라이브러리

In [4]:

```
import time
from selenium import webdriver
import re
from bs4 import BeautifulSoup
import unicodedata
import pandas as pd # 저장할때
```

자동 크롤링

In [5]:

```
# 자동 크롤링
driver = webdriver.Chrome("./chromedriver.exe")
time.sleep(1)

driver.get('https://www.naver.com/')
time.sleep(2)

search = '코로나현황'
# input_id = driver.find_elements_by_css_selector('input_text')
# input_id.clear()
element = driver.find_element_by_name('query')
element.send_keys(search)
element.submit()
time.sleep(2)

#클릭하기
def select_first(driver):
    first = driver.find_elements_by_css_selector('#_cs_production_type > div > div:nth-child(4)
    first.click()
    time.sleep(2)

select_first(driver)
```

현재 페이지 **html** 정보 가져오기

In [10]:

```
#현재 페이지 html 정보 가져오기
html = driver.page_source
html
```

Out[10]:

```
'<html lang="ko" data-useragent="mozilla/5.0 (windows nt 10.0; win64; x64) applew
ebkit/537.36 (khtml, like gecko) chrome/92.0.4515.159 safari/537.36" data-platfor
m="win32"><head> <meta charset="utf-8"> <meta name="referrer" content="always">
<meta name="format-detection" content="telephone=no,address=no,email=no"> <meta n
ame="viewport" content="width=device-width,initial-scale=1.0,maximum-scale=2.0">
<meta property="og:title" content="코로나현황 : 네이버 통합검색"> <meta property
="og:image" content="https://ssl.pstatic.net/sstatic/search/common/og_v3.png"> <m
eta property="og:description" content="\'코로나현황\'의 네이버 통합검색 결과입니
다."> <meta name="description" lang="ko" content="\'코로나현황\'의 네이버 통합검
색 결과입니다."> <title>코로나현황 : 네이버 통합검색</title> <link rel="shortcut
icon" href="https://ssl.pstatic.net/sstatic/search/favicon/favicon_191118_pc.ic
o">  <link rel="search" type="application/opensearchdescription+xml" href="http
s://ssl.pstatic.net/sstatic/search/opensearch-description.https.xml" title="Nave
r"><link rel="stylesheet" type="text/css" href="https://ssl.pstatic.net/sstatic/s
earch/pc/css/search1_210812.css"> <link rel="stylesheet" type="text/css" href="ht
tps://ssl.pstatic.net/sstatic/search/pc/css/search2_210812a.css"> <link rel="styl
esheet" type="text/css" href="https://ssl.pstatic.net/sstatic/search/pc/css/sp_au
tocomplete_210318.css"><script type="text/javascript"> if (!String.prototype.tri
```

In [6]:

```
soup = BeautifulSoup(html, 'html.parser')
# BeautifulSoup의 html.parser 를 통해 우리가 파이썬으로 읽을 수 있게 데이터를 변환시켜 준다
```

In [9]:

```
print(soup) #물론 우리가 큰 변화를 느낄 수는 없다
```

```
<html data-platform="win32" data-useragent="mozilla/5.0 (windows nt 10.0; win64;
x64) applewebkit/537.36 (khtml, like gecko) chrome/92.0.4515.159 safari/537.36" l
ang="ko"><head> <meta charset="utf-8"/> <meta content="always" name="referrer"/>
<meta content="telephone=no,address=no,email=no" name="format-detection"/> <meta
content="width=device-width,initial-scale=1.0,maximum-scale=2.0" name="viewport"/
> <meta content="코로나현황 : 네이버 통합검색" property="og:title"/> <meta conten
t="https://ssl.pstatic.net/sstatic/search/common/og_v3.png" property="og:image"/>
<meta content="'코로나현황'의 네이버 통합검색 결과입니다." property="og:descripti
on"/> <meta content="'코로나현황'의 네이버 통합검색 결과입니다." lang="ko" name
="description"/> <title>코로나현황 : 네이버 통합검색</title> <link href="https://
ssl.pstatic.net/sstatic/search/favicon/favicon_191118_pc.ico" rel="shortcut ico
n"/> <link href="https://ssl.pstatic.net/sstatic/search/opensearch-description.ht
tps.xml" rel="search" title="Naver" type="application/opensearchdescription+xml"/
><link href="https://ssl.pstatic.net/sstatic/search/pc/css/search1_210812.css" re
l="stylesheet" type="text/css"/> <link href="https://ssl.pstatic.net/sstatic/sear
ch/pc/css/search2_210812a.css" rel="stylesheet" type="text/css"/> <link href="htt
ps://ssl.pstatic.net/sstatic/search/pc/css/sp_autocomplete_210318.css" rel="style
sheet" type="text/css"/><script type="text/javascript"> if (!String.prototype.tri
m) { String.prototype.trim = function () { return this.replace(/^[\s\uFEFF\xA0]+|
```

