기초 공학 설계 (CSE2003)

Introduction to Engineering Design

12-1 함수의 응용

Python 라이브러리

- 라이브러리(library)
  특정 기능의 함수들을 각 프로그램 파일에 저장하여 두고, 이들 함수를 다른 프로그램에서 사용할 수 있도록 하는 방식

- Python 라이브러리
  - 내장(Built-in) 라이브러리
    : 파이썬이 기본적으로 제공하는 것으로 import문을 이용해 적재(load) 하지 않아도 사용할 수 있음
  - 빌트인이 아닌 라이브러리(표준 모듈)
    : 파이썬 플랫폼에 기본적으로 내장되어 제공된다는 점에서 같지만, 해당 모듈은 import문으로 명시적으로 적재(load)한 후에 사용할 수 있음

모듈 정의

- ★ 파이썬의 모듈(module) 정의
  - 함수나 변수들을 모아 놓은 파일
  - 다른 파이썬 프로그램에서 호출해서 사용할 수 있게 만들어진 파이썬 프로그램 파일
  - 각각의 소스 파일을 일컬어 모듈이라 함
- ★ 표준 모듈(라이브러리) : 파이썬 설치시 함께 설치되는 모듈
  - calendar, time, random, math, turtle 모듈 등....
- ★ 사용자 생성 모듈 : 프로그래머가 직접 작성한 모듈

모듈 사용 : import

★ 파이썬 제공 표준 모듈 : random 모듈

```
import random

print( random.randint(1, 6) )  # 1~ 6 사이의 임의의 정수를 반환
print( random.randint(1, 6) )

myList = [ "red", "green", "blue" ]
color = random.choice( myList )  # 리스트에서 임의의 원소를 선택해서 반환
print(color)
```

출력

```
5
2
blue
```

모듈 사용 : import

★ 파이썬 제공 표준 모듈 : calendar 모듈

```
import   calendar

cal = calendar.month(2020, 12)      # 인수로 지정한 년,월의 달력을 반환
print(cal)
print(type(cal))
```

출력

```
 December 2020
Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31

<class 'str'>
```

사용자 정의 모듈

★ 다음 프로그램을 모듈로 구성

```
def add_multiply(x,y):
    sum = x + y
    mul = x * y
    return sum, mul      # 반환값 2개를 튜플로 반환

a = int(input('Enter a : '))
b = int(input('Enter b : '))
m, n = add_multiply(a,b) # 변수 m은 a+b의 값, 변수 n은 a*b의 값을 할당 받음
print(m,n)
```

- 위의 스크립트 파일을 사용자 정의 모듈로 아래와 같이 구성
- ① **userF.py** 파일
  : add_multiply(x,y) 함수 정의를 포함하는 모듈 파일
- ② **test.py** 파일
  : 모듈 userF을 import하여 add_multiply() 함수를 호출하는 파이썬 스크립트 파일

사용자 정의 모듈

test.py 파일

```
from userF import *

a = int(input('Enter a : '))
b = int(input('Enter b : '))
m, n = add_multiply(a,b) # 변수 m은 a+b의 값, 변수 n은 a*b의 값을 할당 받음
print(m,n)
```

userF.py 파일

```
def add_multiply(x,y):
    sum = x + y
    mul = x * y
    return sum, mul      # 반환값 2개를 튜플로 반환
```

- 위와 같이 두개의 파일을 작성한 후, **test.py** 파일을 실행하면 앞 장과 같은 결과를 얻을 수 있음

사용자 정의 모듈

★ calculator 모듈 만들기

- calculator.py 파일 작성
- 해당 파일에 구현 하고자 하는 여러 함수들을 정의

calculator.py

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

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

def   multiply(a, b):
        return a*b;

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

test1.py

```
import   calculator

print( calculator.plus(10, 5) )
print( calculator.minus(10, 5) )
print( calculator.multiply(10, 5) )
print( calculator.divide(10, 5) )
```

test2.py

```
from calculator import *

print( plus(10, 5) )
print( minus(10, 5) )
print( multiply(10, 5) )
print( divide(10, 5) )
```