

Ch14. 모듈과 패키지

- **모듈 (module)**

- 일반적으로는 “독자적인 기능을 갖는 구성 요소”를 의미
- 파이썬에서는 개별 소스 파일을 일컫는 말

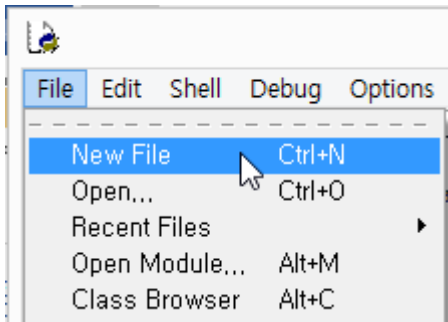
- **제공자에 따른 분류**

- 표준 모듈 : 파이썬과 함께 따라오는 모듈
- 사용자 생성 모듈 : 프로그래머가 직접 작성한 모듈
- 서드 파티(3rd Party) 모듈 : 파이썬 재단도 프로그래머도 아닌 다른 프로그래머, 또는 업체에서 제공한 모듈

- **패키지(package)**

- 여러 모듈을 묶은 것임
- 파이썬을 설치할 때 다양한 모듈과 패키지가 기본으로 설치됨
- 기본 모듈과 패키지로 부족하다면 다른 사람이 만든 유명 모듈과 패키지를 설치해서 쓸 수도 있음

- IDLE을 실행한 후 [File]=>[New File] 메뉴항목을 선택하여 편집창 실행



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

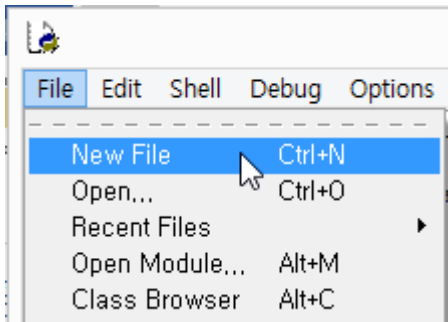
```
def minus(a,b):  
    return a-b
```

```
def mul(a,b):  
    return a*b
```

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

IDLE 편집창에서 [File]→[Save] 메뉴 항목을 선택하고, 디렉토리를 하나 골라 그곳에 "calculator.py"라는 이름으로 모듈을 저장

- IDLE을 실행한 후 [File]=>[New File] 메뉴항목을 선택하여 편집창 실행



IDLE 편집창에서 [File]→[Save] 메뉴 항목을 선택하고, 디렉토리를 하나 골라 그곳에 "test_cal.py"라는 이름으로 모듈을 저장

```
import calculator
```

```
print(calculator.plus(10,5))  
print(calculator.minus(10,5))  
print(calculator.mul(10,5))  
print(calculator.div(10,5))
```

```
=== RESTART: I
```

```
===
```

```
15  
5  
50  
2.0
```

모듈 분리의 장점
코드의 재사용

- **import 의 역할**

- “다른 모듈 내의 코드에 대한 접근”을 가능하게 하는 것
- “다른 코드”에는 변수, 함수, 클래스 등이 모두 포함

- import문을 사용하는 첫 번째 방법

```
import 모듈 #모듈의 실제 파일 명은 “모듈.py”
```

- import문을 사용하는 두 번째 방법

```
from 모듈 import 변수 또는 함수
```

from 모듈	from 모듈 import 변수 또는 함수
<pre>import calculator print(calculator.plus(10, 5)) print(calculator.minus(10, 5)) print(calculator.multiply(10, 5)) print(calculator.divide(10, 5))</pre>	<pre>from calculator import plus from calculator import minus from calculator import multiply from calculator import divide print(plus(10, 5)) print(minus(10, 5)) print(multiply(10, 5)) print(divide(10, 5))</pre>

- “from 모듈 import 변수 또는 함수”의 세 가지 버전

1. 사용할 변수나 함수의 이름을 일일이 명기

```
from calculator import plus
from calculator import minus
print(plus(10,5))
print(minus(10,5))
#print(mul(10,5))
#print(div(10,5))
```

calculator 모듈의 plus라는 함수를 불러들였으므로 “calculator.” 없이 plus() 이름만으로 함수를 호출할 수 있습니다.

multiply()와 divide()는 import하지 않았습니다. 현재 모듈에서는 보이지 않는 함수입니다.

