성균관대학교 SUNGKYUNKWAN UNIVERSITY(SKKU)

컴퓨팅사고력 향상과 SW코딩 능력 배양을 위한
컴퓨팅사고와 SW코딩

모 듈

JEE, Jung Eun
rosaliejee@skku.edu

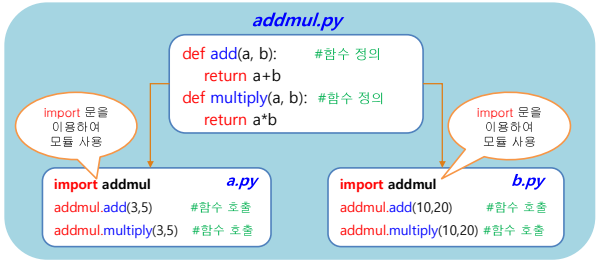
학습 목표

모듈

1. 모듈을 이해하고 여러 방법으로 실행할 수 있다.
2. 패키지를 설치하고 사용할 수 있다.
3. 다양한 모듈 사용을 이해하여 활용할 수 있다.

모듈 개념

- **모듈(module)**
  - 함수의 집합으로 파일 단위로 작성된 코드
  - 함수가 정의되어 있는 파일 자체를 재사용 하기 위함
  - 라이브러리 함수가 포함된 모듈 불러오기
    - import 모듈명
      - 모듈명.변수명
      - 모듈명.함수명()



모듈 실행

- Ex1. 사용할 파일명
  - addmul.py
  - a.py
  - b.py

addmul.py

```
def add(a, b):      # 함수 정의
    z = a+b
    return z

def multiply(a, b):  # 함수 정의
    z = a*b
    return z
```

a.py

```
import addmul

result = addmul.add(3,7)        #함수 호출
print(result)
result = addmul.multiply(3,7)   #함수 호출
print(result)
```

b.py

```
import addmul

result = addmul.add(10,20)      #함수 호출
print(result)
result = addmul.multiply(10,20) #함수 호출
print(result)
```

[실행1]

```
== RESTART: C:\Users\JEE\Desktop\addmul.py ==
>>>
===== RESTART: C:\Users\JEE\Desktop\a.py ====
10
21
>>>
===== RESTART: C:\Users\JEE\Desktop\b.py ====
30
200
>>>
```

모듈 실행

- Ex2. 사용할 파일명 : addmul.py

[실행2]

```
== RESTART: C:₩Users₩JEE₩Desktop₩addmul.py ==
>>>
>>> import addmul          import 모듈명
>>>
>>> addmul.add(3,8)       ① 모듈명.함수명(인수1,인수2)
11
>>> total = addmul.add     ② 변수 = 모듈명.함수명
>>> total(3,8)                  변수(인수1, 인수2)
11
>>>
>>> addmul.multiply(3,8)  ① 모듈명.함수명(인수1,인수2)
24
>>> mul = addmul.multiply  ② 변수 = 모듈명.함수명
>>> mul(3,8)                    변수(인수1, 인수2)
24
>>>
```

[실행3]

```
== RESTART: C:₩Users₩JEE₩Desktop₩addmul.py ==
>>>
>>> import addmul as ad    import 모듈명 as alias
>>>                                  =별칭
>>> ad.add(5,82)
87
>>> ad.multiply(4,7)
28
>>> ad.add(5,9)
14
>>> addmul.add(5,9)
Traceback (most recent call last):
  File "<pyshell#21>", line 1, in <module>
    addmul.add(5,9)
NameError: name 'addmul' is not defined
>>>
```

별칭말고 다른걸로 돌아오면 안됨

[실행4]

```
== RESTART: C:₩Users₩JEE₩Desktop₩addmul.py ==
>>>
>>> from addmul import *   from 모듈명 import *
>>>
>>> add(55,9)
64
>>> multiply(30,8)
240
>>>
```

addmul에서 모든 함수를 가져오겠습니다

[실행5]

```
== RESTART: C:₩Users₩JEE₩Desktop₩addmul.py ==
>>>
>>> from addmul import add
>>>                    from 모듈명 import 함수명
>>> add(11,8)
19
>>> from addmul import add, multiply
>>>
>>> add(4,5)
9
>>> multiply(4,5)
20
>>> from addmul import multiply as mul
>>>
>>> mul(5,90)      from 모듈명 import 함수명 as alias
450
>>> multiply(5,90)
450
>>>
```

