



# 모듈

최석재

lingua@naver.com

# 파이썬 모듈

- 모듈이란 하나의 파이썬 파일(.py)을 말한다
- 모듈 안에는 변수, 함수 등 다양한 구성요소가 들어갈 수 있다
- 모듈을 만드는 이유는 이들을 재사용 하기 위해서이다
- 모듈은 사용하려고 하는 파일과 같은 폴더에 있어야 한다
- myMath.py 라는 모듈을 만들고, "77\_module.py"에서 사용하려면,

6\_모듈

77\_module.py

myMath.py

- 왼쪽과 같이 두 파일이 같은 폴더 내에 있어야 한다

# myMath.py

```
pi = 3.14159265359
```

```
def sum1toN(endNum):  
    '''1부터 endNum까지의 수를 더해주는 함수  
    endNum: 마지막 더할 수'''  
    result = 0  
    for i in range(endNum):  
        result += i+1  
    return result
```

```
def multiply1toN(endNum):  
    '''1부터 endNum까지의 수를 곱해주는 함수  
    endNum: 마지막 곱할 수'''  
    result = 1  
    for i in range(endNum):  
        result *= i + 1  
    return result
```

# 77\_module.py

```
import myMath
```

```
print("pi value: ", myMath.pi)
```

```
sum = myMath.sum1toN(10)  
print(sum)
```

```
mul = myMath.multiply1toN(10)  
print(mul)
```

```
pi value:  3.14159265359  
55  
3628800
```

# 내장 모듈

- 파이썬에는 다양한 내장 모듈이 있다
- math.py 의 주요 함수는 다음과 같다

```
pi value:  3.141592653589793
자연상수:  2.718281828459045
log value base e:  1.0
log value base 10:  1.0
root value:  10.0
```

```
import math
```

```
print("pi value: ", math.pi)
print("자연상수: ", math.e)
print("log value base e: ", math.log(math.e))
print("log value base 10: ", math.log10(10))
print("root value: ", math.sqrt(100))
```

# 모듈 불러오기

```
import myMath  
print(myMath.sum1toN(10))
```

*# 모듈을 임포트한다  
# 모듈 이름을 쓰고, 함수 또는 변수를 부른다*

```
from myMath import sum1toN  
print(sum1toN(10))
```

*# 모듈의 특정 함수만을 임포트한다  
# 모듈 이름을 쓰지 않아도 된다*

```
from myMath import pi  
print(pi)
```

*# 모듈의 특정 변수만을 임포트한다  
# 모듈 이름을 쓰지 않아도 된다*

```
from myMath import pi, sum1toN
```

*# 두 개 이상을 한 번에 부를 수 있다*

```
from myMath import *
```

*# 해당 모듈의 모든 것을 부른다*

- 모듈이 불러와지지 않는다면 기본 경로가 다르기 때문이다
- 방법 1: 아래가 가리키는 곳에 파일을 둔다

```
import os  
print(os.getcwd())
```

- 방법 2: 아래와 같은 코드로 기본 경로를 변경한다

```
import os  
os.chdir("C:/Users/lingu/OneDrive/Python Projects/.../")
```

# 별명으로 사용하기

- 모듈의 이름을 다 쓰지 않고 간단한 별명으로 사용할 수 있다
- "import 모듈명 as 별명"의 형식으로 사용한다

```
import myMath as mm
```

```
print(mm.sum1toN(10))
```



# 내장 모듈 - keyword

- 파이썬의 주요 내장 모듈을 알아본다
- keyword 모듈은 파이썬에서 사용되고 있는 예약어의 종류를 보여준다

```
import keyword
```

```
print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class',  
'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for',  
'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal',  
'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

# 내장 모듈 - random

- 수를 랜덤하게 만들어야 할 때가 종종 있다
- random 모듈은 난수(랜덤 수)를 만들어준다
- randint() 함수는 정수로 된 난수를 만든다

```
import random
```

```
selected = random.randint(1, 30)  
print(selected)
```

# 1 ~ 30 중 하나를 추출

```
for i in range(6):  
    number = random.randint(1, 45)  
    print(number, end=' ')
```

*# 출력 후 마지막에 공백을 만든다 (기본값은 \n)*

25 21 18 22 18 13

# shuffle 함수

```
['a', 'b', 'c', 'd', 'e']  
['e', 'a', 'b', 'c', 'd']  
e
```

- shuffle() 함수는 순서를 무작위로 섞는다

```
import random
```

```
card = ['a', 'b', 'c', 'd', 'e']  
print(card)
```

```
random.shuffle(card)  
print(card)
```

*# 무작위로 섞음. 셔플*

```
print(random.choice(card))
```

*# 무작위로 하나의 항목을 선택*

# sample 함수

```
['a', 'b', 'c', 'd', 'e']  
['c', 'b', 'd']  
[18, 7, 9, 23, 29, 3]  
[3, 7, 9, 18, 23, 29]
```

- sample() 함수는 여러 개의 항목을 랜덤하게 추출한다

```
import random
```

```
card = ['a', 'b', 'c', 'd', 'e']  
print(card)
```

```
selected = random.sample(card, 3)  
print(selected)
```

*# 무작위로 3개를 추출함*

*### 로또 번호 생성기*

```
lotto = random.sample(range(1, 46), 6)  
print(lotto)
```

```
lotto.sort()  
print(lotto)
```

*# 리스트 정렬 (원본변경)*

# random 함수

- 0.0 ~ 1.0 사이의 난수를 만들게 될 때가 있다
- random() 함수는 실수로 된 난수를 만든다

```
import random
```

```
print(random.random())
```

*# 0.0 ~ 1.0 사이의 난수 생성*

```
print(random.uniform(0, 10))
```

*# 0.0 ~ 10.0 사이의 난수 생성*

# 내장 모듈 - time

```
start: 1656295839.4370155
몇 번을 반복할까요?: 100000
*****
3.617486000061035 초 경과
start: 1656295843.0545015
*****
0.00599360466003418 초 경과
```

```
import time
```

```
start = time.time()                                # 1970년 1월 1일 0시부터 지금까지의 경과 시간(seconds)
print("start: ", start)
```

```
def manyloop(num):
    for i in range(num):
        if i % 10000 == 0:                          # 나머지 연산자를 이용하여 10000 번째마다 *를 찍음
            print("*", end="")
    print()
```

```
number = int(input("몇 번을 반복할까요?: "))        # 문자열을 숫자로 변환. 100000 입력
manyloop(number)
```

```
end = time.time()
print(end - start, '초 경과')
```

# 연습문제 1

- 앞의 내용에서 반복 횟수를 사용자 입력으로 받지 말고, 100,000 회가 입력되도록 고정시킨 프로그램을 작성하세요
- 두 프로그램의 시간이 어느 정도 차이가 나는지 확인하시오



# ctime() 함수

- ctime() 함수는 현재 시간을 문자열로 알려준다

```
import time
```

```
current = time.ctime()  
print(current)
```

```
current_list = current.split(' ') # 문자열을 공백 단위로 분리한다  
print(type(current_list))        # 결과는 리스트이다
```

```
for t in current_list:  
    print(t)
```

# sleep() 함수

- sleep() 함수는 시스템의 속도를 늦춘다

```
import time
```

```
for t in range(6):  
    print(time.ctime())  
    time.sleep(1)
```

*# 1초 씩. 0.5와 같이 실수도 가능하다*