2. 콤마(,)를 이용해서 여러 함수(또는 변수)의 이름을 한 줄에 기입

```
from calculator import plus, minus

print(plus(10,5))
print(minus(10,5))
#print(mul(10,5))
#print(div(10,5))
```

from calculator import plus
from calculator import minus
와 동일한 코드입니다.

- “from 모듈 import 변수 또는 함수”의 세 가지 버전

- 3. 와일드카드 * 사용

```
from calculator import *  
  
print(plus(10,5))  
print(minus(10,5))  
print(mul(10,5))  
print(div(10,5))
```

- 4. import 모듈 as 새이름

```
import calculator as c  
  
print(c.plus(10,5))  
print(c.minus(10,5))  
print(c.mul(10,5))  
print(c.div(10,5))
```

calculator 모듈을 c라는 이름으로 불러옵니다.

calculator라는 이름 대신 c를 이용하여 함수 이름에 접근합니다.

- import로 모듈 가져오기

- 모듈은 import 키워드로 가져올 수 있음
(모듈을 여러 개 가져올 때는 모듈을 콤마로 구분)

```
import 모듈
  • import 모듈1, 모듈2
  • 모듈.변수
  • 모듈.함수()
  • 모듈.클래스()
```

- 파이썬 표준 라이브러리의 수학 모듈 math를 가져와서 원주율을 출력

```
>>> import math
>>> math.pi
3.141592653589793
```

```
>>> import math
>>> math.sqrt(4.0)
2.0
>>> math.sqrt(2.0)
1.4142135623730951
```


- **import as로 모듈 이름 지정하기**

- 모듈의 함수를 사용할 때 `math.sqrt`처럼 일일이 `math`를 입력하고 싶지 않을 때는 `import as`를 사용하여 모듈의 이름을 지정할 수 있음

- `import 모듈 as 이름`

- `math` 모듈을 `m`으로 줄여보자

```
>>> import math as m      # math 모듈을 가져오면서 이름을 m으로 지정
>>> m.sqrt(4.0)           # m으로 제곱근 함수 사용
2.0
>>> m.sqrt(2.0)           # m으로 제곱근 함수 사용
1.4142135623730951
```

- from import로 모듈의 일부만 가져오기

- from 모듈 import 변수

```
>>> from math import pi    # math 모듈에서 변수 pi만 가져옴
>>> pi                    # pi를 바로 사용하여 원주율 출력
3.141592653589793
```

- from 모듈 import 함수

- from 모듈 import 클래스

```
>>> from math import sqrt  # math 모듈에서 sqrt 함수만 가져옴
>>> sqrt(4.0)              # sqrt 함수를 바로 사용
2.0
>>> sqrt(2.0)              # sqrt 함수를 바로 사용
1.4142135623730951
```

- **from import로 모듈의 일부만 가져오기**

- math 모듈에서 가져올 변수와 함수가 여러 개일때는
import 뒤에 가져올 변수, 함수, 클래스를 콤마로 구분하여
여러 개를 지정해주면 됨

- **from 모듈 import 변수, 함수, 클래스**

```
>>> from math import pi, sqrt    # math 모듈에서 pi, sqrt를 가져옴
>>> pi                          # pi로 원주율 출력
3.141592653589793
>>> sqrt(4.0)                   # sqrt 함수 사용
2.0
>>> sqrt(2.0)                   # sqrt 함수 사용
1.4142135623730951
```

- **from 모듈 import ***

```
>>> from math import *          # math 모듈의 모든 변수, 함수, 클래스를 가져옴
>>> pi                          # pi로 원주율 출력
3.141592653589793
>>> sqrt(4.0)                   # sqrt 함수 사용
2.0
>>> sqrt(2.0)                   # sqrt 함수 사용
1.4142135623730951
```