In [5]:

```
#지역 개수
print(len(soup.select('strong.local_name')))

#누적 수치 개수
print(len(soup.select('p.population_number')))

#몇개를 가져왔는데 수치로 나타내 보았다.
```

```
20
20
```

본문 내용 가져오기

각각 크롤링한 데이터들을 우리가 읽을 수 있게 변환해준다.

In [6]:

```
#지역별 누적 인구수 텍스트화 하기
population_number = []
for i in range(20):
    population_number.append(soup.select('p.population_number')[i].text)
print(population_number)
```

```
['5,738', '2,494', '76,245', '11,305', '3,920', '3,920', '6,802', '10,999', '13,19
2', '5,242', '68,158', '2,479', '4,953', '6,547', '6,547', '5,413', '944', '3,396',
'9,615', '3,997']
```

In [7]:

```
#지역별 누적 인구수 텍스트화 하기
local_name = []
for i in range(20):
    local_name.append(soup.select('strong.local_name')[i].text)
print(local_name)
```

```
['검역자세히', '제주', '서울', '인천', '광주', '광주', '경북', '부산', '대구', '강
원', '경기', '전남', '충북', '충남', '충남', '대전', '세종', '전북', '경남', '울산']
```

In [8]:

```
# 둘의 데이터를 데이터 프라임에 넣어주기 위해를 zip 함수를 통해 list로 합쳐 주었다.
result=[[i,v] for i,v in zip(local_name,population_number)]
result
```

Out[8]:

```
[['검역자세히', '5,738'],
 ['제주', '2,494'],
 ['서울', '76,245'],
 ['인천', '11,305'],
 ['광주', '3,920'],
 ['광주', '3,920'],
 ['경북', '6,802'],
 ['부산', '10,999'],
 ['대구', '13,192'],
 ['강원', '5,242'],
 ['경기', '68,158'],
 ['전남', '2,479'],
 ['충북', '4,953'],
 ['충남', '6,547'],
 ['충남', '6,547'],
 ['대전', '5,413'],
 ['세종', '944'],
 ['전북', '3,396'],
 ['경남', '9,615'],
 ['울산', '3,997']]
```

크롤링 결과 저장하기

In [9]:

```
# 만든결과를 잃어버리지 않게 저장해준다.
results_df = pd.DataFrame(result)
results_df.columns = ['지역','누적 인구']
results_df.to_excel('../미니프로젝트/코로나 지역별 누적인구.xlsx',index=False)
```

In [ ]:

```

```

excel데이터 SQL로 옮기기

In [24]:

```
#우리가 저장한 excel 데이터를 가져와서 준비한다.

import pandas as pd
import pymysql

data = pd.read_excel("./코로나 지역별 누적인구.xlsx")
print(data)
```

```
      지역   누적 인구
0   검역자세히   5,738
1      제주   2,494
2      서울  76,245
3      인천  11,305
4      광주   3,920
5      광주   3,920
6      경북   6,802
7      부산  10,999
8      대구  13,192
9      강원   5,242
10     경기  68,158
11     전남   2,479
12     충북   4,953
13     충남   6,547
14     충남   6,547
15     대전   5,413
16     세종     944
17     전북   3,396
18     경남   9,615
19     울산   3,997
```

In [25]:

```
#SQL과 연결시켜주기
conn = pymysql.connect(host='127.0.0.1', user='root', password='root', db='coronadb',charset =
#원격 마우스가 생겼다.
cur = conn.cursor()

```

In [26]:

```
cur.execute('create table if not exists corona19(지역이름 char(5), 누적_인구 char(7))')
```

Out[26]:

```
0
```

In [ ]:

```
#테이블 만들고 저장하기!
conn.commit()
```

In [27]:

```
지역이름_v=data['지역']
누적_인구_v=data['누적 인구']
```

In [28]:

```
지역이름 = []
for i in range(len(지역이름_v)):
    지역이름.append(지역이름_v[i])
print(지역이름)
```

```
['검역자세히', '제주', '서울', '인천', '광주', '광주', '경북', '부산', '대구', '강
원', '경기', '전남', '충북', '충남', '충남', '대전', '세종', '전북', '경남', '울산']
```

