 문제를 작게 만들 수 있기 때문 ← 내가 생각하는 함수를 사용하는 이유

함수는 왜 쓰는 것일까?

1. 반복적인 프로그래밍을 피할 수 있기 때문

$$ax^2 + bx + c = 0, (a \neq 0)$$

2차 방정식의 근을 구하기 위해서는 근의 공식을 사용해서 구할 수 있다.

함수는 왜 쓰는 것일까?

1. 반복적인 프로그래밍을 피할 수 있기 때문

```
if __name__ == '__main__':
    def main():
        a, b, c = 1, 2, -8

        r1 = (-b + (b ** 2) - 4 * a * c) ** 0.5
        r2 = (-b - (b ** 2) - 4 * a * c) ** 0.5

        print(f'해는 {r1} 또는 {r2}이다.')

    main()
```

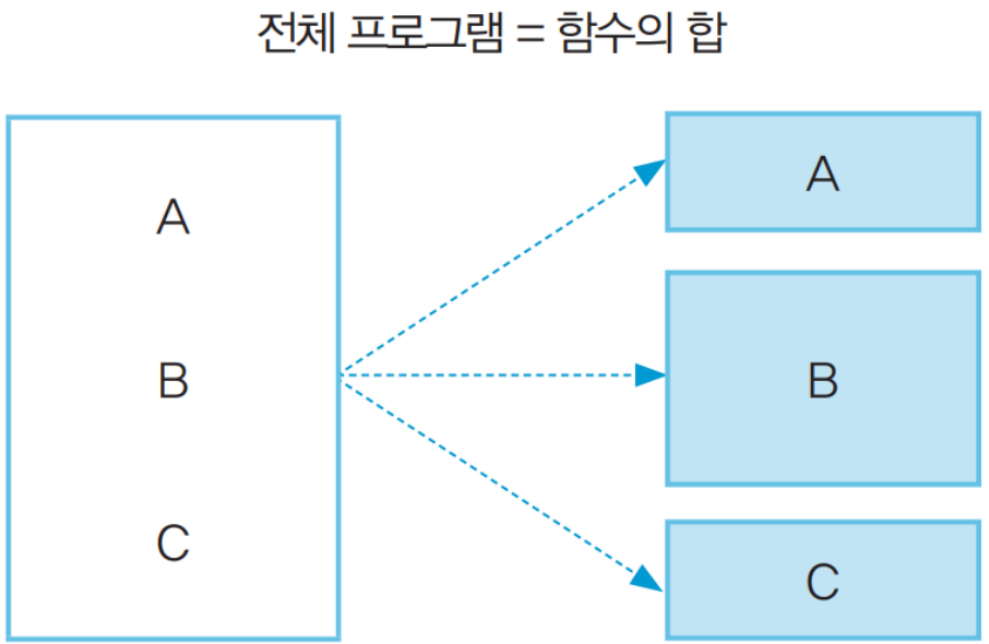
함수는 왜 쓰는 것일까?

1. 반복적인 프로그래밍을 피할 수 있기 때문

한 번에 두 이차방정식을 풀기 위해서는 근의 공식을 두 번 입력해야 할까?

함수는 왜 쓰는 것일까?

2. 문제를 작게 만들 수 있기 때문



함수는 어떻게 만들까?

```
def function_name(arg1, arg2, …):
```

•키워드, •함수명, •인자

함수는 어떻게 만들까?

```
def function_name(arg1, arg2, …):
```

•키워드, •함수명, •인자

함수는 어떻게 만들까?

```
def function_name(arg1, arg2, …):
```

•키워드, •함수명, •인자

함수는 어떻게 만들까? (return이 없는 함수)

```
def greeting():
    print("Hello World!")

if __name__ == '__main__':
    def main():
        greeting()
        greeting()

    main()
```