- **from import로 모듈의 일부를 가져온 뒤 이름 지정하기**

- `from 모듈 import 변수 as 이름`
- `from 모듈 import 함수 as 이름`
- `from 모듈 import 클래스 as 이름`

```
>>> from math import sqrt as s      # math 모듈에서 sqrt 함수를 가져오면서 이름을 s로 지정
>>> s(4.0)                          # s로 sqrt 함수 사용
2.0
>>> s(2.0)                          # s로 sqrt 함수 사용
1.4142135623730951
```

- 여러 개를 가져왔을 때 각각 이름을 지정할 때는 각 변수, 함수, 클래스 등을 콤마로 구분하여 as를 여러 개 지정하면 됨

- from 모듈 import 변수 as 이름1, 함수 as 이름2, 클래스 as 이름3

```
>>> from math import pi as p, sqrt as s
>>> p           # p로 원주를 출력
3.141592653589793
>>> s(4.0)      # s로 sqrt 함수 사용
2.0
>>> s(2.0)      # s로 sqrt 함수 사용
1.4142135623730951
```

- import로 패키지 가져오기

- 패키지는 특정 기능과 관련된 여러 모듈을 묶은 것인데, 패키지에 들어있는 모듈도 import를 사용하여 가져옴

- import 패키지.모듈
- import 패키지.모듈1, 패키지.모듈2
- 패키지.모듈.변수
- 패키지.모듈.함수()
- 패키지.모듈.클래스()

- 여기서는 파이썬 표준 라이브러리에서 urllib 패키지의 request 모듈을 가져와보자 (urllib은 URL 처리에 관련된 모듈을 모아 놓은 패키지임)

```
>>> import urllib.request
>>> response = urllib.request.urlopen('http://www.google.co.kr')
>>> response.status
200
```

- import urllib.request와 같이 패키지.모듈 형식으로 가져옴
- 모듈의 함수를 사용할 때도 urllib.request.urlopen()과 같이 패키지.모듈.함수() 형식으로 패키지 이름과 모듈 이름을 모두 입력해줌

- import as로 패키지 모듈 이름 지정하기

- import 패키지.모듈 as 이름

```
>>> import urllib.request as r    # urllib 패키지의 request 모듈을 가져오면서 이름을 r로 지정
>>> response = r.urlopen('http://www.google.co.kr')    # r로 urlopen 함수 사용
>>> response.status
200
```

cf) urlopen 함수

- urllib.request.urlopen
- URL을 여는 함수
- URL 열기에 성공 하면 response.status 값이 200을 반환
- 200 : HTTP 상태 코드, 웹서버가 요청을 제대로 처리 했다는 뜻

- from import로 패키지의 모듈에서 일부만 가져오기

- from 패키지.모듈 import 변수
- from 패키지.모듈 import 함수
- from 패키지.모듈 import 클래스
- from 패키지.모듈 import 변수, 함수, 클래스

```
>>> from urllib.request import Request, urlopen    # urlopen 함수, Request 클래스를 가져옴
>>> req = Request('http://www.google.co.kr')      # Request 클래스를 사용하여 req 생성
>>> response = urlopen(req)                        # urlopen 함수 사용
>>> response.status
200
```

- urlopen 함수에 URL을 바로 넣어도 되고,
Request('http://www.google.co.kr')와 같이 Request 클래스에 URL을 넣은 뒤
req를 생성해서 urlopen 함수에 넣어도 됨

- from import로 패키지의 모듈에서 모든 변수, 함수, 클래스를 가져오는 방법

- from 패키지.모듈 import *

```
>>> from urllib.request import *      # urllib의 request 모듈에서 모든 변수, 함수, 클래스를 가져옴
>>> req = Request('http://www.google.co.kr')  # Request를 사용하여 req 생성
>>> response = urlopen(req)              # urlopen 함수 사용
>>> response.status
200
```

- from import로 패키지의 모듈의 일부를 가져온 뒤 이름 지정하기

- from 패키지.모듈 import 변수 as 이름

- from 패키지.모듈 import 변수 as 이름, 함수 as 이름, 클래스 as 이름

```
>>> from urllib.request import Request as r, urlopen as u
>>> req = r('http://www.google.co.kr')      # r로 Request 클래스 사용
>>> response = u(req)                        # u로 urlopen 함수 사용
>>> response.status
200
```


