파이썬 프로그래밍

함수와 모듈

목차

함수란 무엇일까?

수학에서 배운 함수를 떠올려보자.

X값에 따라서 새로운 Y값이 도출이 된다.

하지만 파이썬에서는 받은 인자값(X)에 Y값을 도출 시키는 것도 중요하지만 반복되어서 많이 사용되는 작업을 줄이기 위해 '반복적으로 사용되는 가치 있는 부분'을 한 뭉치로 묶어서 사용자에게 편리함을 준다는 것에도 큰 의미가 있다.

함수의 장점

- 큰 프로그램을 나누어 작성하여 구조성을 갖게 할 수 있다
- 다른 프로그램에서 함수를 재사용하게 할 수 있다
- 코드의 가독성이 증가한다
- 프로그램 수정 시 일부 함수만 수정하면 되기 때문에 유지보수가 쉬워진다
- 이미 개발된 함수를 사용하면 프로그램 개발의 시간과 비용을 절약할 수 있다.

파이썬 내장함수(Built-in Function)란?

파이썬에서 제공하는 함수
`max` , `min`, `input`, `print`, `range` 등등 여러 가지 함수를 제공한다.

(cf) https://docs.python.org/ko/3/library/functions.html

MAX/MiN/INPUT/PRINT/RANGE 함수

```
big = max('Helloworld') # ASCII 값으로 제일 큰 w 출력
print(big)

tiny = min('Helloworld') #ASCII 값으로 제일 작은 H 출력
print(tiny)

#입력받은 문자열을 minmax 변수에 저장
minmax = input('MAX / MIN를 구하고 싶은 문자열을 입력하시오 :')
print(max(minmax) ,' / ', min(minmax))

#랜덤 숫자 출력하기
import random

#0~9사이의 값 중에서 10번 랜덤 출력
for roll in range(10):
  print(random.randrange(0,9), end=' ')
```

사용자 지정 함수(user-defined Function)란?

```
def function_name(x1,x2, ...):
  code1
  code2
  ...
  return n1[,n2,...]
```

- 함수 정의 키워드 `def`
- 함수 이름 `function_name`(사용자 임의로 설정)
- 함수에서 사용할 매개변수 `(x1, x2 ..)` (생략 가능)
- 함수 내의 코드블록 `code1 ,code2..` (반드시 들여쓰기!)
- 함수 내에서 코드를 수행한 결과 반환 `return n1[, n2..]`(생략 가능)

간단한 함수 만들어보기

```
#주소를 입력받고 출력하는 함수를 만들어보기

def print_adress():
  address = input('우리 집 주소는?')
  print('우리 집 주소는',address, '입니다')

print_adress()
```

```
#인자(Parameter)가 있는 함수 만들어보기

def hello_user(name):
  print(f'Hello, {name}! Welcome to the program.')

hello_user("LuckyVicky")
```

간단한 함수 만들어보기

```
#return 값이 있는 함수 만들어보기

#사각형 넓이 구하기
def calc_area(length, width):

  area = length * width
  return area

rect_length = int(input('사각형의 밑면 :')) #input으로 받으면 문자열이기 때문에
rect_width = int(input('사각형의 높이 :'))  #int()함수를 통해서 형변환을 시켜줘야함
area =calc_area(rect_length, rect_width)

print(f'사각형의 넓이는 {area}입니다') #print('사각형의 넓이는',area,'입니다')
```

매개변수(parameter)와 반환문(return) 실습

```
def add(a,b):
  #return 작성해보기

def minus(a,b):
  #return 작성해보기

def multi(a,b):
  #return 작성해보기

def divide(a,b):
  #return 작성해보기

A = 10
B = 2

# ?에 값 넣기
print(add(?,?),minus(?,?),multi(?,?),divide(?,?))
```

매개변수(parameter)와 반환문(return) 실습 답안

```
def add(a,b): #cf)함수를 정의 할 때 넣는 값은 매개변수(parameter)라 한다
  return a+b

def minus(a,b):
  return a - b

def multi(a,b):
  return a * b

def divide(a,b):
  return a / b

A = 10
B = 2

#cf)함수를 자용할 때 넣는 값은 인자(argument)라 한다
print(add(A,B),minus(A,B),multi(A,B),divide(A,B))
```

지역변수(Local Scope)와 전역변수(Global Scope)

1. 지역변수(Local Scope)

함수 내에서 선언되고 사용되는 변수

2. 전역변수(Global Scope)

함수 밖에서 선언된 변수

```
k = 10

def add(a,b):
  c = a + b
  c = c * k

print(k, add(2,3)) #오류 X
print(a,b,c)  #오류
```

예제

다음 코드를 보고 오류가 나는 이유를 생각해보고, 여러가지 방법으로 고쳐보자

```
import math

def calculate_area (radius):
  result = math.pi * radius**2
  return result

r = float(input("Circle's radius: "))
area = calculate_area(r)
print(result)
```

답안

```
def calculate_area(radius):
    result = math.pi * radius**2
    return result

r = float(input("Circle's radius: "))
area = calculate_area(r)
print(area)  # 정확한 변수 넣기
```

```
def calculate_area():
    result = math.pi * g_radius **2 #직접 전역변수 참조
    return result

g_radius = float(input("Circle's radius: "))
area = calculate_area()
print(area)
```

모듈이란?

