

OPEN API 개요

- Open Application Programming Interface
- 특정 웹 사이트에서 자신이 가진 기능을 이용할 수 있도록 공개한 프로그래밍 인터페이스
- 누구나 사용할 수 있도록 공개된 API로 응용 소프트웨어나 웹 서비스의 프로그래밍적인 권한을 제공
- 대한민국 정부의 공공데이터포털(www.data.go.kr)을 통해 제공되는 API 서비스, 도로명 주소 조회 서비스, 동네예보정보조회서비스
- 네이버 지도, 구글맵, 코로나 맵 등
- JSON, XML 등으로 제공됨

JSON

- JavaScript Object Notation
- 키와 값으로 구성된 자바스크립트의 데이터 형태
- 파이썬의 JSON 레퍼런스

<https://docs.python.org/ko/3/library/json.html>

- JSON 예시

```
{  
  "이름": "서태웅",  
  "성별": "남",  
  "학교": "북산 고등학교 1학년 10반 22번",  
  "특기": "농구",  
  "포지션": "스몰 포워드(SF)",  
  "별명": "여우",  
  "체격": {"키": "187 cm", "몸무게": " 75 kg", "혈액형": " AB형(Rh+)"},  
  "라이벌": ["윤대협", "정우성"]  
}
```

JSON

- JavaScript Object Notation
- 키와 값으로 구성된 자바스크립트의 데이터 형태
- 파이썬의 JSON 레퍼런스

<https://docs.python.org/ko/3/library/json.html>

- JSON 예시

```
{  
  "이름": "서태웅",  
  "성별": "남",  
  "학교": "북산 고등학교 1학년 10반 22번",  
  "특기": "농구",  
  "포지션": "스몰 포워드(SF)",  
  "별명": "여우",  
  "체격": {"키": "187 cm", "몸무게": " 75 kg", "혈액형": " AB형(Rh+)"},  
  "라이벌": ["윤대협", "정우성"]  
}
```

JSON

- json 샘플 사이트 : <http://jsonplaceholder.typicode.com/>
- url 요청 => json 데이터 => 리스트 딕셔너리 => 데이터프레임

```
1 import os
2 import requests
3 import pandas as pd
4 import json
```

```
1 res = requests.get("https://jsonplaceholder.typicode.com/todos")
2 print(res)
3 json_str = res.text
4 print(json_str)
```

JSON

```
1 todos = json.loads(json_str)
2 print(type(todos), len(todos))
3 print(todos[0])
4 print(todos[-1])
5 print(todos[0:5])
```

```
1 cnt = 0
2 for i in range(0, len(todos)):
3     if todos[i]['userId']==1:
4         print(todos[i])
5         cnt += 1
6 print(f'총 갯수 - {cnt}')
7 print('-'*20)
```

JSON

```
1 cnt = 0
2 for i in range(0, len(todos)):
3     if todos[i]['userId']==1 and todos[i]['completed']==True:
4         print(todos[i])
5         cnt += 1
6 print('-' * 20)
7 print(f'총 갯수 - {cnt}')
```

```
1 df_todos = pd.DataFrame(todos, columns=['userId', 'id', 'title', 'completed'])
2 df_todos
```

JSON

```
1 df_todos.to_csv('output/todos.csv', index=False)
```

```
1 ls output
```

todos.csv - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

userId,id,title,completed

1,1,delectus aut autem,False

1,2,quis ut nam facilis et officia qui,False

1,3,fugiat veniam minus,False

1,4,et porro tempora,True

1,5,laboriosam mollitia et enim quasi adipisci quia provident illum,False

JSON 퀴즈

1) json 샘플 사이트를 이용하여 딕셔너리 리스트 형태로 변경하고 첫번째 회원의 데이터만 출력하여라

<http://jsonplaceholder.typicode.com/users>

2) 모두 몇명인가?

3) id, name, email 만 출력하여라

4) id, city, phone만 출력하여라

5) phone 이 '210.067.6132'인 회원은 누구인가?

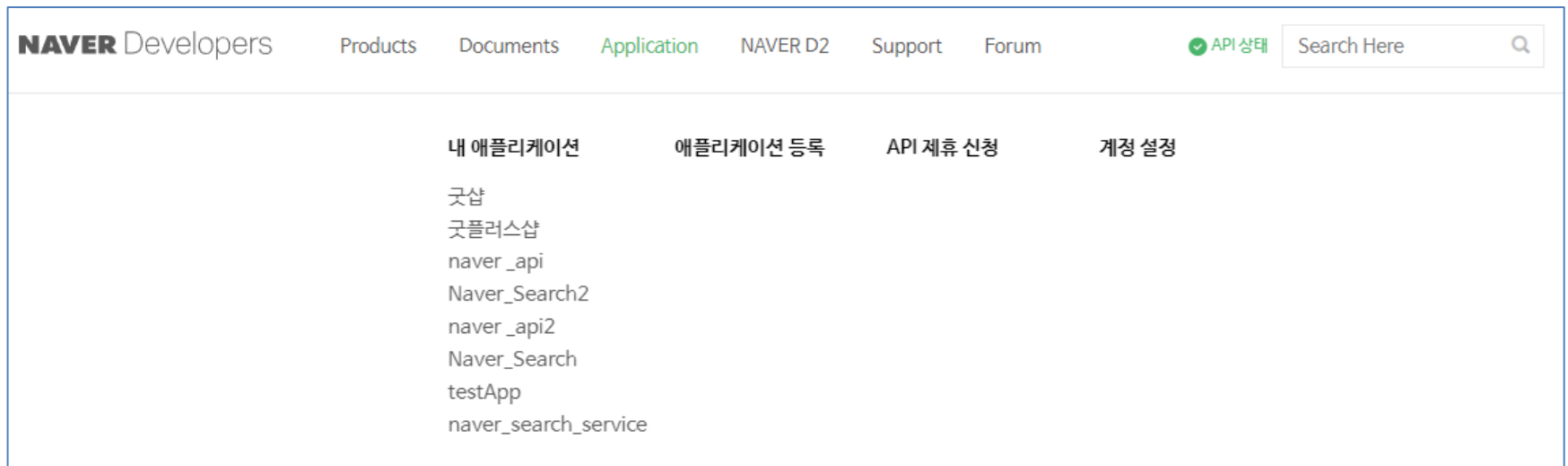
6) name이 'C'로 시작하는 회원목록만 아래와 같은 형태로 출력하여라

7) 데이터프레임으로 변경하여라

	id	name	username	email	address	phone	website	company
0	1	Leanne Graham	Bret	Sincere@april.biz	{'street': 'Kulas Light', 'suite': 'Apt. 556',...	1-770-736-8031 x56442	hildegard.org	{'name': 'Romaguera-Crona', 'catchPhrase': 'Mu...
1	2	Ervin Howell	Antonette	Shanna@melissa.tv	{'street': 'Victor Plains', 'suite': 'Suite 87...	010-692-6593 x09125	anastasia.net	{'name': 'Deckow-Crist', 'catchPhrase': 'Proac...