- **파이썬 패키지 인덱스에서 패키지 설치하기**

- 파이썬은 파이썬 표준 라이브러리(Python Standard Library, PSL) 이외에도 파이썬 패키지 인덱스(Python Package Index, PyPI)를 통해 다양한 패키지를 사용할 수 있음
- 명령만 입력하면 원하는 패키지를 인터넷에서 다운로드하여 설치해줄 뿐만 아니라 관련된 패키지(의존성)까지 자동으로 설치해주므로 매우 편리함

- **pip 설치하기**

- pip는 파이썬 패키지 인덱스의 패키지 관리 명령어이며

- Windows용 파이썬에는 기본으로 내장되어 있음**

- 리눅스와 macOS에서는 콘솔(터미널)에서 다음과 같은 방법으로 설치하면 됨







리눅스, macOS

```
$ curl -O https://bootstrap.pypa.io/get-pip.py
$ sudo python3 get-pip.py
```

- pip로 패키지 설치하기

- pip install 패키지

- Windows에서는 명령 프롬프트를 실행(윈도우 키+R을 누른 뒤 cmd를 입력)하고, 리눅스와 macOS에서 콘솔(터미널)을 실행한 뒤 pip install requests 명령을 입력함 (pip 명령은 파이썬 셸 >>>에 입력하면 안 됨. 반드시 명령 프롬프트, 콘솔, 터미널에 입력)
 - 참고로 requests는 파이썬 표준 라이브러리의 urllib.request와 비슷한 역할을 하는 패키지인데 좀 더 기능이 많고 편리함

내 PC > 로컬 디스크 (C:) > 사용자 > raipiel > AppData > Local > Programs > Python > Python37-32 > Scripts				
이름	수정한 날짜	유형	크기	
 chardetect	2019-11-18 오후 3:52	응용 프로그램	91KB	
 easy_install	2019-09-06 오전 8:56	응용 프로그램	91KB	
 easy_install-3.7	2019-09-06 오전 8:56	응용 프로그램	91KB	
 pip	2019-11-18 오후 3:51	응용 프로그램	91KB	
 pip3.7	2019-11-18 오후 3:51	응용 프로그램	91KB	
 pip3	2019-11-18 오후 3:51	응용 프로그램	91KB	

- pip로 패키지 설치하기

- pip install 패키지

```
C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip
Usage:
  C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32\python.exe -m pip <command> [options]

Commands:
  install           Install packages.
  download          Download packages.
  uninstall         Uninstall packages.
  freeze            Output installed packages in requirements format.
  list              List installed packages.
  show              Show information about installed packages.
  check             Verify installed packages have compatible dependencies.
  config            Manage local and global configuration.
  search            Search PyPI for packages.
  wheel             Build wheels from your requirements.
  hash              Compute hashes of package archives.
  completion        A helper command used for command completion.
  help              Show help for commands.
```

- cd 폴더 이름으로 이동 (Change Directory)

- pip로 패키지 설치하기

- pip install 패키지

```
C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip install --upgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/00/b6/9cfa56b4081ad13874b0c6f96af8ce16cfbc1cb06bedf8e9164ce5551ec/pip-19.3.1-py2.py3-none-any.whl (1.4MB)
    100% |#####| 1.4MB 4.1MB/s
Installing collected packages: pip
  Found existing installation: pip 19.0.3
    Uninstalling pip-19.0.3:
      Successfully uninstalled pip-19.0.3
Successfully installed pip-19.3.1
```

