모두를 위한 파이썬 프로그래밍

10주 모듈 활용

Module 1

모듈(Module)

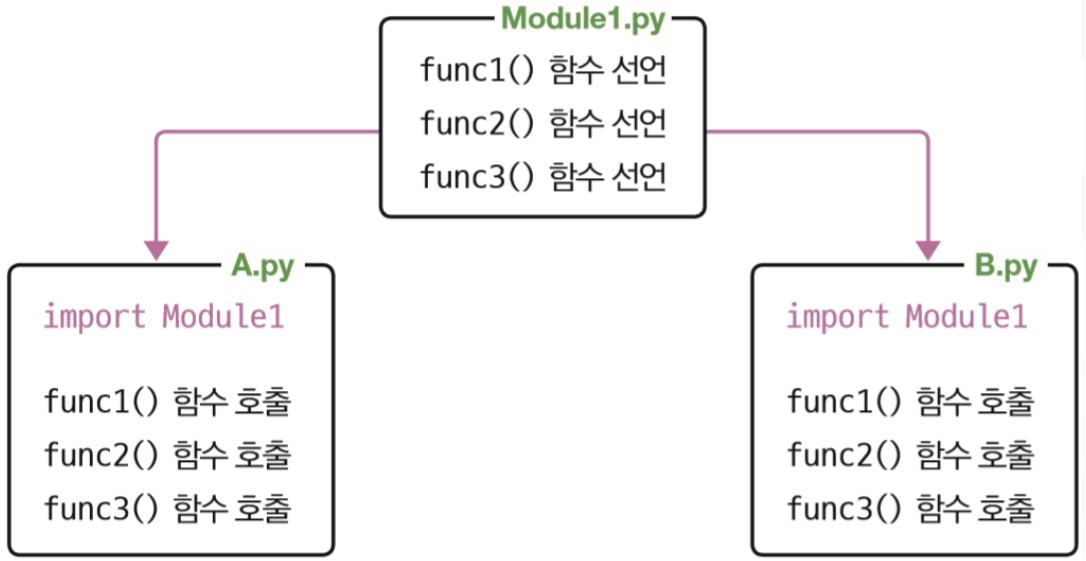
■ 모듈

✓ 코드를 작성할 때 이미 만들어져 있는 함수들을 활용하면 보다 효율적이고 빠르게 개발할 수 있음 → 프로그램 만드는 데 드는 시간과 노력 절감

✓이미 만들어져 있고 안정성이 검증된 함수들을 성격에 맞게 하나의 파이썬 파일에 묶어 만들 어 놓은 것을 모듈이라고 함

✓ import

- 외부 모듈에 있는 함수들을 활용하기 위해 가져오는 것
- 불러온 모듈 뒤에 마침표 붙여 해당 모듈이 포함한 함수 사용



모듈(Module)

Three kinds of modules

■ 모듈의 종류 3가지

- 표준 모듈 : 파이썬에서 제공하는 모듈
- 서드 파티(3rd Party) 모듈 : 파이썬이 아닌 외부 회사나 단체에서 제공하는 모듈
  - 파이썬 표준 모듈이 모든 기능을 제공하지 못함
  - 서드 파티 모듈 덕분에 파이썬에서도 다양한 분야의 고급 프로그래밍 가능
  - pyGame: 게임 제작

    Pyinstaller: 파이썬 스크립트를 독립으로 실행할 수 있는 실행파일로 생성

    Scrapy: 화면 스크랩과 웹크롤링

    KNLPY: 자연어 처리, 한국어 형태소 분석

    Beautiful Soup: a Python package for parsing HTML and XML documents.
- 사용자 정의 모듈 : 우리가 직접 만들어서 사용하는 모듈

모듈(Module)

How to import modules

■ 모듈 임포트 하는 방법

- 외부의 모듈을 가져와 사용한다고 명시해 주어야 함
- 임포트를 하지 않고 작성하면 interpreter 오류가 발생하게 됨
- 모듈 임포트 예

```
import math                 # 수학 표준 모듈 전체 import
import time                 # 시간 표준 모듈 전체 import
import myCal                # 사용자가 만든 모듈 전체 import
import mypackage.myprint    # 사용자가 만든 package 모듈 import
```

- 모듈에 들어 있는 원하는 함수만 임포트 하는 예

```
from math import sqrt, sin  # 수학 모듈에서 2개 함수만 import해서 사용
from time import sleep      # 시간 모듈중에서 sleep 함수만 import
```