NAVER API 검색 서비스

- 네이버 개발자 센터 사이트에서 오픈API 신청 - 네이버 아이디로 로그인
<https://developers.naver.com/apps/#/register?defaultScope=search>



NAVER API 검색 서비스

- 어플리케이션 등록 : 검색/Web 설정 으로 신청

애플리케이션 이름 ⇄	<input type="text" value="naver_search_service"/> ✓ • 네이버 로그인할 때 사용자에게 표시되는 이름이므로 서비스 브랜드를 대표할 수 있는 이름으로 가급적 10자 이내로 간결하게 설정해주세요. • 40자 이내의 영문, 한글, 숫자, 공백문자, 쉼표(,), "/" , "-" , "_" , 만 입력 가능합니다.
사용 API ⇄	<div> <input type="text" value="선택하세요."/> ▼ ✓ </div> <div> <input type="text" value="검색"/> ✕ </div>
비로그인 오픈 API 서비스 환경	<div> <input type="text" value="환경 추가"/> ▼ </div> <div> <div>WEB 설정 ✕ ^</div> <div> <input type="text" value="웹 서비스 URL (최대 10개)"/> <input type="text" value="http://naver.com"/> + ✓ </div> </div>

NAVER API 검색 서비스

- 네이버 서비스키와 비번 정보 확인 [Application]-[내 애플리케이션]

naver_search_service

개요	API 설정	멤버관리	로그인 통계	API 통계	Playground(Beta)
----	--------	------	--------	--------	------------------

애플리케이션 정보

Client ID	AGNKsP_38rD7z7lbPNxz
Client Secret 보기

NAVER API 검색 서비스

- 파이썬 예시 소스 참조

<https://developers.naver.com/docs/serviceapi/search/blog>

naver_search_service

[개요](#)[API 설정](#)[멤버관리](#)[로그인 통계](#)[API 통계](#)[Playground \(Beta\)](#)

애플리케이션 정보

Client ID

AGNKsP_38rD7z7lbPNxz

Client Secret

.....

[보기](#)[Documents](#) > [서비스 API](#) > [검색](#)[블로그](#)[뉴스](#)[책](#)[성인 검색어 판별](#)[백과사전](#)[영화](#)[카페글](#)[지식iN](#)[지역](#)[오타변환](#)[웹문서](#)[이미지](#)[쇼핑](#)[전문자료](#)[검색 > 블로그](#)

• 블로그 검색 개요

- 개요
- 사전 준비 사항


• 블로그 검색 API 레퍼런스

- 블로그 검색 결과 조회
- 오류 코드

• 검색 API 블로그 검색 구현 예제

- Java
- PHP
- Node.js
- Python
- C#

NAVER API 검색 서비스

Python 

```
# 네이버 검색 API 예제 - 블로그 검색
import os
import sys
import urllib.request
client_id = "YOUR_CLIENT_ID"
client_secret = "YOUR_CLIENT_SECRET"
encText = urllib.parse.quote("검색할 단어")
url = "https://openapi.naver.com/v1/search/blog?query=" + encText # JSON 결과
# url = "https://openapi.naver.com/v1/search/blog.xml?query=" + encText # XML 결과
request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)
response = urllib.request.urlopen(request)
rescode = response.getcode()
if(rescode==200):
    response_body = response.read()
    print(response_body.decode('utf-8'))
else:
    print("Error Code:" + rescode)
```

NAVER API 검색 서비스 - 블로그

- 블로그 검색 서비스 이용하기

```
1 import json
2 import pandas as pd
3 import requests
4 import urllib.request
```

```
1 # 본인 아이디와 시크릿 키 설정 |
2 client_id = "본인 아이디"
3 client_secret = "시크릿 키"
```

```
1 encText = urllib.parse.quote("송도 맛집")
2 url = "https://openapi.naver.com/v1/search/blog?display=50&query=" + encText
3 print(url)
```

<https://openapi.naver.com/v1/search/blog?display=50&query=%EC%86%A1%EB%8F%84%20%EB%A7%9B%EC%A7%91>

```
1 request = urllib.request.Request(url)
2 request.add_header("X-Naver-Client-Id", client_id)
3 request.add_header("X-Naver-Client-Secret", client_secret)
4 response = urllib.request.urlopen(request)
5 rescode = response.getcode()
6
7 if(rescode==200):
8     response_body = response.read()
9     print(response_body.decode('utf-8'))
10 else:
11     print("Error Code:" + rescode)
```

NAVER API 검색 서비스 - 블로그

```
1 print(type(response_body))
2 blog_dict = json.loads(response_body.decode('utf-8'))
3 print(type(blog_dict))
4 blog_dict
```

```
<class 'bytes'>
<class 'dict'>
```

```
{'lastBuildDate': 'Tue, 25 Oct 2022 23:08:50 +0900',
 'total': 396866,
 'start': 1,
 'display': 50,
 'items': [{'title': '<b>송도 맛집</b> 추천 푸짐한 송도 한정식 밥상편지',
 'link': 'https://blog.naver.com/joojoo5623/222901759267',
 'description': '제이유입니다 :) 오늘 소개해 드릴 <b>송도 맛집</b> 밥상편지는 송도 한정식을 맛볼 수 있는 곳이에요. 이전에 한번 다녀온 적이 있는데 맛있게 먹었던 기억이 있어 부모님 두 분을 모시고 다녀왔어요 밥상편지 주소...',
 'bloggername': '바삭바삭한 일상그리기',
 'bloggerlink': 'blog.naver.com/joojoo5623',
 'postdate': '20221017'}]}
```

```
1 blog_list = blog_dict['items']
2 print(type(blog_list))
```

```
<class 'list'>
```

```
1 len(blog_list)
```

```
50
```

NAVER API 검색 서비스 - 블로그

```
1 blog_list[0]
```

```
{'title': '<b>송도 맛집</b> 추천 푸짐한 송도 한정식 밥상편지',
 'link': 'https://blog.naver.com/joojoo5623/222901759267',
 'description': '제이유입니다 :) 오늘 소개해 드릴 <b>송도 맛집</b> 밥상편지는 송도 한정식을 맛볼 수 있는 곳
이예요. 이전에 한번 다녀온 적이 있는데 맛있게 먹었던 기억이 있어 부모님 두 분을 모시고 다녀왔어요 밥상편지 주
소...',
 'bloggername': '바삭바삭한 일상그리기🍴',
 'bloggerlink': 'blog.naver.com/joojoo5623',
 'postdate': '20221017'}
```

```
1 df = pd.DataFrame(blog_list)
2 df.head()
```

	title	link	description	bloggername	bloggerlink	postdate
0	송도 맛집 추천 푸짐한 송도 한정식 밥상편지	https://blog.naver.com/joojoo5623/222901759267	제이유입니다 :) 오늘 소개해 드릴 송도 맛집 밥상편지는 송도 한정식을...	바삭바삭한 일상그리기🍴	blog.naver.com/joojoo5623	20221017

NAVER API 검색 서비스 - 블로그

```

1 result_list = []
2 for item in blog_list:
3     title = item['title'].replace('<b>', '').replace('</b>', '')
4     description = item['description'].replace('<b>', '').replace('</b>', '')
5     bloggername = item['bloggername']
6     postdate = item['postdate']
7     result_list.append([title, description, bloggername, postdate])

```

```

1 df = pd.DataFrame(result_list, columns=['제목', '내용', '블로거', '등록날짜'])
2 df.head()