- pip로 패키지 설치하기

- pip install 패키지

```
C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip install requests
Collecting requests
  Downloading https://files.pythonhosted.org/packages/51/bd/23c926cd341ea6b7dd0b2a00aba99ae0f828be89d72b2190f27c11d4b7fb/requests-2.22.0-py2.py3-none-any.whl (57kB)
    |#####| 61kB 435kB/s
Collecting urllib3!=1.25.0,!1.25.1,<1.26,>=1.21.1
  Downloading https://files.pythonhosted.org/packages/b4/40/a9837291310ee1ccc242ceb6ebfd9eb21539649f193a7c8c86ba15b98539/urllib3-1.25.7-py2.py3-none-any.whl (125kB)
    |#####| 133kB 1.3MB/s
Collecting certifi>=2017.4.17
  Downloading https://files.pythonhosted.org/packages/18/b0/8146a4f8dd402f60744fa380bc73ca47303cccf8b9190fd16a827281eac2/certifi-2019.9.11-py2.py3-none-any.whl (154kB)
    |#####| 163kB 3.3MB/s
Collecting idna<2.9,>=2.5
  Downloading https://files.pythonhosted.org/packages/14/2c/cd551d81dbe15200be1cf41cd03869a46fe7226e7450af7a6545bfc474c9/idna-2.8-py2.py3-none-any.whl (58kB)
    |#####| 61kB 3.8MB/s
Collecting chardet<3.1.0,>=3.0.2
  Downloading https://files.pythonhosted.org/packages/bc/a9/01ffebfb562e4274b6487b4bb1ddec7ca55ec7510b22e4c51f14098443b8/chardet-3.0.4-py2.py3-none-any.whl (133kB)
    |#####| 143kB 6.4MB/s
Installing collected packages: urllib3, certifi, idna, chardet, requests
  WARNING: The script chardet.exe is installed in 'C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32\Scripts' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed certifi-2019.9.11 chardet-3.0.4 idna-2.8 requests-2.22.0 urllib3-1.25.7

C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>
```

```
>>> import requests
>>> r=requests.get('http://www.google.co.kr')
>>> r.status_code
200
```

- pip로 패키지 설치하기

```
C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip list
Package      Version
-----
certifi      2019.9.11
chardet      3.0.4
idna         2.8
pip          19.3.1
requests     2.22.0
setuptools   40.8.0
urllib3      1.25.7

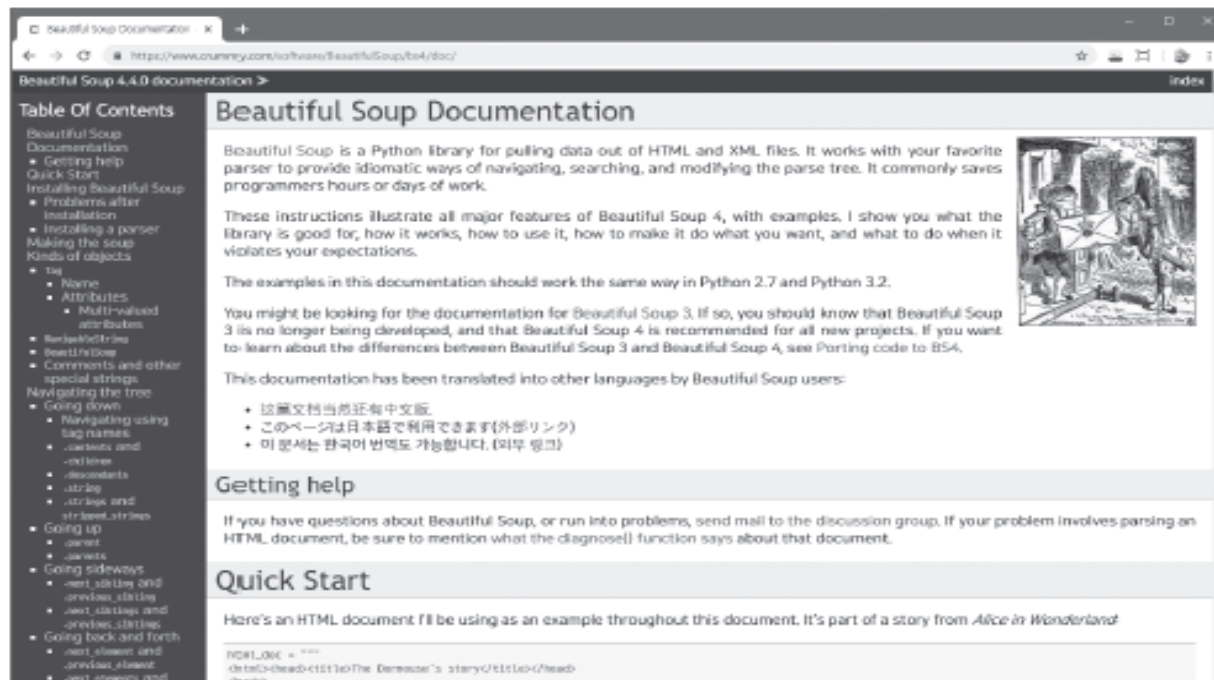
C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip show urllib3
Name: urllib3
Version: 1.25.7
Summary: HTTP library with thread-safe connection pooling, file post, and more.
Home-page: https://urllib3.readthedocs.io/
Author: Andrey Petrov
Author-email: andrey.petrov@shazow.net
License: MIT
Location: c:\Users\raipiel\AppData\Local\Programs\Python\Python37-32\lib\site-packages
Requires:
Required-by: requests

C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip show certifi
Name: certifi
Version: 2019.9.11
Summary: Python package for providing Mozilla's CA Bundle.
Home-page: https://certifi.io/
Author: Kenneth Reitz
Author-email: me@kennethreitz.com
License: MPL-2.0
Location: c:\Users\raipiel\AppData\Local\Programs\Python\Python37-32\lib\site-packages
Requires:
Required-by: requests
```