In [29]:

```
누적_인구 = []
for i in range(len(누적_인구_v)):
    누적_인구.append(누적_인구_v[i])
print(누적_인구)
```

```
['5,738', '2,494', '76,245', '11,305', '3,920', '3,920', '6,802', '10,999', '13,19
2', '5,242', '68,158', '2,479', '4,953', '6,547', '6,547', '5,413', '944', '3,396',
'9,615', '3,997']
```

In [31]:

```
#반복문을 통해서 테이블에 넣어주기

for i in range(1,len(지역이름)):
    a =지역이름.loc[i][0]
    b =누적_인구.loc[i][1]
    cur.execute(f"INSERT INTO corona19 VALUES({a}, {b}")
```

In [32]:

```
#처음에는 반복 노동

# cur.execute("INSERT INTO corona19 VALUES( '검역자세히' , '5,738' )")
# cur.execute("INSERT INTO corona19 VALUES( '제주' , '2,494' )")
# cur.execute("INSERT INTO corona19 VALUES( '서울' , '76,245')")
# cur.execute("INSERT INTO corona19 VALUES( '인천' , '11,305')")
# cur.execute("INSERT INTO corona19 VALUES( '광주' ,  '3,920')")
# cur.execute("INSERT INTO corona19 VALUES( '광주' ,  '3,920')")
# cur.execute("INSERT INTO corona19 VALUES( '경북' ,  '6,802')")
# cur.execute("INSERT INTO corona19 VALUES( '부산' , '10,999')")
# cur.execute("INSERT INTO corona19 VALUES( '대구' , '13,192')")
# cur.execute("INSERT INTO corona19 VALUES( '강원' ,  '5,242')")
# cur.execute("INSERT INTO corona19 VALUES( '경기' , '68,158')")
# cur.execute("INSERT INTO corona19 VALUES( '전남' ,  '2,479')")
# cur.execute("INSERT INTO corona19 VALUES( '충북' , '4,953' )")
# cur.execute("INSERT INTO corona19 VALUES( '충남' ,  '6,547')")
# cur.execute("INSERT INTO corona19 VALUES( '충남' ,  '6,547')")
# cur.execute("INSERT INTO corona19 VALUES( '대전' , '5,413' )")
# cur.execute("INSERT INTO corona19 VALUES( '세종' ,   '944' )")
# cur.execute("INSERT INTO corona19 VALUES( '전북' ,  '3,396')")
# cur.execute("INSERT INTO corona19 VALUES( '경남' , '9,615' )")
# cur.execute("INSERT INTO corona19 VALUES( '울산' , '3,997' )")
```

Out[32]:

```
1
```

In [33]:

```
#끝날때만 두개 동시에
conn.commit()
#끝내기를 해야 서버가 닫힌다!
conn.close()
```

일주일치 신규확진자 저장하기

In [ ]:

```

import pandas as pd
import pymysql

data = pd.read_excel("./week.xlsx")
print(data)

conn = pymysql.connect(host='127.0.0.1', user='root', password='root', db='coronadb',charset =
cur = conn.cursor()


#테이블을 만들자
cur.execute("CREATE TABLE IF not EXISTS corona_week(날짜 varchar(10), 국내발생 int(8),해외유입
#만들 테이블을 저장해 주자
conn.commit()
```

In [ ]:

```
for i in range(1,len(data)):
    a =data.loc[i][0]
    b =data.loc[i][1]
    c =data.loc[i][2]
    d =data.loc[i][3]
    cur.execute(f"INSERT INTO corona_week VALUES('{a}', {b}, {c}, {d})")

conn.commit()

conn.close()
```

In [ ]:

```

```

In [1]:

```
import pandas as pd
import pymysql

data = pd.read_excel("./코로나 지역별 누적인구.xlsx")
```

In [2]:

```
#SQL데이터 불러오기 ,튜플 -> 리스트 변환
# 지역 and 누적 확진자수
conn = pymysql.connect(host = '127.0.0.1', user = 'root' , password = 'root', db = 'coronadb')
cur = conn.cursor()
cur.execute("SELECT * FROM corona19;")
fetchall = cur.fetchall()

local = []
for i in fetchall:
    local.append(i)
local
```