- 이름이 긴 모듈은 간단한 별명으로 지정하고 사용할 수 있음

```
import turtle as t  # turtle 모듈을 t 이름으로 줄여서사용
import math as m    # math 모듈을 m 으로 줄여서 사용
```

모듈(Module)

Python Standard Modules

■ 파이썬에서 제공하는 표준 모듈들

```
1    import sys
2    print(sys.builtin_module_names)
```

```
('_abc', '_ast', '_bisect', '_blake2', '_codecs', '_codecs_cn', '_codecs_hk', '_codecs_iso2022',
'_codecs_jp', '_codecs_kr', '_codecs_tw', '_collections', '_contextvars', '_csv', '_datetime',
'_functools', '_heapq', '_imp', '_io', '_json', '_locale', '_lsprof', '_md5', '_multibytecodec',
'_opcode', '_operator', '_pickle', '_random', '_sha1', '_sha256', '_sha3', '_sha512', '_signal',
'_sre', '_stat', '_string', '_struct', '_symtable', '_thread', '_tracemalloc', '_warnings',
'_weakref', '_winapi', 'array', 'atexit', 'audioop', 'binascii', 'builtins', 'cmath', 'errno',
'faulthandler', 'gc', 'itertools', 'marshal', 'math', 'mmap', 'msvcrt', 'nt', 'parser', 'sys',
'time', 'winreg', 'xxsubtype', 'zipimport', 'zlib')
```

표준 Module 활용

2

파이썬 내장 함수

Python built-in functions

■ 내장 함수들

```
print(dir(__builtins__))
```

['ArithmeticError', 'AssertionError', 'AttributeError', 'BaseException', 'BlockingIOError', 'BrokenPipeError', 'BufferError', 'BytesWarning', 'ChildProcessError', 'ConnectionAbortedError', 'ConnectionError', 'ConnectionRefusedError', 'ConnectionResetError', 'DeprecationWarning', 'EOFError', 'Ellipsis', 'EnvironmentError', 'Exception', 'False', 'FileExistsError', 'FileNotFoundError', 'FloatingPointError', 'FutureWarning', 'GeneratorExit', 'IOError', 'ImportError', 'ImportWarning', 'IndentationError', 'IndexError', 'InterruptedError', 'IsADirectoryError', 'KeyError', 'KeyboardInterrupt', 'LookupError', 'MemoryError', 'ModuleNotFoundError', 'NameError', 'None', 'NotADirectoryError', 'NotImplemented', 'NotImplementedError', 'OSError', 'OverflowError', 'PendingDeprecationWarning', 'PermissionError', 'ProcessLookupError', 'RecursionError', 'ReferenceError', 'ResourceWarning', 'RuntimeError', 'RuntimeWarning', 'StopAsyncIteration', 'StopIteration', 'SyntaxError', 'SyntaxWarning', 'SystemError', 'SystemExit', 'TabError', 'TimeoutError', 'True', 'TypeError', 'UnboundLocalError', 'UnicodeDecodeError', 'UnicodeEncodeError', 'UnicodeError', 'UnicodeTranslateError', 'UnicodeWarning', 'UserWarning', 'ValueError', 'Warning', 'WindowsError', 'ZeroDivisionError', '__build_class__', '__debug__', '__doc__', '__import__', '__loader__', '__name__', '__package__', '__spec__', 'abs', 'all', 'any', 'ascii', 'bin', 'bool', 'breakpoint', 'bytearray', 'bytes', 'callable', 'chr', 'classmethod', 'compile', 'complex', 'copyright', 'credits', 'delattr', 'dict', 'dir', 'divmod', 'enumerate', 'eval', 'exec', 'exit', 'filter', 'float', 'format', 'frozenset', 'getattr', 'globals', 'hasattr', 'hash', 'help', 'hex', 'id', 'input', 'int', 'isinstance', 'issubclass', 'iter', 'len', 'license', 'list', 'locals', 'map', 'max', 'memoryview', 'min', 'next', 'object', 'oct', 'open', 'ord', 'pow', 'print', 'property', 'quit', 'range', 'repr', 'reversed', 'round', 'set', 'setattr', 'slice', 'sorted', 'staticmethod', 'str', 'sum', 'super', 'tuple', 'type', 'vars', 'zip']