- pip로 패키지 설치하기

```
C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip show requests  
Name: requests  
Version: 2.22.0  
Summary: Python HTTP for Humans.  
Home-page: http://python-requests.org  
Author: Kenneth Reitz  
Author-email: me@kennethreitz.org  
License: Apache 2.0  
Location: c:\users\raipiel\appdata\local\programs\python\python37-32\lib\site-packages  
Requires: idna, chardet, certifi, urllib3  
Required-by:
```


- Beautiful Soup 모듈
 - 웹 페이지 분석 모듈
 - <http://www.crummy.com/software/BeautifulSoup/bs4/doc/>



- pip로 패키지 설치하기 (beautifulsoup4)

```
C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32>python -m pip install beautifulsoup4
Collecting beautifulsoup4
  Downloading https://files.pythonhosted.org/packages/3b/c8/a55eb6ea11cd7e5ac4bacdf92bac4693b90d3ba79268be16527555e186f0
/beautifulsoup4-4.8.1-py3-none-any.whl (101kB)
    | 102kB 233kB/s
Collecting soupsieve>=1.2
  Downloading https://files.pythonhosted.org/packages/81/94/03c0f04471fc245d08d0a99f7946ac228ca98da4fa75796c507f61e688c2
/soupsieve-1.9.5-py2.py3-none-any.whl
Installing collected packages: soupsieve, beautifulsoup4
Successfully installed beautifulsoup4-4.8.1 soupsieve-1.9.5
```

```
>>> import bs4
>>> bs4
<module 'bs4' from 'C:\Users\raipiel\AppData\Local\Programs\Python\Python37-32\lib\site-packages\bs4\__init__.py'>
```

- <http://www.weather.go.kr/weather/forecast/mid-term-rss3.jsp?stdId=108>

```
<rss version="2.0">
  <channel>
    <title>기상청 육상 중기예보</title>
    <link>
      http://www.kma.go.kr/weather/forecast/mid-term_01.jsp
    </link>
    <description>기상청 날씨 웹서비스</description>
    <language>ko</language>
    <generator>기상청</generator>
    <pubDate>2019년 11월 25일 (월)요일 06:00</pubDate>
  </channel>
  <item>
    <author>기상청</author>
    <category>육상중기예보</category>
    <title>전국 육상 중기예보 - 2019년 11월 25일 (월)요일 06:00 발표</title>
    <link>
      http://www.kma.go.kr/weather/forecast/mid-term_01.jsp
    </link>
    <guid>
      http://www.kma.go.kr/weather/forecast/mid-term_01.jsp
    </guid>
    <description>
      <header>
        <title>전국 육상중기예보</title>
        <tm>201911250600</tm>
      </header>
      <wf>
        <![CDATA[
          동풍의 영향으로 28일에 강원영동에 비 또는 눈이 오겠고, 기압골의 영향으로 12월 1일은 전국에 비(강원도는 비 또는 눈)가 오겠습니다. <br />그
          거나 조금 높고 후반에는 조금 낮겠습니다.<br />강수량은 평년(1~3mm)과 비슷하겠으나, 강원영동은 많겠습니다.<br /><br />* 28일 강원산지에는 많
        ]]>
      </wf>
    </description>
  </item>
  <location wl_ver="3">
    <province>서울 · 인천 · 경기도</province>
    <city>서울</city>
    <data>
      <mode>A02</mode>
      <tmEf>2019-11-28 00:00</tmEf>
      <wf>구름많음</wf>
      <tmn>1</tmn>
      <tmx>9</tmx>
      <reliability>
        <rnSt>30</rnSt>
      </data>
    <data>
      <mode>A02</mode>
      <tmEf>2019-11-28 12:00</tmEf>
      <wf>구름많음</wf>
      <tmn>1</tmn>
      <tmx>9</tmx>
      <reliability>
        <rnSt>30</rnSt>
      </data>
    </data>
  </location>
</rss>
```

```
from urllib import request
from bs4 import BeautifulSoup
```