함수는 어떻게 만들까? (return이 있는 함수)

```
def quadratic_formula(a, b, c):
    r1 = (-b + (b ** 2) - 4 * a * c) ** 0.5
    r2 = (-b - (b ** 2) - 4 * a * c) ** 0.5
    return r1, r2

if __name__ == '__main__':
    def main():
        r1_1, r1_2 = quadratic_formula(1, 2, -8)
        r2_1, r2_2 = quadratic_formula(2, -6, -8)
        print(f'1. 해는 {r1_1} 또는 {r1_2}이다.')
        print(f'2. 해는 {r2_1} 또는 {r2_2}이다.')

    main()
```

함수는 어떻게 만들까? (return이 있는 함수)

```
def quadratic_formula(a, b, c):
    r1 = (-b + (b ** 2) - 4 * a * c) ** 0.5
    r2 = (-b - (b ** 2) - 4 * a * c) ** 0.5
    return r1, r2

if __name__ == '__main__':
    def main():
        r1_1, r1_2 = quadratic_formula(1, 2, -8)
        r2_1, r2_2 = quadratic_formula(2, -6, -8)
        print(f'1. 해는 {r1_1} 또는 {r1_2}이다.')
        print(f'2. 해는 {r2_1} 또는 {r2_2}이다.')

    main()
```

실습 #1

- 다음 조건에 맞는 코드를 작성하여 문장을 출력하여 보자.
- 입력 받은 두 수 중, 가장 큰 값을 구하는 ‘find_max()’ 함수를 코딩해 보자.

실습 #2

- 다음 조건에 맞는 코드를 작성하여 문장을 출력하여 보자.
- ‘divmod()’ 함수를 직접 만들어 보자.
  - divmod는 먼저 넣은 인자를 나중에 넣은 인자로 나눈 몫과 나머지를 리턴 해주는 함수이다.
  - 우리가 함수를 만들 때에는 가장 큰 수를 가장 작은 수로 나눈 몫과 나머지를 리턴 해주자.

모듈의 개념

- '모듈(module)'은 함수나 변수 또는 객체를 정의하는 클래스를 모아 놓은 파일이고, 패키지는 모듈을 모아 놓은 디렉토리다.

- 파이썬에서 기본으로 제공하는 내장함수 외에 추가 기능을 위한 함수나 변수, 클래스는 모듈이나 패키지로 제공되므로 필요한 것을 다운로드한 뒤 import해서 사용한다.

- 코드를 작성할 때 이미 만들어진 모듈을 활용하면 코드를 효과적으로 작성할 수 있으며 대부분의 프로그래밍 언어에서는 모듈이라는 개념을 사용한다.

모듈의 개념

작성할 프로그램에 수학적인 계산 기능이 필요하다면 'math'라는 모듈을 불러와서 사용하면 된다.

```
import math

if __name__ == '__main__':
    def main():
        print(math.pi) # math 모듈에 정의되어 있는 pi 변수를 사용

    main()
```

모듈의 개념

작성할 프로그램에 랜덤 기능이 필요하다면 'random'라는 모듈을 불러와서 사용하면 된다.

```
import random

if __name__ == '__main__':
    def main():
        print(random.randint(1, 10)) # random 모듈에 정의되어 있는 randint() 함수를 호출

    main()
```

모듈 불러오기

```
import math

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

- ‘import 모듈’로 불러와 ‘모듈명.변수’, ‘모듈명.함수( )’와 같은 형식을 이용하여 사용한다.
- 이 경우 호출하기 위해 계속해서 모듈명을 앞에 써줘야 하므로 코드를 작성할 때 불편하다.

모듈 불러오기

```
from math import sqrt

print(sqrt(9))
```

- 이런 식(‘from 모듈 import 이름’)으로 불러오게 되면, 모듈에 사전 정의 된 변수와 함수, 그리고 클래스를 모듈 명 없이 직접 호출해서 이용할 수 있다.

모듈 불러오기 (별명 붙이기)

```
import time as t

print(t.time())
```

- 매번 코드에서 모듈명을 입력하기가 번거롭고 모듈명이 길다면 입력이 더욱 번거로울 수 있다.
- 이런 문제는 다음과 같이 모듈명에 새로운 이름(별명)을 선언하면 해결할 수 있다.

Thank You!

송기태 (kitae040522@gmail.com)
Soongsil Univ. (Computer Science and Engineering)