Math (수학) Module

- math 모듈이 제공하는 함수의 목록

```
1  import math
2  print(dir(math))
```

['__doc__', '__loader__', '__name__', '__package__', '__spec__', 'acos', 'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil', 'copysign', 'cos', 'cosh', 'degrees', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs', 'factorial', 'floor', 'fmod', 'frexp', 'fsum', 'gamma', 'gcd', 'hypot', 'inf', 'isclose', 'isfinite', 'isinf', 'isnan', 'ldexp', 'lgamma', 'log', 'log10', 'log1p', 'log2', 'modf', 'nan', 'pi', 'pow', 'radians', 'remainder', 'sin', 'sinh', 'sqrt', 'tan', 'tanh', 'tau', 'trunc']

Math (수학) Module

math 모듈

함수	설명
sqrt(x)	x의 제곱근을 구한다. 세제곱근은 1/3승을 계산하여 구한다.
pow(x, y)	x의 y승을 계산한다. ** 연산자와 기능은 같지만 인수를 모두 실수로 바꾼 후 연산한다는 차이가 있다.
hypot(x, y)	피타고라스의 정리에 의거 x제곱 + y제곱의 제곱근을 구한다.
factorial(x)	x의 계승을 구한다. 인수 x는 양의 정수만 가능하다.
sin(x), cos(x), tan(x)	삼각함수를 계산한다. 인수 x는 라디안 값이다.
asin(x), acos(x), atan(x), atan2(y,x)	역삼각함수를 계산한다. 인수 x는 라디안 값이다.
sinh(x), cosh(x), tanh(x)	쌍곡선 삼각함수를 계산한다. 인수 x는 라디안 값이다.
asinh(x), acosh(x), atanh(x)	쌍곡선 역삼각함수를 계산한다. 인수 x는 라디안 값이다.
degrees(x)	라디안 값을 각도로 바꾼다.
radians(x)	각도를 라디안 값으로 바꾼다.
ceil(x)	수직선 오른쪽의 올림 값을 찾는다.
floor(x)	수직선 왼쪽의 내림 값을 찾는다.
fabs(x)	x의 절대값을 구한다.
trunc(x)	x의 소수점 이하를 버린다.
log(x, base)	base에 대한 x의 로그를 구한다. base가 생략되면 자연 로그를 구한다.
log10(x)	10의 로그를 구한다. log(x, 10)과 같다.
gcd(a, b)	a, b의 최대공약수를 구한다.

Math (수학) Module

import - 수학모듈 전체 사용하기

- ✓파이썬에는 자주 사용하는 수학함수는 표준 모듈로 함께 설치되어 있음
- ✓ math 모듈에 작성된 모든 상수와 함수를 가져와서 사용할 수 있음

```
1  import math
2  print(math.sqrt(2))
3  print(math.pi)
4  print(math.factorial(4))
```

```
1.4142135623730951
3.141592653589793
24
```

```
1  import math
2  print(math.)
```

```
cos(x)                math
cosh(x)               math
degrees(x)            math
e                     math
erf(x)                math
erfc(x)               math
exp(x)                math
expm1(x)              math
fabs(x)               math
factorial(x)          math
floor(x)              math
Press Ctrl+. to choose the selected (or first) suggestion and insert a dot afterwards  Next Tip
```

Math (수학) Module

Use only the functions you need

■ import - 필요한 함수만 사용하기

from [module name] import [function name] .. [function name]

- ✓math 모듈에 작성된 일부 함수를 가져와서 사용할 수도 있음
- ✓이 경우 가져온 함수외에는 사용할 수 없음
- ✓ 일부함수만 사용한다고 명시한 경우 가져온 함수명으로 짧게 작성됨

```
1    from math import sqrt, pi
2    print(sqrt(2))
3    print(pi)
```

```
1.4142135623730951
3.141592653589793
```

time (시간) Module

Date and time functions

■ 날짜와 시간 관련 기능 제공

```
1  import time
2  print(time.time())   # 유닉스 시간
3  print(time.ctime())
```

1572511912.074937
Thu Oct 31 17:51:52 2019

유닉스 시간(**Unix** time) 표시 방식:
1970년 1월 1일 00:00:00 협정 세계시(UTC) 부터의 경과 **시간**을 초로 환산하여 나타낸 것

✓프로그램 Running time을 측정하기(컴퓨터 속도와 관련)

```
1  import time
2  from math import factorial
3  start = time.time()
4  print(factorial(10000))
5  end = time.time()
6  print(end - start)
```