#urlopen()함수로 기상청의 전국날씨를 읽는다.

```
target = request.urlopen("http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stndId=108")
```

#BeautifulSoup을 사용해 웹 페이지 분석

```
soup = BeautifulSoup(target, "html.parser")
```

#location 태그 찾기

```
for location in soup.select("location"):
    #내부의 city, wf, tmn, tmx 태그를 찾아 출력
    print("도시 :", location.select_one("city").string)
    print("날씨 :", location.select_one("wf").string)
    print("최저기온 :", location.select_one("tmn").string)
    print("최고기온 :", location.select_one("tmx").string)
    print()
```

#태그 여러개 선택 select(), 하나만 선택 select_one()

```
-----
도시 : 서울
날씨 : 구름 많음
최저기온 : 1
최고기온 : 9
```

```
도시 : 평택
날씨 : 구름 많음
최저기온 : -1
최고기온 : 10
```

```
도시 : 인천
날씨 : 구름 많음
최저기온 : 2
최고기온 : 8
```

```
도시 : 춘천
날씨 : 구름 많음
최저기온 : 0
최고기온 : 8
```

```
도시 : 수원
날씨 : 구름 많음
최저기온 : -1
최고기온 : 10
```

```
도시 : 원주
날씨 : 구름 많음
최저기온 : 0
최고기온 : 9
```

```
도시 : 파주
날씨 : 구름 많음
최저기온 : -2
최고기온 : 8
```

```
도시 : 강릉
날씨 : 흐리고 비/눈
최저기온 : 4
최고기온 : 7
```

```
도시 : 이천
날씨 : 구름 많음
최저기온 : -2
최고기온 : 9
```

```
도시 : 대전
날씨 : 구름 많음
최저기온 : 0
최고기온 : 11
```

- 모듈 만들기

- 2의 거듭제곱을 구하는 모듈 생성
- 프로젝트 폴더(C:\wproject) 안에 square2.py 파일로 저장

square2.py

```
base = 2          # 변수  
  
def square(n):    # 함수  
    return base ** n
```

- 모듈을 만들었을 때 모듈 이름은 square2임
- 스크립트 파일에서 확장자 .py를 제외하면 모듈 이름이 됨

- 모듈 사용하기

- 프로젝트 폴더(C:\Wproject) 안에 main.py 파일로 저장한 뒤 실행
- square2.py 파일과 main.py 파일은 반드시 같은 폴더에 있어야 함

import 모듈

- 모듈.변수
- 모듈.함수()

main.py

```
import square2          # import로 square2 모듈을 가져옴

print(square2.base)     # 모듈.변수 형식으로 모듈의 변수 사용
print(square2.square(10)) # 모듈.함수() 형식으로 모듈의 함수 사용
```

실행 결과

```
2
1024
```

- from import로 변수, 함수 가져오기
 - 모듈에서 from import로 변수와 함수를 가져온 뒤 모듈 이름을 붙이지 않고 사용할 수도 있음

• from 모듈 import 변수, 함수

```
>>> from square2 import base, square
>>> print(base)
2
>>> square(10)
1024
```