```

	제목	내용	블로거	등록날짜
0	송도 맛집 추천 푸짐한 송도 한정식 밥상편지	제이유입니다 :) 오늘 소개해 드릴 송도 맛집 밥상편지는 송도 한정식을 맛볼 수 있...	바삭바삭한 일상그리기	20221017
1	마담 샤브칼국수 막국수 송도맛집 탕수육맛집	면을 좋아하시는 분들이라면 꼭 드시면 너무 좋을 것 같아요! #송도샤브샤브 #송도생...	건강한 하루를 보내는 일기	20221023
2	호감이었던 부산 송도 맛집	친구들과 평일 휴가를 맞춰 현지인이 추천해준 부산 송도 맛집에 다녀왔어요. 칼칼하면...	고테네 사진관	20220716
3	부산 송도 맛집 제대로 찾은 곳	며칠전 친구와 함께 송도로 나들이를 떠났다가 미리 찾아둔 부산 송도 맛집에 방문했어...	Ranco Holic	20221020

```

1 df.to_csv('output/송도맛집.csv')

```

NAVER API 검색 서비스 - 지식인

- 지식인 검색 서비스 이용하기

<https://developers.naver.com/docs/serviceapi/search/kin/kin.md#%EC%A7%80%EC%8B%9DiN>



NAVER API 검색 서비스 - 지식인

```
1 import json
2 import pandas as pd
3 import requests
4 import urllib.request
```

```
1 # 본인 아이디와 시크릿 키 설정 |
2 client_id = "본인 아이디"
3 client_secret = "시크릿 키"
```

```
1 encText = urllib.parse.quote("해운대 코스")
2 url = "https://openapi.naver.com/v1/search/kin.json?sort=date&display=50&query=" + encText
3 print(url)
```

```
1 request = urllib.request.Request(url)
2 request.add_header("X-Naver-Client-Id", client_id)
3 request.add_header("X-Naver-Client-Secret", client_secret)
4 response = urllib.request.urlopen(request)
5 rescode = response.getcode()
6
7 if(rescode==200):
8     response_body = response.read()
9     print(response_body.decode('utf-8'))
10 else:
11     print("Error Code:" + rescode)
```

```
1 know_dict = json.loads(response_body.decode('utf-8'))
2 print(type(know_dict))
3 know_list = know_dict['items']
4 print(type(know_list))
```

NAVER API 검색 서비스 - 지식인

```
1 know_list[0]
```

```
{'title': '부산 오랫동안 가서',  
 'link': 'https://kin.naver.com/qna/detail.naver?d1id=9&dirId=90111&docId=431220652&qb=7ZW07Jq064yAl  
0y910yKpA==&enc=utf8&section=kin.qna&rank=1&search_sort=2&spq=0',  
 'description': '부산여행 당일 <b>코스</b> 좀 잡아주세요 . 오전10시부터 오후6시까지 있지 싶습니다. 당일치  
기면 가장까지는 무리일거같고 광안리,<b>해운대</b> / 남포, 영도 이쪽중에 고르셔야 할거같네요 광안리를 간다하  
면 광안리 바닷가, 광안대교... '}
```

```
1 df = pd.DataFrame(know_list)  
2 df.sample(2)
```

	title	link	description
19	2박 3일 부산여행 코스 질문드립니다 ㅎㅎ	https://kin.naver.com/qna/detail.naver?d1id=9&...	... 부산의 대표적인 해운대, 광안리, 송정 해수욕장 코스...
7	부산 환상 좀 깨주세요ㅠ	https://kin.naver.com/qna/detail.naver?d1id=9&...	... 생각나요 직장을 부산에서 잡고 정착하면 되죠~~ 환상을 현실로 만드는겁니다~...

```
1 result_list = []  
2 for item in know_list:  
3     title = item['title'].replace('<b>', '').replace('</b>', '')  
4     description = item['description'].replace('<b>', '').replace('</b>', '')  
5     link = item['link']  
6     result_list.append([title, description, link])
```

NAVER API 검색 서비스 - 지식인

```
1 df = pd.DataFrame(result_list, columns=['제목', '내용', '링크'])
2 df.sample(2)
```

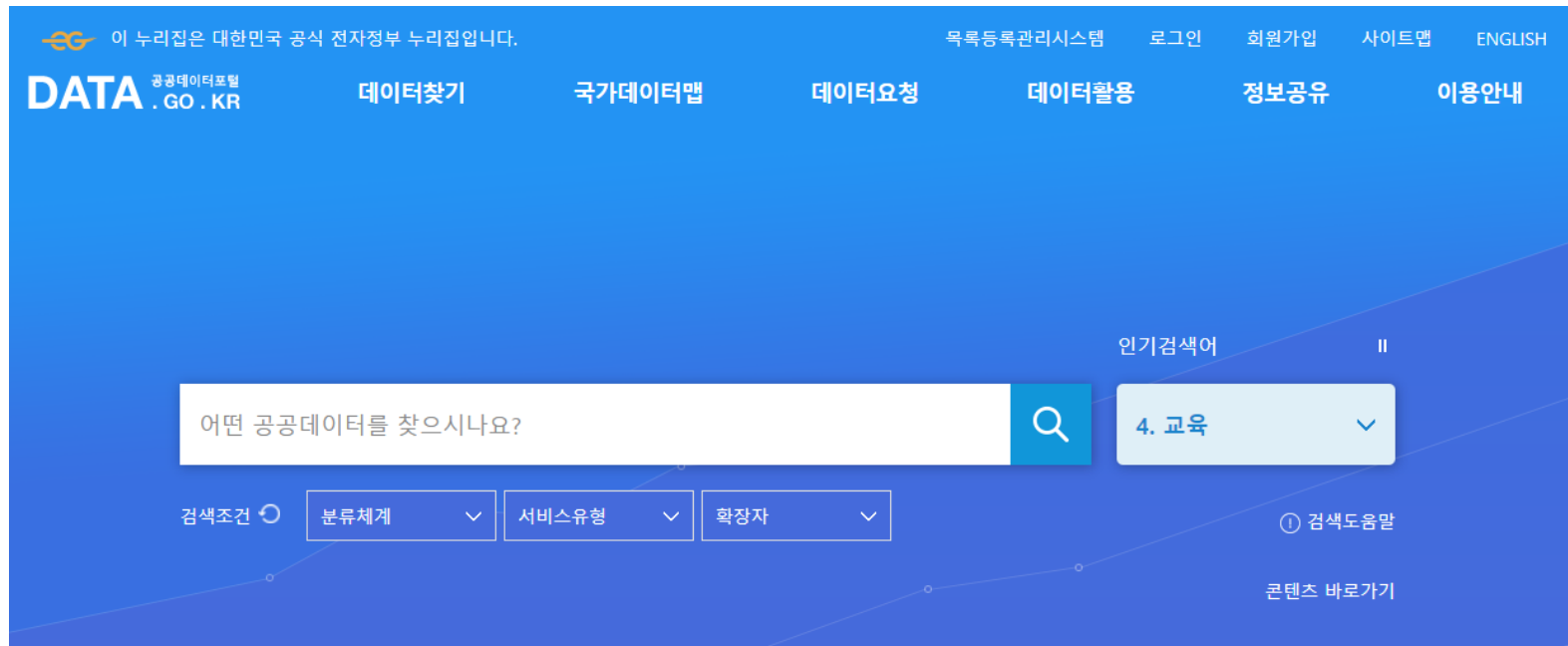
	제목	내용	링크
40	cj택배배송 언제 올까요?	... 04 16:38:40 보내시는 고객님의로부터 상품을 인수받 았습니다 부산해운대...	https://kin.naver.com/qna/detail.naver? d1id=8&...
39	부산 맛집 (갑각류맛집 위주로) 추천부탁드려요!!	... 해운대랩스터맛집 덴포라 추천드려요. !!! 코스요리맛집 인데 모임맛집으로도 유...	https://kin.naver.com/qna/detail.naver? d1id=12...

```
1 df.to_csv('output/해운대지식인검색결과.csv')
```

```
1 ls output/*.csv
```