0.07878804206848145

time (시간) Module

Finding Date and Time Information

■ 오늘의 날짜와 시간 정보 추출하기

오늘은 2020년 11월 2일 입니다.

```
import time
print(time.time())  # 유닉스 시간
print(time.ctime()) # Fri Nov  1 11:38:55 2019
print(time.localtime()) # time.struct_time(tm_year=2019, tm_mon=11, tm_mday=1,
tm_hour=11, tm_min=38, tm_sec=55, tm_wday=4, tm_yday=305, tm_isdst=0)
print('오늘은 %d월 %d일 입니다' % (time.localtime().tm_mon, time.localtime().tm_mday))
# print('오늘은 %d월 %d일 입니다' % (time.localtime()[1], time.localtime()[2]))
```

```
1572575935.2434707
Fri Nov  1 11:38:55 2019
time.struct_time(tm_year=2019, tm_mon=11, tm_mday=1, tm_hour=11, tm_min=38, tm_sec=55,
tm_wday=4, tm_yday=305, tm_isdst=0)
오늘은 11월 38일 입니다
```

datetime Module

Calculate program running time

■ 실행 시간 계산하기

```
import time     # 요일 월 일 시:분:초 연도 형태
import datetime # 년-월-일 시:분:초 형태
import math
print(time.ctime()) # Fri Nov  1 12:18:15 2019
print(datetime.datetime.now()) # 2019-11-01 12:18:15.372375
start_time = time.time()
start_datetime = datetime.datetime.now()
math.factorial(1000000)
end_time = time.time()
end_datetime = datetime.datetime.now()
elapsed_time = end_time - start_time
elapsed_datetime = end_datetime - start_datetime
print('Calculate time :', elapsed_time)
print('Calculate time :', elapsed_datetime)
```

```
Mon Nov  2 19:02:40 2020
2020-11-02 19:02:40.444203
Calculate time : 6.792816638946533
Calculate time : 0:00:06.792816
```

calendar Module

Provide calendar function

■ 달력 기능 제공

✓ 년도별, 월별 출력 가능

```
1  import calendar
2  print(calendar.calendar(2019))
3  print(calendar.month(2020, 1))
```

```
                                  2019

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

random module

Extract random numbers that can't predict which one will come out

■ 난수 생성 모듈

✓ 어떤 수가 나올지 예측할 수 없는 무작위 수를 추출

- random.random()
  - 0 ~ 1사이의 랜덤값 추출
- random.randint(시작_값, 끝_값)
  - 시작~끝값 범위에서 정수 하나를 임의로 선택
- random.choice(리스트)
  - 리스트의 값 중 하나를 임의로 선택
- random.sample(리스트, 뽑을_개수)
  - 리스트의 값 중에서 지정한 개수만큼 중복 없이 임의로 선택
- random.shuffle(리스트)→ 파괴적 함수
  - 리스트의 배열 순서를 무작위로 섞음

```
1  import random
2  for i in range(100):
3      print(random.)
```

```
random
randrange
randint
choice
shuffle
choices
```

random module

Extract random values between 0 and 1 (greater than 0 and less than 1)

- random.random()
  - 0 ~ 1사이의 랜덤값 추출(0보다 크고 1보다 작은수)

```
1    import random
2    for i in range(10):
3        print(random.random())
```

```
0.9430078586041801
0.5981054070885321
0.20364861412719149
0.09682533388277637
0.0932799288716829
0.4648300119954124
0.6490506178036367
0.777335753813548
0.39056673436735745
0.23615434979106864
```

random module

Generate random numbers from start and end ranges

- random.randint(start value, end value)
  - 시작값과 끝값을 포함한 범위에서 난수 생성

```
1    import random
2    for i in range(10):
3        print(random.randint(1, 10))
```

```
7
4
7
7
9
1
9
4
5
10
```

random module

Create Random Arithmetic Quiz

- 무작위 산수 문제 풀기

```
import random

x = random.randint(1,9)
y = random.randint(1,9)
hap = [0,x,y]
hap[0] = sum(hap)
answer = int(input('%d + %d = ' % (x, y)))
if answer == hap[0]:
    print('정답입니다. \n')
else:
    print('아쉽군요 \n')
```

```
7 + 4 = 11
정답입니다
```

- 자동(난이도) 문제 풀기 응용

```
문제 : 1
1 + 1 = 2
정답입니다.

문제 : 2
1 + 1 = 2
정답입니다.

문제 : 3
1 + 3 = 4
정답입니다.

문제 : 4
3 + 2 = 5
정답입니다.

문제 : 19
13 + 7 = 20
정답입니다.