Out[2]:

```
[('검역자세히', '5,738'),
 ('제주', '2,494'),
 ('서울', '76,245'),
 ('인천', '11,305'),
 ('광주', '3,920'),
 ('광주', '3,920'),
 ('경북', '6,802'),
 ('부산', '10,999'),
 ('대구', '13,192'),
 ('강원', '5,242'),
 ('경기', '68,158'),
 ('전남', '2,479'),
 ('충북', '4,953'),
 ('충남', '6,547'),
 ('충남', '6,547'),
 ('대전', '5,413'),
 ('세종', '944'),
 ('전북', '3,396'),
 ('경남', '9,615'),
 ('울산', '3,997')]
```

In [3]:

```
#데이터 프레임 만들기
col_name = ['지역', '누적_인구']
localdp = pd.DataFrame(local, columns=col_name)
```

In [4]:

```
#데이터 확인 중복이 있다.
localdp
```

Out[4]:

	지역	누적_인구
0	검역자세히	5,738
1	제주	2,494
2	서울	76,245
3	인천	11,305
4	광주	3,920
5	광주	3,920
6	경북	6,802
7	부산	10,999
8	대구	13,192
9	강원	5,242
10	경기	68,158
11	전남	2,479
12	충북	4,953
13	충남	6,547
14	충남	6,547
15	대전	5,413
16	세종	944
17	전북	3,396
18	경남	9,615
19	울산	3,997

In [5]:

```
#중복 행 없애주기
localdp=localdp.drop([0],axis = 0)
localdp=localdp.drop([4],axis = 0)
localdp=localdp.drop([14],axis = 0)
#인덱스 다시 잡아주기
localdp=localdp.set_index('지역')
localdp=localdp.reset_index(drop = False)

localdp
```

In [6]:

```
# 없어진 열 확인
n=len(localdp)
print(f'인덱스 총 개수는 {n}개 입니다')
```

	지역	누적_인구
0	제주	2,494
1	서울	76,245
2	인천	11,305
3	광주	3,920
4	경북	6,802
5	부산	10,999
6	대구	13,192
7	강원	5,242
8	경기	68,158
9	전남	2,479
10	충북	4,953
11	충남	6,547
12	대전	5,413
13	세종	944
14	전북	3,396
15	경남	9,615
16	울산	3,997

인덱스 총 개수는 17개 입니다

In [8]:

```
places = []
for i in localdp['지역']:
    places.append(i)
print(places)
```

['제주', '서울', '인천', '광주', '경북', '부산', '대구', '강원', '경기', '전남', '충북', '충남', '대전', '세종', '전북', '경남', '울산']

In [9]:

```
# googlemaps를 사용하기위한 라이브러리 다운
# !pip install -U googlemaps
```

In [10]:

```
import googlemaps
import pandas as pd

my_key = "AIzaSyCwsP9RBvIKTMKdLOilSob3R_4dhaJO6iQ"
maps = googlemaps.Client(key=my_key)  # my key값 입력
lat = []  #위도
lng = []  #경도

# 위치를 찾을 장소나 주소를 넣어준다.
places = ['제주시', '서울특별시', '인천광역시', '광주광역시', '경상북도', '부산광역시', '대구',
          '충정북도', '충정남도', '대전광역시', '세종특별자치시청', '전라북도', '경상남도', '울

i=0
for place in places:
    i = i + 1
    try:
        print("%d번 인덱스에서 %s의 위치를 찾고있습니다"%(i, place))
        geo_location = maps.geocode(place)[0].get('geometry')
        lat.append(geo_location['location']['lat'])
        lng.append(geo_location['location']['lng'])


    except:
        lat.append('')
        lng.append('')
        print("%d번 인덱스 위치를 찾는데 실패했습니다."%(i))


# 데이터프레임만들어 출력하기
df = pd.DataFrame({'위도':lat, '경도':lng}, index=places)
print(df)
```

```
1번 인덱스에서 제주시의 위치를 찾고있습니다
2번 인덱스에서 서울특별시의 위치를 찾고있습니다
3번 인덱스에서 인천광역시의 위치를 찾고있습니다
4번 인덱스에서 광주광역시의 위치를 찾고있습니다
5번 인덱스에서 경상북도의 위치를 찾고있습니다
6번 인덱스에서 부산광역시의 위치를 찾고있습니다
7번 인덱스에서 대구의 위치를 찾고있습니다
8번 인덱스에서 강원의 위치를 찾고있습니다
9번 인덱스에서 경기도의 위치를 찾고있습니다
10번 인덱스에서 전라남도의 위치를 찾고있습니다
11번 인덱스에서 충정북도의 위치를 찾고있습니다
12번 인덱스에서 충정남도의 위치를 찾고있습니다
13번 인덱스에서 대전광역시의 위치를 찾고있습니다
14번 인덱스에서 세종특별자치시청의 위치를 찾고있습니다
15번 인덱스에서 전라북도의 위치를 찾고있습니다
16번 인덱스에서 경상남도의 위치를 찾고있습니다
17번 인덱스에서 울산광역시의 위치를 찾고있습니다
                위도          경도
제주시        33.499621  126.531188
서울특별시      37.566535  126.977969
인천광역시      37.456256  126.705206
광주광역시      35.159545  126.852601
경상북도       36.491900  128.888900
부산광역시      35.179554  129.075642
대구         35.871435  128.601445
강원         37.822800  128.155500
```

```
경기도          37.413800  127.518300
전라남도         34.867900  126.991000
충정북도         36.800000  127.700000
충정남도         36.518400  126.800000
대전광역시        36.350412  127.384547
세종특별자치시청   36.480132  127.288765
전라북도         35.717500  127.153000
경상남도         35.460600  128.213200
울산광역시        35.538377  129.311360
```