공공데이터포털의 API 서비스

- <https://www.data.go.kr/>
- 회원 가입후에야 API서비스 신청 가능




미세먼지(금속성분) 실시간 정보

- 환경부 국립환경과학원
- 조회날짜, 항목코드, 측정소코드, 시간구분을 기준으로 중금속 성분 측정 결과를 2시간 평균, 24시간 평균 측정 수치 자료로 제공하는 서비스
- <https://www.data.go.kr/tcs/dss/selectApiDataDetailView.do?publicDataPk=15016368>
- [활용신청] 클릭후 서비스키 정보 확인 : [마이페이지]-[오픈 API]-[개발 계정]

환경부 국립환경과학원_미세먼지(금속성분) 실시간 정보

[활용신청](#)

국립환경과학원 대기연구분야 데이터 입니다.(대기중 중금속 성분 2시간, 24시간 평균 측정 자료)

 3 0 관심

미세먼지(금속성분) 실시간 정보

요청변수(Request Parameter)

항목명(국문)	항목명(영문)	항목크기	항목구분	샘플데이터	항목설명
서비스키	ServiceKey	4	필수	-	공공데이터포털에서 받은 인증키
페이지 번호	pageNo	4	옵션	1	페이지번호
한 페이지 결과 수	numOfRows	4	옵션	10	한 페이지 결과 수
결과형식	resultType	4	옵션	XML	결과형식(XML/JSON)
검색조건 날짜	date	8	옵션	20171208	검색조건 날짜 (예시 : 20171208)
검색조건 측정소 코드	stationcode	6	옵션	1	검색조건 측정소코드
검색조건 항목코드	itemcode	5	옵션	90303	검색조건 항목코드
검색조건 시간구분	timecode	4	옵션	RH02	검색조건 시간구분

미세먼지(금속성분) 실시간 정보

출력결과(Response Element)

항목명(국문)	항목명(영문)	항목크기	항목구분	샘플데이터	항목설명
결과코드	resultCode	2	필수	00	결과코드
결과메시지	resultMsg	50	필수	OK	결과메시지
한 페이지 결과 수	numOfRows	4	필수	10	한 페이지 결과 수
페이지 번호	pageNo	4	필수	1	페이지번호
전체 결과 수	totalCount	4	필수	3	전체 결과 수
날짜	sdate	14	필수	20171208040000	날짜
측정소 코드	stationcode	1	필수	1	측정소 코드
측정항목 코드	itemcode	5	필수	90303	측정항목 코드
측정데이터 시간 구분	timecode	4	필수	RH02	측정데이터 시간 구분
측정수치	value	8	필수	7.5	측정수치(단위:ng/m3)

미세먼지(금속성분) 실시간 정보

샘플코드

Java

Javascript

C#

PHP

Curl

Objective-C

Python

Nodejs

R

Python3 샘플 코드

```
import requests
```

```
url = 'http://apis.data.go.kr/1480523/MetalMeasuringResultService/MetalService'
```

```
params = {'serviceKey' : '서비스키', 'pageNo' : '1', 'numOfRows' : '10', 'resultType' : 'XML', 'date' : '20171208', 'stationcode' : '1', 'itemcode' : '90303', 'timecode' : 'RH02' }
```

```
response = requests.get(url, params=params)
```

```
print(response.content)
```

미세먼지(금속성분) 실시간 정보

```
1 import json
2 import pandas as pd
3 import requests
4 import urllib.request
```

```
1 url = 'http://apis.data.go.kr/1480523/MetalMeasuringResultService/MetalService'
2 service_key = 시크릿키값
3 params = {'serviceKey' : service_key,
4           'pageNo' : '1',
5           'numOfRows' : '12',
6           'resultType' : 'JSON',
7           'date' : '20220101',
8           'stationcode' : '3',
9           'itemcode' : '90303',
10          'timecode' : 'RH02'}
11
12 response = requests.get(url, params=params)
13 print(response)
14 print(response.text)
```

미세먼지(금속성분) 실시간 정보

```
<Response [200]>
{"MetalService":{"header":{"code":"00","message":"NORMAL SERVICE"},"item":[{"SDATE":"20220602000000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":0.3307},{ "SDATE":"20220602020000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":0.1621},{ "SDATE":"20220602040000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":3.6916},{ "SDATE":"20220602060000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":8.8124},{ "SDATE":"20220602080000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":0.9511},{ "SDATE":"20220602100000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":1.9522},{ "SDATE":"20220602120000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":2.3849},{ "SDATE":"20220602140000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":2.3718},{ "SDATE":"20220602160000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":1.914},{ "SDATE":"20220602180000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":0.6874},{ "SDATE":"20220602200000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":0.0927},{ "SDATE":"20220602220000","STATIONCODE":1,"ITEMCODE":"90303","TIMECODE":"RH02","VALUE":0.4508}], "rows":null, "numOfRows":12, "pageNo":1, "totalCount":12}}
```

```
1 dust = json.loads(response.content)
2 dust
```

```
1 dust['MetalService']['item']
```

미세먼지(금속성분) 실시간 정보

```
1 df_dust = pd.DataFrame(dust['MetalService']['item'], columns=['SDATE', 'STATIONCODE', 'ITEMCODE', 'TIMECODE', 'VALUE'])
2 df_dust
```

	SDATE	STATIONCODE	ITEMCODE	TIMECODE	VALUE
0	20220101000000	3	90303	RH02	3.2933
1	20220101020000	3	90303	RH02	2.3088
2	20220101040000	3	90303	RH02	3.7324
3	20220101060000	3	90303	RH02	6.3342
4	20220101080000	3	90303	RH02	6.3293
5	20220101100000	3	90303	RH02	6.4359
6	20220101120000	3	90303	RH02	4.2071
7	20220101140000	3	90303	RH02	3.4410
8	20220101160000	3	90303	RH02	2.9664
9	20220101180000	3	90303	RH02	48.9368
10	20220101200000	3	90303	RH02	112.6491

미세먼지(금속성분) 실시간 정보

station code 1~10

레코드 저장

1 수도권 2 백령도

3 호남권 4 중부권

5 제주권 6 영남권

7 경기권 8 충청권

9 전북권 10 강원권

```
1 url = 'http://apis.data.go.kr/1480523/MetalMeasuringResultService/MetalService'
2 service_key = 서비스키
3
4 dust_list = []
5 for i in range(1, 11):
6     params = {'serviceKey' : service_key,
7               'pageNo' : '1',
8               'numOfRows' : '12',
9               'resultType' : 'JSON',
10              'date' : '20220510',
11              'stationcode' : str(i),
12              'itemcode' : '90303',
13              'timecode' : 'RH02'}
14
15     response = requests.get(url, params=params)
16     try:
17         temp = json.loads(response.text)
18         dust_list += temp['MetalService']['item']
19     except:
20         pass
21     print(i)
```