문제 : 20
2 + 20 =
```

random module

Select one from the list

- random.choice(list)
  - 리스트에서 하나를 선택

```
1  import random
2  menu = ['피자', '떡볶이', '학식', '라면', '햄버거', '초밥', '짜장면']
3  print('오늘은', random.choice(menu), '강추 합니다')
4  print(random.sample(menu, 2))    # 2개 메뉴를 선택
```

오늘은 학식 강추 합니다
['짜장면', ' 떡뽁이']

random module

Reorder arrays in list

- random.shuffle(list)
  - 리스트에서 배열의 순서를 바꿈
  - 파괴적 함수 : 함수가 수행된 후 리스트에 값을 되돌릴 수 없음

```
import random
list_a = [1, 2, 4, 5, 6]
random.shuffle(list_a)
print(list_a)
```

[6, 5, 4, 1, 2]

```
import random
list_a = ['kor', 'eng', 'mat']
random.shuffle(list_a)
print(list_a)
```

['mat', 'kor', 'eng']

random module

Reorder arrays in list

- random.choices(list, weights=[,,], k = n)
  - weights: 배열에서 선택될 확률을 요소별로 지정, 상대 확률
  - k = n: 선택할 n개의 수 지정

```
import random

input('추첨을 통해 경품을 드립니다. ₩n준비가 되면 엔터키를 누르세요')
product = ['여행용 휴지', '휴대폰 고리', '스타벅스 쿠폰', '문화상품권 5만원', '아이패드' ]
lucky_box = random.choices(product, weights=[50, 30, 5, 2, 1], k=1)
print('축하합니다. %s에 당첨되었습니다.' % lucky_box[0])
```

sys module

Python interpreter version and execution platform (OS) query

■ 파이썬 인터프리터 버전과 실행 플랫폼(운영체제)기능 조회

```
import sys
print('Python version :', sys.version)
print('Platform :', sys.platform)
```

```
Python version : 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)]
Platform : win32
```

sys module

Passing arguments when running Python

■ 명령행 인수

✓ 파이썬 실행 파일 뒤에 인수를 전달할 수 있음
- 크롬프로그램에서 네이버 주소를 인수로 전달하여 실행한 예

```
C:\Program Files (x86)\Google\Chrome\Application>chrome www.naver.com
```

✓ sys.argv 읽어 명령행 인수의 값 읽을 수 있음

```
# 두 수를 명령행 인수로 입력 받아 합계를 출력해 주는 프로그램
import sys
a = int(sys.argv[1])
b = int(sys.argv[2])
print(a + b)
```

```
C:\python\ch10>exam_sys.py 322 123
445
```

Package

A collection of modules organized into folders

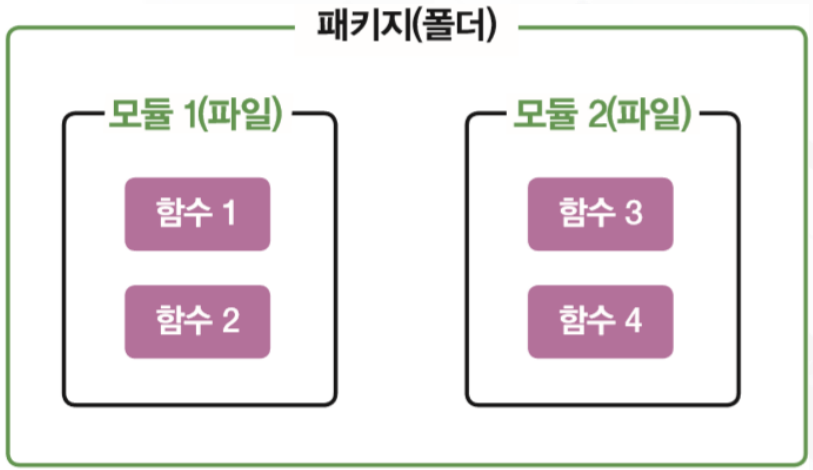
■ 패키지

- 모듈이 하나의 *.py 파일 안에 함수가 여러 개 들어 있는 것이라면, 패키지(Package)는 여러 모듈을 모아 폴더의 형태로 구성한 것
- 모듈을 주제별로 분리할 때 주로 사용
- 임포트 형식