addmul에서 특정한 함수만 가져오겠습니다

모듈 실행

- 사용자가 정의한 모듈 실행시, 에러가 발생했을 경우 해결 방법

```
>>> import addmul  #addmul 모듈 불러오기
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    import addmul
ModuleNotFoundError: No module named 'addmul'
>>>
>>> import sys  #시스템 관리 기본 모듈
>>> sys.path    #현재 파이썬 연결 폴더 표시
['', 'C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩Lib₩₩idlelib'
, 'C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩python39.zip', '
C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩DLLs', 'C:₩₩Users₩₩
JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩lib', 'C:₩₩Users₩₩JEE₩₩AppDat
a₩₩Local₩₩Programs₩₩Python₩₩Python39', 'C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Program
s₩₩Python₩₩Python39₩₩lib₩₩site-packages']
>>>
>>> sys.path.append('C:₩₩Users₩₩JEE₩₩Desktop')  #현재 파이썬 연결 폴더 추가
>>>
>>> sys.path                    \ 이 2개!
['', 'C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩Lib₩₩idlelib'
, 'C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩python39.zip', '
C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩DLLs', 'C:₩₩Users₩₩
JEE₩₩AppData₩₩Local₩₩Programs₩₩Python₩₩Python39₩₩lib', 'C:₩₩Users₩₩JEE₩₩AppDat
a₩₩Local₩₩Programs₩₩Python₩₩Python39', 'C:₩₩Users₩₩JEE₩₩AppData₩₩Local₩₩Program
s₩₩Python₩₩Python39₩₩lib₩₩site-packages', 'C:₩₩Users₩₩JEE₩₩Desktop']
>>>
>>> import addmul
>>>
>>> addmul.add(4,13)
17
```


유용한 모듈

time 모듈

- 시간에 대해 다양한 형식으로 표시해주는 모듈
  - **time()**
    - 1970년 1월 1일 자정 이후의 시간을 측정하여 초 단위로 출력하는 함수
  - **asctime()**
    - 현재 날짜와 시간을 문자열 형태로 출력하는 함수 *cf.* ctime()

```
>>> import time
>>>
>>> time.time()    #시간 측정하여 반환
1619792256.8530035
>>>
>>> start = time.time()
>>> end = time.time()
>>>
>>> exetime = end - start
>>> exetime
0.345198392868042
>>> int(exetime)
0
>>>
```

```
>>> time.asctime()   #현재 날짜/시간
'Fri Apr 30 23:21:48 2021'
>>>
>>> birthinfo=(2000,9,16,11,34,55,3,0,0)  → #튜플 값(연도,월,일,시간,분,초,요일,0,0)
>>>
>>> time.asctime(birthinfo)   #특정 날짜/시간
'Thu Sep 16 11:34:55 2000'
>>>
```


유용한 모듈

- **localtime()**
  - 현재 날짜와 시간을 튜플 객체 형태로 변환하는 함수

```
>>> time.localtime()  요소0 요소1 요소2 ....
time.struct_time(tm_year=2021, tm_mon=4, tm_mday=30, tm_hour=23,
tm_min=28, tm_sec=5, tm_wday=4, tm_yday=120, tm_isdst=0)
>>>
>>> ptime = time.localtime()
>>>
>>> year = ptime[0]
>>> month = ptime[1]
>>> day = ptime[2]          #ptime[인덱스]의 값을 변수에 저장
>>>
>>> print(year,'년',month,'월',day,'일')
2021 년 4 월 30 일
>>>
>>>
>>> hour = ptime[3]
>>> minute = ptime[4]
>>> second = ptime[5]       #ptime[인덱스]의 값을 변수에 저장
>>>
>>> print(hour,'시',minute,'분',second,'초')
23 시 28 분 10 초
>>>
```

- tm_year=년
- tm_mon=월
- tm_mday=일
- tm_hour=시
- tm_min=분
- tm_sec=초
- tm_wday=요일(0:월요일, 6:일요일)
- tm_yday=1월1일부터 경과한 일수
- tm_isdst=서머타임(0 또는 -1)

- **sleep(second)**
  - 지정된 초(second) 만큼 현재 동작 중인 프로세스를 정지시키는 함수

유용한 모듈

- **random 모듈** (랜덤 가챠 돌릴때 사용하는거)
  - 난수(random variable)를 발생할 때 사용하는 모듈

1) randint(start, stop)