미세먼지(금속성분) 실시간 정보

```
1 len(dust_list)
```

```
1 dust_list[0]
```

```
: {'SDATE': '20220510000000',  
  'STATIONCODE': 1,  
  'ITEMCODE': '90303',  
  'TIMECODE': 'RH02',  
  'VALUE': 8.4252}
```

```
1 df_dust2 = pd.DataFrame(dust_list, columns=['SDATE', 'STATIONCODE', 'ITEMCODE', 'TIMECODE', 'VALUE'])  
2 df_dust2.sample(2)
```

	SDATE	STATIONCODE	ITEMCODE	TIMECODE	VALUE
54	20220510120000	6	90303	RH02	3.8903
53	20220510100000	6	90303	RH02	4.6156

퀴즈

- 고속도로 휴게소별 날씨 정보를 제공하는 API 서비스 신청하기
<https://www.data.go.kr/data/15076661/openapi.do>
<http://data.ex.co.kr/openapi/basicinfo/openApiInfoM?apild=0508>
- 신청한 api 서비스를 이용하여 특정 날짜(예시-2022년10월15일)의 날씨 데이터를 리스트 딕셔너리 형태로 저장하여라.
- 경부선에 위치한 휴게소의 날씨 데이터만 출력하여라
- 판다스의 데이프레임 구조로 저장하고 출력하여라
- 2022년 10월 1~5 일간의 데이터를 API 서비스를 이용하여 추출하고 데이타프레임으로 저장하여라

웹페이지 명세서 (인증키 확인)

<http://data.ex.co.kr/openapi/basicinfo/openApiInfoM?apild=0508&pn=-1>

퀴즈

	unitName	unitCode	sdate	stdHour	routeNo	routeName	updownTypeCode	xValue	yValue
0	죽전휴게소	002	20221001	10	0010	경부선	E	127.104165	37.332651
1	안성휴게소(서울)	004	20221001	10	0010	경부선	E	127.132081	37.076107
2	옥산휴게소(서울)	031	20221001	10	0010	경부선	E	127.371859	36.657168
3	천안(삼)휴게소(서울)	033	20221001	10	0010	경부선	E	127.173092	36.787388
933	벌곡휴게소(논산)	043	20221005	10	2510	호남지선	S	127.273793	36.217488
934	현풍휴게소(현풍)	090	20221005	10	4510	중부내륙지선	S	128.436431	35.701990

935 rows × 38 columns

XML 개요

- eXtensible Markup Language
- 다목적의 성격을 가진 마크업언어
- XML은 미리 정의된 태그 집합이나 문법을 규정하고 있지 않기 때문에 완전한 확장성을 가진다.
- XML 문서 작성자는 원하는 태그를 사용할 수 있으며, 태그에 원하는 속성을 지정할 수 있으며, 원하는 형태로 태그를 중첩시킬 수 있다.
- https://www.w3schools.com/xml/xml_tree.asp

XML 예시

짝이 있는 태그 형태

<태그 속성=값>

내용

</태그>

짝이 없는 태그 형태 : 내용이 없음

<태그 />

예시 - 노트

<?xml version="1.0" encoding="UTF-8"?>

<note>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me this weekend!</body>

</note>

BS4를 이용한 XML 파싱

```
1 import requests
2 from bs4 import BeautifulSoup
3 import pandas as pd
```

```
1 with open('bookstore.xml', 'r') as f:
2     xml_str = f.read()
3     print(type(xml_str))
4     print()
5     print(xml_str)
```

```
1 soup = BeautifulSoup(xml_str, 'xml')
2 soup
```

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

BS4를 이용한 XML 파싱

```

1 target_tag = soup.find('book')
2 print(target_tag)
3 print()
4 print(target_tag.text)
5 print()
6 print(target_tag['category'])
7 print(target_tag.get('category'))

```

```

1 print(target_tag.title)
2 print(target_tag.title.text)
3 print(target_tag.title['lang'])

```

```

<book category="cooking">
<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
<year>2005</year>
<price>30.00</price>
</book>

```

```

Everyday Italian
Giada De Laurentiis
2005
30.00

```

```

cooking
cooking

```

```

<title lang="en">Everyday Italian</title>
Everyday Italian
en

```

```

1 print(target_tag.find('title'))
2 print(target_tag.find('title').text)
3 print(target_tag.find('title')['lang'])

```

```

<title lang="en">Everyday Italian</title>
Everyday Italian
en

```

BS4를 이용한 XML 파싱

```
1 target_list = soup.find_all('book')
2 len(target_list)
```

```
1 target_list[0]
```

```
<book category="cooking">
<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
<year>2005</year>
<price>30.00</price>
</book>
```

```
1 book_list = []
2 for book in target_list:
3     category = book['category']
4     title = book.title.text
5     author = book.author.text
6     year = book.year.text
7     price = book.price.text
8     book_list.append([category, title, author, year, price])
9
10 book_list
```

```
1 for book in target_list:
2     print('category => ', book['category'])
3     print('title => ', book.title.text)
4     print('author => ', book.author.text)
5     print('year => ', book.year.text)
6     print('price => ', book.price.text)
7     print('='*50)
```

```
[['cooking', 'Everyday Italian', 'Giada De Laurentiis', '2005', '30.00'],
 ['children', 'Harry Potter', 'J K. Rowling', '2005', '29.99'],
 ['web', 'Learning XML', 'Erik T. Ray', '2003', '39.95']]
```

BS4를 이용한 XML 파싱

```
1 df = pd.DataFrame(book_list,
2                     columns=[ 'category', 'title', 'author', 'year', 'price' ])
3 df
```

	category	title	author	year	price
0	cooking	Everyday Italian	Giada De Laurentiis	2005	30.00
1	children	Harry Potter	J K. Rowling	2005	29.99
2	web	Learning XML	Erik T. Ray	2003	39.95

```
1 print(soup.find(lang="en"))
2 print(soup.find_all(lang="en"))
3 print()
4 print(soup.find('title', {'lang':'en'}))
5 print(soup.find('book', {'lang':'en'}))
```

```
<title lang="en">Everyday Italian</title>
[<title lang="en">Everyday Italian</title>, <title lang="en">Harry Potter</title>,
<title lang="en">Learning XML</title>]
```

```
<title lang="en">Everyday Italian</title>
None
```

퀴즈

country.xml 파일을 open() 함수를 이용하여 불러오기한 후
파싱 과정을 거쳐 아래와 같은 데이터프레임으로 변경하여라
neighbor 컬럼의 값은 리스트안의 튜플구조를 이용한다

	name	rank	year	gdppc	neighbor
0	Liechtenstein	1	2008	141100	[(Austria, E), (Switzerland, W)]
1	Singapore	4	2011	59900	[(Malaysia, N)]
2	Panama	68	2011	13600	[(Costa Rica, W), (Colombia, E)]

xml 소스

<https://docs.python.org/3/library/xml.etree.elementtree.html>

퀴즈

food_menu.xml파일을 `open()` 함수를 이용하여 불러오기한 후
파싱 과정을 거쳐 아래와 같은 데이터프레임으로 변경하여라

	메뉴명	가격	칼로리	메뉴 설명
0	Belgian Waffles	\$5.95	650 cal	Two of our famous Belgian Waffles with plenty ...
1	Strawberry Belgian Waffles	\$7.95	900 cal	Light Belgian waffles covered with strawberrie...
2	Berry-Berry Belgian Waffles	\$8.95	900 cal	Light Belgian waffles covered with an assortme...
3	French Toast	\$4.50	600 cal	Thick slices made from our homemade sourdough ...
4	Homestyle Breakfast	\$6.95	950 cal	Two eggs, bacon or sausage, toast, and our eve...