In [11]:

```
location_df=df
location_df.head()
```

Out[11]:

	위도	경도
제주시	33.499621	126.531188
서울특별시	37.566535	126.977969
인천광역시	37.456256	126.705206
광주광역시	35.159545	126.852601
경상북도	36.491900	128.888900

In [12]:

```
#이름 바꾸기
name = ['제주시', '서울특별시', '인천광역시', '광주광역시', '경상북도', '부산광역시', '대구', '

localdp['지역']=name
```

In [13]:

```
#데이터 프레임 합치기
location_data = pd.merge( localdp,location_df, how = 'right', left_on = '지역',right_index = Tr

location_data
```

Out[13]:

	지역	누적_인구	위도	경도
0	제주시	2,494	33.499621	126.531188
1	서울특별시	76,245	37.566535	126.977969
2	인천광역시	11,305	37.456256	126.705206
3	광주광역시	3,920	35.159545	126.852601
4	경상북도	6,802	36.491900	128.888900
5	부산광역시	10,999	35.179554	129.075642
6	대구	13,192	35.871435	128.601445
7	강원	5,242	37.822800	128.155500
8	경기도	68,158	37.413800	127.518300
9	전라남도	2,479	34.867900	126.991000
10	충정북도	4,953	36.800000	127.700000
11	충정남도	6,547	36.518400	126.800000
12	대전광역시	5,413	36.350412	127.384547
13	세종특별자치시청	944	36.480132	127.288765
14	전라북도	3,396	35.717500	127.153000
15	경상남도	9,615	35.460600	128.213200
16	울산광역시	3,997	35.538377	129.311360

In [ ]:

```
# location_data.to_excel('../미니프로젝트/지역_인구_위도_경도.xlsx',index=False)
```

In [14]:

```
location_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 17 entries, 0 to 16
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   지역      17 non-null     object 
 1   누적_인구   17 non-null     object 
 2   위도      17 non-null     float64
 3   경도      17 non-null     float64
dtypes: float64(2), object(2)
memory usage: 680.0+ bytes
```

In [15]:

```
#누적_인구 타입변환 object => int
for i in localdp['누적_인구']:
    a=i.replace(',','')
    print(a,end = ',')
```

```
2494,76245,11305,3920,6802,10999,13192,5242,68158,2479,4953,6547,5413,944,3396,9615,
3997,
```

In [16]:

```
# 컬럼 값 바꾸기
int_v = [2494,76245,11305,3920,6802,10999,13192,5242,68158,2479,4953,6547,5413,944,3396,9615,39
localdp['누적_인구'] = int_v
```

In [19]:

```
# 바뀌것 확인하기
localdp
```

Out[19]:

	지역	누적_인구
0	제주시	2494
1	서울특별시	76245
2	인천광역시	11305
3	광주광역시	3920
4	경상북도	6802
5	부산광역시	10999
6	대구	13192
7	강원	5242
8	경기도	68158
9	전라남도	2479
10	충청북도	4953

In [20]:

```
# 데이터 프레임 합쳐주기
location_data = pd.merge( localdp,location_df, how = 'right', left_on = '지역',right_index = Tr
location_data
```

Out[20]:

	지역	누적_인구	위도	경도
0	제주시	2494	33.499621	126.531188
1	서울특별시	76245	37.566535	126.977969
2	인천광역시	11305	37.456256	126.705206
3	광주광역시	3920	35.159545	126.852601
4	경상북도	6802	36.491900	128.888900
5	부산광역시	10999	35.179554	129.075642
6	대구	13192	35.871435	128.601445
7	강원	5242	37.822800	128.155500
8	경기도	68158	37.413800	127.518300
9	전라남도	2479	34.867900	126.991000
10	충정북도	4953	36.800000	127.700000
11	충정남도	6547	36.518400	126.800000
12	대전광역시	5413	36.350412	127.384547
13	세종특별자치시청	944	36.480132	127.288765
14	전라북도	3396	35.717500	127.153000
15	경상남도	9615	35.460600	128.213200
16	울산광역시	3997	35.538377	129.311360

In [21]:

```
# 지도 라이브러리

#!pip install folium
```

In [24]:

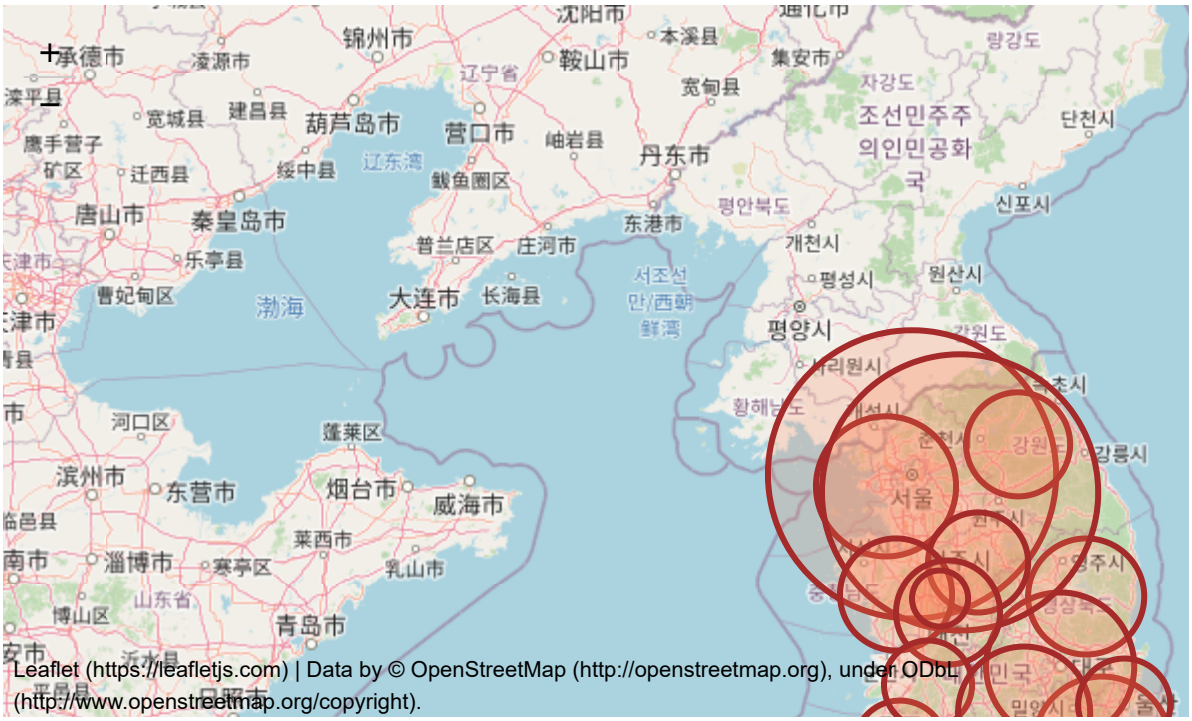
```
#지도 표시해보기
import folium

대한민국_가운데 = [36.523522, 127.732926]
대한민국 = folium.Map(location = 대한민국_가운데, zoom_start = 6)

for i in range(len(location_data)):
    name = location_data ['지역'][i]         # 지역 위치
    count = location_data ['누적_인구'][i]   # 인구수
    size = int(count)**0.38
    long = float(location_data['위도'][i])
    lat = float(location_data['경도'][i])
    folium.CircleMarker([long,lat], radius = size, color='brown', fill = True,fill_color = 'cor

대한민국
```

Out[24]:



In [ ]:

```
# 결론: 데이터를 통해 코로나 누적 확진자 수를 한번에 알 수 있게 되었고 이를 통해
#       우리는 서울과 경기 지역에서 특히 마스크를 잘 쓰며 조심히 다녀야 합니다.
```