- 파이썬 함수나 변수 또는 클래스들을 모아놓은 스크립트 파일
- 파이썬은 수많은 개발자들에 의해서 개발된 많은 모듈이 있음
- 만들어진 모듈을 가져올 때는 `import`와 함께 모듈 이름을 써준다
- 사용할 때는 모듈 이름에 점(.)을 찍은 후 모듈 안의 구성요소를 작성

`import` 사용법

```
import 모듈이름 (as 별명)
import 모듈1, 모듈2, ...
import 모듈.변수
import 모듈.함수()
import 모듈.클래스()

#필요한 부분만 import 하고 싶을 때
from 모듈이름 import 변수
from 모듈이름 import 함수
from 모듈이름 import 클래스
from 모듈이름 import 변수, 함수, 클래스
from 모듈이름 import * #모듈의 모든 변수, 함수, 클래스를 가죠옴
```

파이썬 표준 라이브러리(python standard libaray)란?

- 파이썬 설치와 함께 제공되는 모듈
- 문자열과 텍스트 처리를 위한 모듈, 이진 데이터 처리, 날짜, 시간, 배열 등의 자료형 처리를 위한 모듈, 수치 연산과 수학 함수 모듈, 파일과 디렉터리 접근, 유닉스 시스템의 데이터베이스 접근을 위한 모듈, 데이터 압축, 그래픽 모듈 등 100여 가지 이상의 표준 라이브러리들이 있다

(ex)os, random, sys, math, datetime, pandas, numpy

(cf) https://docs.python.org/ko/3/library/index.html

파이썬 표준 라이브러리 예시

```
import os
# 지금 작업을 수행하고 있는 위치를 파악할 수 있다
current_directory = os.getcwd()
print(current_directory)
```

```
import time
print('now')
time.sleep(3)# time 모듈의 sleep함수를 실행중인 쓰레드 일시 중지
print('3초 후')
```

```
from datetime import datetime
#datetime 모듈을 통해서 현재 시간을 불러올 수 있다
current_time = datetime.now()
print(current_time)
```

패키지(package)란?

- 패키지는 모듈의 묶음이다. 일종의 디렉터리처럼 하나의 패키지 안에 여러 개의 모듈이 있고, 이 모듈들은 서로 포함관계를 가지며 거대한 패키지를 만든다.
- 즉 다시말해서 함수들과 변수들이 모여 모듈을 만들고 그 모듈들이 모여서 패키지를 만든다.

모듈 실습하기

1. 모듈 작성하기

- 주피터 노트북이라면 지금 실행하고 있는 파일이 있는 디렉토리로 이동한다. `new`를 누르고 `Text File`을 생성한다.
  - math_operation.py라는 이름으로 다음 장에 있는 내용을 작성한다 (들여쓰기 주의하기!!)
- 주피터 노트북이 아니라면 메모장을 실행한다.
  - 다음 장에 있는 내용을 작성하고 math_operation.py라는 이름으로 저장 후 실행 파일이 있는 곳에 저장한다.

```python
#같은 디렉토리 안에 math_operation.py라고 저장하기
def add(a, b):

    return a + b

def subtract(a, b):

    return a - b

def multiply(a, b):

    return a * b

def divide(a, b):

    if b == 0:
        #분모가 0일 때 오류발생하게 만들기
        raise ValueError("Cannot divide by zero.")
    return a / b
```

모듈 실습하기

2-1. 모듈 사용하기

```
#math_operation은 너무 긴 관계로 mo라고 축약해서 사용하기 위한 as mo
import math_operation as mo

num1 = float(input('첫번 째 숫자를 입력해주세요'))
num2 = float(input('두번 째 숫자를 입력해주세요'))

#as mo를 사용하지 않았다면
#math_operation.add(), math_operation.subtract(),
#math_operation.multiply(), math_operation.divide() 를 사용했어야함
print(mo.add(num1,num2),mo.subtract(num1,num2),mo.multiply(num1,num2),mo.divide(num1,num2))
```

모듈 실습하기

2-2. 모듈 사용하기

```
#math_operation 모듈에서 함수 직접 불러오기
from math_operation import add, subtract, multiply, divide



num1 = float(input('첫번 째 숫자를 입력해주세요'))
num2 = float(input('두번 째 숫자를 입력해주세요'))

#mo.함수() 즉 math_operation.함수()를 사용하지 않아도 된다
#주의할 점은 모듈에 있는 함수와 같은 이름의 함수를 사용하지 않도록 주의한다.
print(add(num1,num2), subtract(num1,num2), multiply(num1,num2), divide(num1,num2))
```

Thank you

Q / A