기상청 전국날씨 RSS

RSS(Really Simple Syndication, Rich Site Summary)란?

블로그처럼 콘텐츠 업데이트가 자주 일어나는 웹사이트에서,
업데이트된 정보를 쉽게 구독자들에게 제공하기 위해 XML을 기초
로 만들어진 데이터 형식

기상청 전국날씨 RSS – 시간별 예보

- <https://www.weather.go.kr/w/index.do> 하단 [RSS] 메뉴 클릭
- [동네예보 > 시간별 예보] 에서 지역 지정 후 [RSS] 클릭
예시) 부산광역시 수영구 광안1동
- 주소 복사 : <http://www.kma.go.kr/wid/queryDFSRSS.jsp?zone=2650076000>
- XML 가이드 =>

https://www.kma.go.kr/images/weather/lifenindustry/timeseries_XML.pdf

| 부서·직원찾기 | 문서보기 프로그램 다운받기 | RSS

전국 국번없이
기상콜센터 131



< 추천콘텐츠

www.weather.go.kr 내용:

Ctrl + c 를 눌러 클립보드로 복사하세요

<http://www.kma.go.kr/wid/queryDFSRSS.jsp?zone=2650076000>

확인

취소

동네예보 > 시간별 예보

시도

부산광역시



선택

구군

수영구



선택

읍면동

광안제1동



선택

RSS

기상청 전국날씨 RSS – 시간별 예보

```
1 import requests
2 from bs4 import BeautifulSoup
3 import pandas as pd
```

```
1 url = 'http://www.kma.go.kr/wid/queryDFSRSS.jsp?zone=2650076000'
2 res = requests.get(url)
3 xml_str = res.text
4 xml_str
```

```
1 soup = BeautifulSoup(xml_str, 'xml')
2 soup
```

```
1 target_list = soup.find_all('data')
2 len(target_list)
```

17

```
1 target_list[0]
```

```
<data seq="0">
<hour>24</hour>
<day>0</day>
<temp>15.0</temp>
<tmx>-999.0</tmx>
<tmn>-999.0</tmn>
<sky>4</sky>
<pty>0</pty>
<wfKor>흐림</wfKor>
<wfEn>Cloudy</wfEn>
<pop>30</pop>
<r12>0.0</r12>
<s12>0.0</s12>
<ws>3.7</ws>
<wd>0</wd>
<wdKor>북</wdKor>
<wdEn>N</wdEn>
<reh>75</reh>
<r06>0.0</r06>
<s06>0.0</s06>
</data>
```

기상청 전국날씨 RSS – 시간별 예보

```

1 day_dict = {'0':'오늘', '1':'내일', '2':'모레'}
2 day = day_dict[target_list[0].day.text]
3 hour = target_list[0].hour.text
4 temp = target_list[0].temp.text
5
6
7 sky_dict = {'1':'맑음', '2':'구름조금', '3':'구름많음', '4':'흐림'}
8 sky = sky_dict[target_list[0].sky.text]
9
10 wf = target_list[0].wfKor.text + '/' + target_list[0].wfEn.text
11
12 pty_dict = {'0':'없음', '1':'비', '2':'비/눈', '3':'눈/비', '4':'눈'}
13 pty = pty_dict[target_list[0].pty.text]
14
15 pop = target_list[0].pop.text + '%'
16 ws = target_list[0].ws.text

```

```

1 print(soup.category.text)
2 print()
3 print('날짜 시간 기온 하늘상태 날씨상태 강수상태 강수확률 풍속(m/s)')
4 print('='*80)
5 print(f'{day} {hour:>3} {temp:>5} {sky:>5} {wf:>11} {pty:>5} {pop:>7}
6       {float(ws):10.2f}')

```

부산광역시 수영구 광안제1동

날짜 시간 기온 하늘상태 날씨상태 강수상태 강수확률 풍속(m/s)

=====

오늘	24	15.0	흐림	흐림/Cloudy	없음	30 %	3.70
----	----	------	----	-----------	----	------	------

기상청 전국날씨 RSS – 시간별 예보

```

1 wether_list = []
2 day_dict = {'0':'오늘', '1':'내일', '2':'모레'}
3 sky_dict = {'1':'맑음', '2':'구름조금', '3':'구름많음', '4':'흐림'}
4 pty_dict = {'0':'없음', '1':'비', '2':'비/눈', '3':'눈/비', '4':'눈'}
5
6 for w in target_list:
7     day = day_dict[w.day.text]
8     hour = w.hour.text
9     temp = w.temp.text
10    sky = sky_dict[w.sky.text]
11    wf = w.wfKor.text + '/' + w.wfEn.text
12    pty = pty_dict[w.pty.text]
13    pop = w.pop.text + '%'
14    ws = f'{float(w.ws.text):.2f}'
15    wether_list.append([day, hour, temp, sky, wf, pty, pop, ws])

```

```

1 df = pd.DataFrame(wether_list,
2                   columns=['날짜', '시간', '기온', '하늘상태',
3                           '날씨상태', '강수상태', '강수량', '풍속(m/s)'])
4 df

```

	날짜	시간	기온	하늘상태	날씨상태	강수상태	강수량	풍속(m/s)
0	오늘	24	15.0	흐림	흐림/Cloudy	없음	30 %	3.70
1	내일	3	15.0	흐림	흐림/Cloudy	없음	30 %	4.00
2	내일	6	15.0	흐림	흐림/Cloudy	없음	30 %	3.90
3	내일	9	17.0	흐림	흐림/Cloudy	없음	30 %	3.70

기상청 전국날씨 RSS – 중기 예보

- <https://www.weather.go.kr/w/index.do> 하단 [RSS] 메뉴 클릭
- [동네예보 > 중기 예보] 에서 해당 지역의 [RSS] 클릭
- 주소 복사 : 서울.경기도 예시

<https://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId=109>

- XML 가이드

https://www.weather.go.kr/w/resources/pdf/midtermforecast_rss.pdf

▪ 동네예보 > 중기예보

중기 예보	전국	RSS	전라북도	RSS
	서울·경기도	RSS	전라남도	RSS
	강원도	RSS	경상북도	RSS
	충청북도	RSS	경상남도	RSS
	충청남도	RSS	제주특별자치도	RSS

기상청 전국날씨 RSS – 중기 예보

```
1 url = 'http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId=109'
2 res = requests.get(url)
3 xml_str = res.text
```

```
1 soup = BeautifulSoup(xml_str, 'xml')
2 soup
```

```
1 target_list = soup.find_all('location')
2 len(target_list)
```