범위: 스타트~스톱

- 정수 범위(start~stop)의 난수를 생성해주는 함수

```
>>> import random
>>>
>>> random.randint(1,5)
2
>>> for i in range(3):
        print(random.randint(1,5))

1
4
5
>>> for i in range(20):
        print(random.randint(0,9),end=' ')

7 4 5 3 1 0 9 0 3 3 7 0 2 2 6 0 7 9 5 8
>>> for i in range(10):
        print(random.randint(11,20),end=' ')

18 18 17 18 12 14 18 11 18 20
>>>
```

1~5중 랜덤

#1~5 중 난수 생성

1~5중 랜덤3개

#0~9 중 난수 생성

#11~20 중 난수 생성

2) randrange(start, stop[,step])

범위: 스타트~스톱-1

- range(start, stop, step) 정수 범위의 난수를 생성해주는 함수

```
>>> import random
>>>
>>> random.randrange(0,20,3)
9
>>> for i in range(3):
        print(random.randrange(0,20,3))

12
18
6
>>> for i in range(10):
        print(random.randrange(0,20,3),end=' ')

6 12 9 0 0 18 3 18 12 15
>>> for i in range(15):
        print(random.randrange(5,100,5),end=' ')

15 50 15 70 80 95 60 10 50 20 60 80 95 75 5
>>>
```

#0~19 중 3씩 증가하는 정수 중 난수 생성

#5~99 중 5씩 증가하는 정수 중 난수 생성

3) random( )

- 0~1 사이의 난수를 생성해주는 함수

```
>>> import random
>>>
>>> random.random() #0~1 사이의 난수 생성
0.690954084660592
>>>
>>> for i in range(3):
        print(random.random())

0.8721538422844732
0.40328811533008524
0.16603325464327268
>>>
>>> for i in range(3):
        print(random.random()*10)
                          #0~9 사이의 난수 생성
2.1008190522945047
7.872155467831492
5.1774725623084805
>>>
>>> for i in range(3):
        print(int(random.random()*10))
                          #0~9 사이 난수 생성 후 정수로 변환
0
4
4
>>>
```


유용한 모듈

4) choice( )

- 시퀀스의 항목을 랜덤하게 반환하는 함수

```
>>> import random
>>>
>>> fruits = ['apple','orange','mango']
>>>
>>> random.choice(fruits)
'orange'
>>>
>>> for i in range(7):
        print(random.choice(fruits))

mango
mango
mango
apple
mango
apple
mango
>>>
```

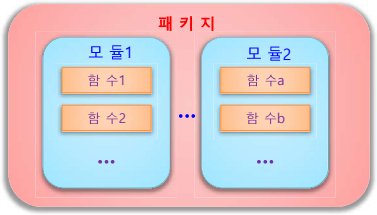
5) shuffle( )

- 리스트의 항목을 랜덤하게 섞어서 반환하는 함수

```
>>> import random
>>>
>>> fruits = ['apple','orange','mango','banana','cherry','kiwi']
>>>
>>> random.shuffle(fruits)
>>>
>>> print(fruits)
['banana', 'cherry', 'apple', 'kiwi', 'orange', 'mango']
>>>
>>> number = [[a] for a in range(1,8)]
>>>
>>> random.shuffle(number)
>>>
>>> print(number)
[[7], [3], [4], [1], [5], [6], [2]]
>>>
```


패키지 개념

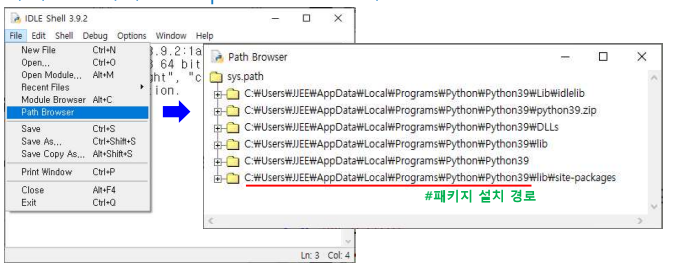
- **패키지(package)**
  - 여러 개의 모듈을 하나로 묶은 단위
  - 모듈의 컬렉션을 의미함
  - 패키지 안에 있는 모듈을 사용하는 문법
    - import 패키지명.모듈명
      - 패키지 내에 있는 모듈 불러오기
      - **패키지명.모듈명.변수명**
      - **패키지명.모듈명.함수명()**