```
import 패키지명.모듈명

또는

from 패키지명.모듈명 import 함수명
```



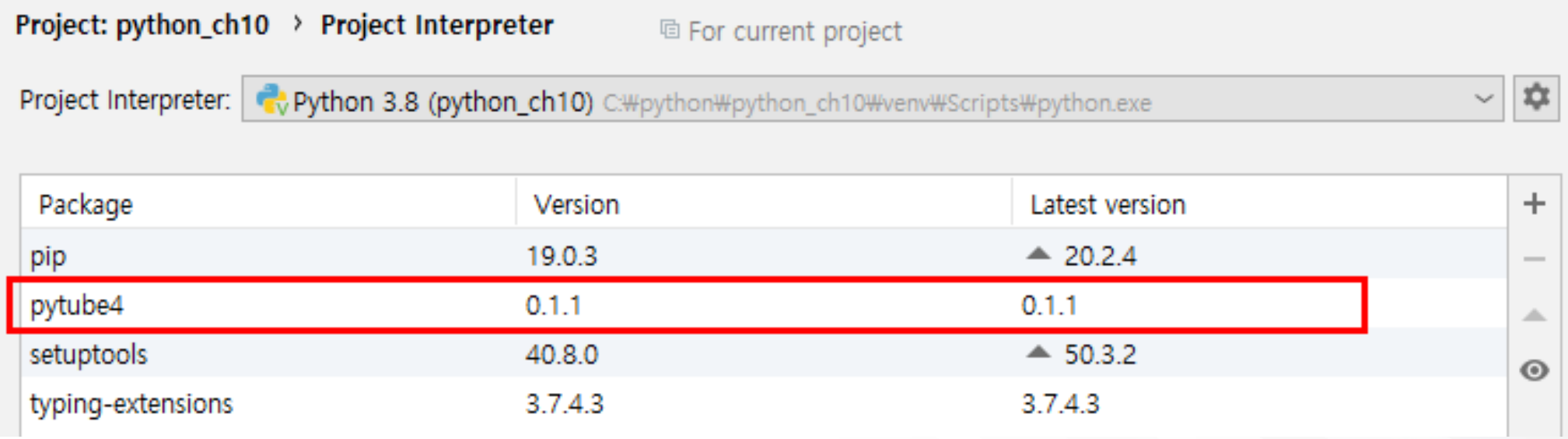
모듈 활용 예
(YouTube 동영상 다운받기)

3

비디오 데이터 수집하기

■ 유튜브 영상 주소로 광고없는 동영상을 직접 다운받아 저장하는 프로그램을 작성해 보자

✓ pytube4 Package 설치



모듈활용 실습문제

✓ 모듈을 활용한 코딩

```
from pytube import YouTube

YouTubeVideo = YouTube('https://www.youtube.com/watch?v=zeLiGf3WIP0')
myVideo = YouTubeVideo.streams.get_highest_resolution()
myVideo.download()
```

모듈활용 실습문제

- ✓ 수정 사항
  - ✓ 고려 사항1: 어떤 동영상을 다운받고 있는 지 알 수 없음 → 다운되는 동영상 제목이 출력 되도록
  - ✓ 고려 사항2: 다운로드 중, 다운로드 완료 등의 진행사항을 추가
  - ✓ 고려 사항3: 소스코드에 매번 다운받을 주소를 매번 입력하고 매번 실행해 주어야 하는 번거로움 해결
  - ✓ 고려 사항4: 파이썬, 파이참이 없어도 누구나 실행해서 사용할 수 있도록 독립 프로그램으로 만들기

```
다운로드할 동영상의 주소를 입력해 주세요 > https://www.youtube.com/watch?v=zeLiGf3WIP0
<동영상 제목> :  윈도우에 이런 기능이? 혼자 알기 아까운 꿀팁 대방출합니다.
>> 다운로드 중....
☆ 다운로드가 완료되었습니다.^^

다운로드할 동영상의 주소를 입력해 주세요 >
```

파이썬 프로그램없이 실행가능한 파일 만들기

파이썬, 파이참이 없어도 누구나 실행해서 사용할 수 있도록 독립 프로그램으로 만들기

pyinstaller 모듈을 활용하여 실행파일(exe) 생성

- ✓ pyinstaller 패키지 설치
- ✓ 파이썬 명령창에서 실행

```
pyinstaller --onefile xx.py
→ 하나의 실행파일로 생성
```

```
pyinstaller --noconsol xx.py
→ GUI로 구현되어 콘솔창이 필요없는 경우
```

```
pyinstaller --onefile --noconsol xx.py
→ GUI로 구현되어 콘솔창을 보이지 않게 처리하고, 하나의 실행파일로 압축해서 생성
```

Thank you