35

```
1 target_list[0]
```

```
1 target_data_list[0]
```

```
1 target_data_list = target_list[0].find_all('data')
2 len(target_data_list)
```

```
<data>
<mode>A02</mode>
<tmEf>2022-10-29 00:00</tmEf>
<wf>맑음</wf>
<tmn>10</tmn>
<tmx>18</tmx>
<reliability/>
<rnSt>0</rnSt>
</data>
```

13

기상청 전국날씨 RSS – 중기 예보

```
1 province = target_list[0].province.text
2 city = target_list[0].city.text
3 tmEf = target_list[0].find_all('data')[0].tmEf.text
4 wf = target_list[0].find_all('data')[0].wf.text
5 tmn = target_list[0].find_all('data')[0].tmn.text
6 tmx = target_list[0].find_all('data')[0].tmx.text
7 rnSt = target_list[0].find_all('data')[0].rnSt.text
8
9 print('지역 : ', province)
10 print('도시 : ', city)
11 print('날짜 : ', tmEf)
12 print('날씨 : ', wf)
13 print('최저온도 : ', tmn)
14 print('최고온도 : ', tmx)
15 print('강수확률 : ', rnSt, '%')
```

지역 : 서울 · 인천 · 경기도
도시 : 서울
날짜 : 2022-10-29 00:00
날씨 : 맑음
최저온도 : 10
최고온도 : 18
강수확률 : 0 %

기상청 전국날씨 RSS – 중기 예보

```
1 province = target_list[0].province.text
2 city = target_list[0].city.text
3 tmEf = target_list[0].find_all('data')[1].tmEf.text
4 wf = target_list[0].find_all('data')[1].wf.text
5 tmn = target_list[0].find_all('data')[1].tmn.text
6 tmx = target_list[0].find_all('data')[1].tmx.text
7 rnSt = target_list[0].find_all('data')[1].rnSt.text
8
9 print('지역 : ', province)
10 print('도시 : ', city)
11 print('날짜 : ', tmEf)
12 print('날씨 : ', wf)
13 print('최저온도 : ', tmn)
14 print('최고온도 : ', tmx)
15 print('강수확률 : ', rnSt, '%')
```

지역 : 서울·인천·경기도

도시 : 서울

날짜 : 2022-10-29 12:00

날씨 : 구름많음

최저온도 : 10

최고온도 : 18

강수확률 : 20 %

기상청 전국날씨 RSS – 중기 예보

```

1 midterm_forecast_list = []
2
3 for location in target_list:
4     province = location.province.text
5     city = location.city.text
6
7     target_data_list = location.find_all('data')
8     for data in target_data_list:
9         tmEf = data.tmEf.text
10        wf = data.wf.text
11        tmn = data.tmn.text
12        tmx = data.tmx.text
13        rnSt = data.rnSt.text
14
15        midterm_forecast_list.append([province, city, tmEf, wf, tmn, tmx, rnSt])

```

```

1 df = pd.DataFrame(midterm_forecast_list,
2                   columns=['지역', '도시', '날짜', '날씨',
3                           '최저기온', '최고기온', '강수량'])
4 df

```

	지역	도시	날짜	날씨	최저기온	최고기온	강수량
0	서울 · 인천 · 경기도	서울	2022-10-29 00:00	맑음	10	18	0
1	서울 · 인천 · 경기도	서울	2022-10-29 12:00	구름많음	10	18	20
2	서울 · 인천 · 경기도	서울	2022-10-30 00:00	맑음	9	18	0

퀴즈

1) 동네예보 > 중기예보 > 전국 에서 도시명(city 태그 이용)만 출력하여라
[중기예보]-[전국]

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

['서울', '인천', '수원', '파주', '이천', '평택', '춘천', '원주', '강릉', '대전', '세종', '홍성', '청주', '충주', '영동', '광주', '목포', '여수', '순천', '광양', '나주', '전주', '군산', '정읍', '남원', '고창', '무주', '부산', '울산', '창원', '진주', '거창', '통영', '대구', '안동', '포항', '경주', '울진', '울릉도', '제주', '서귀포']

2) '동네예보 > 중기예보 > 전국' 의 날씨 데이터를 아래와 같은 형태로 데이터프레임으로 변경하여라

	도시	날짜	날씨	최저기온	최고기온	강수량
0	서울	2022-06-10 00:00	구름많음	18	28	40
1	서울	2022-06-10 12:00	구름많음	18	28	30
2	서울	2022-06-11 00:00	구름많음	19	29	30
3	서울	2022-06-11 12:00	구름많음	19	29	40
4	서울	2022-06-12 00:00	구름많음	19	28	20
...

퀴즈

3) '동네예보 > 중기예보 > 전국' 에서 부산, 대구의 날씨 데이터만 아래와 같은 형태로 데이터프레임으로 변경하여라

	도시	날짜	날씨	최저기온	최고기온	강수량
0	부산	2022-06-10 00:00	구름많음	18	24	20
1	부산	2022-06-10 12:00	맑음	18	24	10
2	부산	2022-06-11 00:00	구름많음	18	25	20
3	부산	2022-06-11 12:00	구름많음	18	25	20
4	부산	2022-06-12 00:00	구름많음	19	25	20
5	부산	2022-06-12 12:00	구름많음	19	25	30
6	부산	2022-06-13 00:00	구름많음	18	25	30
19	대구	2022-06-13 00:00	구름많음	18	28	30
20	대구	2022-06-13 12:00	흐림	18	28	40
21	대구	2022-06-14 00:00	흐림	18	26	40
22	대구	2022-06-14 12:00	흐림	18	26	40
23	대구	2022-06-15 00:00	흐림	19	27	40
24	대구	2022-06-16 00:00	구름많음	18	29	30
25	대구	2022-06-17 00:00	구름많음	20	29	30

부산광역시_부산명소정보 서비스

- 부산관광명소의 이름, 주소, 홈페이지, 전화번호, 좌표, 상세정보를 조회하는 서비스
- <https://www.data.go.kr/data/15063481/openapi.do>
- [활용신청] 클릭후 서비스키 정보 확인 [마이페이지]-[오픈 API]-[개발 계정]

- 활용승인 절차 개발단계 : 자동승인 / 운영단계 :
- 신청가능 트래픽 개발계정 : 10,000 / 운영계정 : 활용사례 등록시 신청하면 트래픽 증가 가능
- 요청주소 <http://apis.data.go.kr/6260000/AttractionService/getAttractionKr>
- 서비스URL <http://apis.data.go.kr/6260000/AttractionService>

[활용신청](#)

요청변수(Request Parameter)

항목명(국문)	항목명(영문)	항목크기	항목구분	샘플데이터	항목설명
서비스키	ServiceKey	4	필	-	공공데이터포털에서 받은 인증키
페이지 번호	pageNo	4	필	1	페이지번호
한 페이지 결과 수	numOfRows	4	필	10	한 페이지 결과 수
JSON방식 호출	resultType	4	옵		JSON방식으로 호출 시 파라미터 resultType=json 입력
콘텐츠 ID	UC_SEQ	11	옵		콘텐츠 ID

부산광역시_부산명소정보 서비스

출력결과(Response Element)