패키지 설치

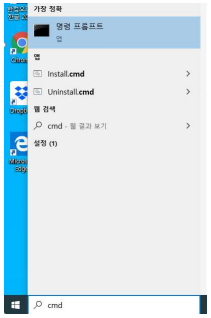
- **pip (pip installs packages)**
  - 패키지 관리자로 파이썬에서 작성된 패키지들을 설치할 때 필요한 명령어
    1) **python -m pip install -U pip setuptools** ***또는*** **python -m pip install --upgrade pip**
    2) **python –m pip install** 패키지명 ***또는*** **pip install** 패키지명 #pip 버전 업그레이드 후 명령 가능

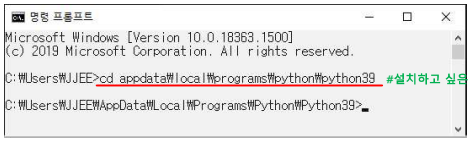
1) 파이썬 창에서 File → path browser 클릭 #패키지 설치 경로 확인



패키지 설치

2) 윈도우 키 → cmd 창(명령 프롬프트) 열기

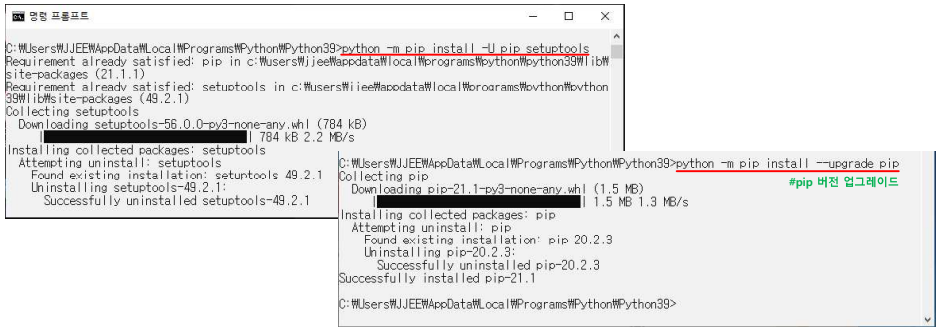




❖ 패키지 설치 경로로 이동

패키지 설치

2) pip 버전 업그레이드 명령어
- python -m pip install -U pip setuptools
- python -m pip install --upgrade pip

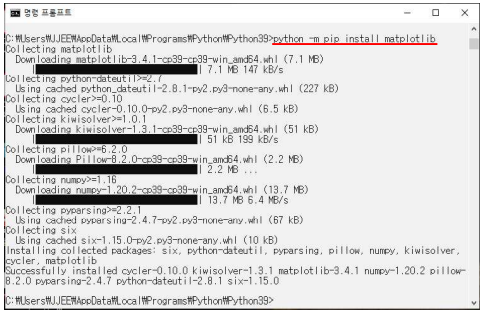


패키지 설치

- **matplotlib 패키지(package)**
  - 데이터를 차트나 플롯(Plot)으로 그려주는 라이브러리 패키지
  - 데이타 시각화(Data Visualization) 패키지
  - 패키지 설치하기
    - **pip install** 설치할_패키지명
  - 설치된 패키지 제거
    - **pip uninstall** 설치된_패키지명

matplotlib 패키지 설치

- matplotlib 패키지 설치
  - **pip install** matplotlib



matplotlib 패키지 예제

- matplotlib 패키지 활용 Ex.

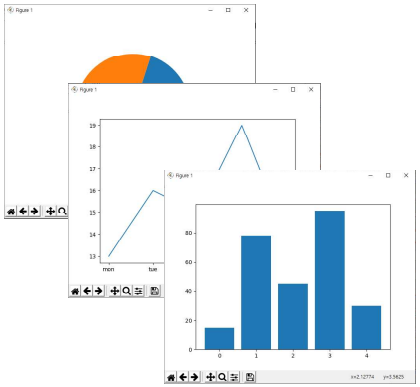
```
#matplotlib 패키지 사용

import matplotlib.pyplot as plt

#pie 차트 그리기
plt.pie([20, 50, 30 ])
plt.show()                  #그래프 보이기

#라인 플롯(plot) 그리기
x=['mon','tue','wed','thur','fri']
y = [13, 16, 15, 19, 14]
plt.plot(x, y);(세미콜론 없어도 됨)
plt.show()                  #그래프 보이기

#막대 차트 그리기
data = [15, 78, 45, 95, 30]
plt.bar(range(len(data)),data)  #수직막대차트
plt.show()                  #그래프 보이기
```



모 듈

- 모듈 개념
- 유용한 모듈
  - time
  - random
- 패키지 개념
  - pip install 패키지명
  - matplotlib 패키지