항목명(국문)	항목명(영문)	항목크기	항목구분	샘플데이터	항목설명
결과코드	resultCode	2	필	00	결과코드
결과메시지	resultMsg	50	필	OK	결과메시지
한 페이지 결과 수	numOfRows	4	필	10	한 페이지 결과 수
페이지 번호	pageNo	4	필	1	페이지번호
전체 결과 수	totalCount	4	필	107	전체 결과 수
콘텐츠ID	UC_SEQ	11	필	255	콘텐츠ID
콘텐츠명	MAIN_TITLE	300	옵	흰여울문화마을	콘텐츠명
구군	GUGUN_NM	100	옵	영도구	구군
위도	LAT	20	옵	35.07885	위도
경도	LNG	20	옵	129.04402	경도
여행지	PLACE	300	옵	흰여울문화마을	여행지

부산광역시_부산명소정보 서비스

제목	TITLE	300	옵	가파른 절벽 끝에 흰여울문화마을	제목
부제목	SUBTITLE	300	옵	흰여울길에서 만난 느림의 미학	부제목
주소	ADDR1	200	옵	부산광역시 영도구 흰여울길	주소
연락처	CNTCT_TEL	200	옵	051-419-4067	연락처
홈페이지	HOMEPAGE_URL	200	옵	http://huinnyeoul.co.kr	홈페이지
교통정보	TRFC_INFO	500	옵	도시철도 1호선 남포역 6번 출구 → 7, 71, 508 버스 환승 → 영선동 백련사 하차 버스 7,71,508 영선동 백련사 하차 부산시티투어버스 부 산역(점보버스) → 흰여울문화마을 하차 주차 절영해안산책로 입구 공 영 주차장 (유료)	교통정보
운영일	USAGE_DAY	500	옵		운영일
휴무일	HLDY_INFO	500	옵		휴무일
운영 및 시간	USAGE_DAY_WEEK_A ND_TIME	500	옵		운영 및 시간
이용요금	USAGE_AMOUNT	500	옵	무료	이용요금

부산광역시_부산명소정보 서비스

편의시설	MIDDLE_SIZE_RM1	500	옵	점보버스 휠체어 리프트 장착 차량 운행, 요금 할인 50%, 장애인 주차장(절영해안산책로 공영주차장, 유료)	편의시설
이미지URL	MAIN_IMG_NORMAL	500	옵	/uploadimgs/files/cntnts/20191222164810529_ttiel	이미지URL
썸네일이미지URL	MAIN_IMG_THUMB	500	옵	/uploadimgs/files/cntnts/20191222164810529_thumbL	썸네일이미지URL
상세내용	ITEMCNTNTS	1000	옵		

```
1 import requests
2 from bs4 import BeautifulSoup
3 import pandas as pd
```

```
1 url = 'http://apis.data.go.kr/6260000/AttractionService/getAttractionKr'
2 myKey = 신청키
3 params = {'serviceKey' : myKey,
4           'pageNo' : '1',
5           'numOfRows' : '20' }
6
7 response = requests.get(url, params=params)
8 print(response)
```

```
1 xml_str = response.text
2 soup = BeautifulSoup(xml_str, 'xml')
3 print(soup)
```

부산광역시_부산명소정보 서비스

```
1 target_list = soup.find_all('item')
2 len(target_list)
```

```
1 target_list[-1]
```

```
1 title = target_list[0].TITLE.text
2 addr = target_list[0].ADDR1.text
3 usage_amount = target_list[0].USAGE_AMOUNT.text
4 cntct_tel = target_list[0].CNTCT_TEL.text
5 homepage_url = target_list[0].HOMEPAGE_URL.text
6 trfc_info = target_list[0].TRFC_INFO0.text
7
8 print('제목 :', title)
9 print('주소 :', addr)
10 print('요금 :', usage_amount)
11 print('연락처 :', cntct_tel)
12 print('홈페이지 :', homepage_url)
13 print('교통정보 :', trfc_info)
```

제목 : 가파른 절벽 끝에 흰여울문화마을
 주소 : 부산광역시 영도구 흰여울길
 요금 : 무료
 연락처 : 051-419-4067
 홈페이지 : <http://www.ydculture.com/huinnyeoulculturetown/>
 교통정보 : 도시철도 1호선 남포역 6번 출구 → 7, 71, 508 버스 환승 → 영선동 백련사 하차
 버스 7, 71, 508 영선동 백련사 하차
 부산시티투어버스 부산역(점보버스) → 흰여울문화마을 하차
 주차 절영해안산책로 입구 공영 주차장 (유료)

부산광역시_부산명소정보 서비스

```

1 data_list = []
2 for item in target_list:
3     data_list.append({'제목' : item.TITLE.text,
4                       '주소' : item.ADDR1.text,
5                       '요금' : item.USAGE_AMOUNT.text,
6                       '연락처' : item.CNTCT_TEL.text,
7                       '홈페이지' : item.HOMEPAGE_URL.text,
8                       '교통정보' : item.TRFC_INFO.text
9                       })
10 print(len(data_list))

```

```

1 df = pd.DataFrame(data_list)
2 df

```

	제목	주소	요금	연락처	홈페이지	교통정보
0	가파른 절벽 끝에 흰여울 문화마을	부산광역시 영도구 흰여울길	무료	051-419-4067	http://www.ydculture.com/huinnyeoulculturetown/	도시철도 1호선 남포역 6번 출구 → 7, 71, 508 버스 환승 → 영선동 백련...
1	역사가 살아 숨 쉬는 강강이예술마을	부산시 영도구 대평북로 36 강강이 안내센터	프로그램별 상이	051-418-3336	http://kangkangee.com	도시철도 1호선 남포역 (6번출구) 도보 18분 \n마을 버스 영도구2 대평동 하차 ...
2	해양 문화의 꽃 국립해양 박물관	부산시 영도구 해양로301번길 45	관람료 무료(단, 40영상관, 유료특별전시 제외)	051-309-1900	https://www.mmk.or.kr/	도시철도 1호선 남포역 6번 출구 → 186, 66번 버스 환승 → 국립해양 박물관 ...

퀴즈

국토교통부_아파트매매 실거래 상세 자료 API 신청하기

<https://www.data.go.kr/tcs/dss/selectApiDataDetailView.do?publicDataPk=15057511>

<https://www.data.go.kr/iim/api/selectAPIAccountView.do>

'부산광역시 수영구 광안동'의 2022년 5월 아파트 매매 데이터를 API로 신청하고 데이터프레임으로 출력하여라

국토교통부_아파트매매 실거래 상세 자료

활용신청

부동산 거래신고에 관한 법률에 따라 신고된 주택의 실거래 자료를 제공

46

4

관심

	apartment_name	build_year	area	floor	dong	road_name	road_name_bonbun	jibun	deal_amount	deal_day	dealer_lawdnm
0	산호맨션	1986	84.15	5	망미동	연수로249번길	00020	834-3	28,300	2022.5.10	부산 수영구
1	삼성	1991	84.59	4	망미동	망미로30번길	00023	777	37,900	2022.5.16	부산 수영구
2	망미한신	1989	84.975	2	망미동	망미배산로70번길	00045	880-84	30,000	2022.5.27	부산 수영구
3	수영협성트네상스타운	2004	84.9468	6	수영동	수영로741번길	00046	432-1	90,000	2022.5.4	부산 수영구