CSSE COVID-19 Dataset

เป็นข้อมูลสรุปรายงานการติดเชื้อ COVID-19 จากทั่วโลกตั้งแต่วันที่ 22 มกราคม 2020 ถึง 12 มีนาคม 2020

ประกอบด้วยไฟล์

- time_series_covid19_confirmed_global.csv
- time_series_covid19_deaths_global.csv
- time_series_covid19_recovered_global.csv

Field description

- **Province/State**: China - province name; US/Canada/Australia/ - city name, state/province name; Others - name of the event (e.g., "Diamond Princess" cruise ship); other countries - blank.
- **Country/Region**: country/region name conforming to WHO (will be updated).
- **Confirmed**: the number of confirmed cases(accumulated).
- **Deaths**: the number of deaths(accumulated).
- **Recovered**: the number of recovered cases(accumulated).

จากไฟล์จงทำการเขียนโปรแกรมเพื่อตอบคำถามต่อไปนี้

1. จงแสดงอัตราการรักษาหายของเฉพาะผู้ป่วยในสหภาพยุโรปเป็นเปอร์เซ็นต์
2. จงหา Country/Region ที่มียอดจำนวนผู้ติดเชื้อล่าสุดมากที่สุด 10 อันดับพร้อมพล็อตกราฟแสดงจำนวนผู้ติดเชื้อของแต่ละ Country/Region โดยเรียงลำดับจากมากไปน้อย (ไม่ต้องแยกเป็น Province/State)
3. จงพล็อตกราฟเส้นเพื่อแสดง trend ยอดผู้ติดเชื้อในประเทศไทยในแต่ละสัปดาห์(กำหนดให้นับข้อมูลทุกวันอาทิตย์)
4. จงหายอดผู้ติดเชื้อของ**แต่ละวัน**บนเรือไดม่อนพริ้นเซสพร้อมพล็อตกราฟ แล้วระบุวันที่มีการติดเชื้อมากที่สุด
5. จงหา Country/Region ที่ล่าสุดมีเปอร์เซ็นต์อัตราการเสียชีวิตมากที่สุด 20 อันดับ พร้อมระบุว่า จาก Country/Region ในกลุ่มดังกล่าว ส่วนใหญ่อยู่ในทวีปอะไร
6. จงหายอดผู้ป่วยที่กำลังรักษาตัวล่าสุด (สมมติว่ายังไม่เสียชีวิตหรือหายดี) ของแต่ละ Province/State ในประเทศจีน พร้อมพล็อตกราฟแสดงจำนวนในแต่ละ Province/State

1. จงแสดงอัตราการรักษาหายของเฉพาะผู้ป่วยในสหภาพยุโรปเป็นเปอร์เซ็นต์

```
#Solution
from google.colab import files
files.upload()
```

```
Choose Files time_series..._global.csv
• time_series_covid19_confirmed_global.csv(text/csv) - 67420 bytes, last modified: 4/13/2020 -
100% done
Saving time_series_covid19_confirmed_global.csv to time_series_covid19_confirmed_global.csv
{'time_series_covid19_confirmed_global.csv': b'Province/State,Country/Region,Lat,Long,1/22/20,1/23/20,1/24/20,1/25/20,1/26/20,
```

```
files.upload()
```

```
Choose Files time_series..._global.csv
• time_series_covid19_deaths_global.csv(text/csv) - 54349 bytes, last modified: 4/13/2020 -
100% done
Saving time_series_covid19_deaths_global.csv to time_series_covid19_deaths_global.csv
{'time_series_covid19_deaths_global.csv': b'Province/State,Country/Region,Lat,Long,1/22/20,1/23/20,1/24/20,1/25/20,1/26/20,1/27
```

```
files.upload()
```

```
Choose Files time_series..._global.csv
• time_series_covid19_recovered_global.csv(text/csv) - 56755 bytes, last modified: 4/13/2020 -
100% done
Saving time_series_covid19_recovered_global.csv to time_series_covid19_recovered_global.csv
{'time_series_covid19_recovered_global.csv': b'Province/State,Country/Region,Lat,Long,1/22/20,1/23/20,1/24/20,1/25/20,1/26/20,
```

```
pip install pycountry_convert
```

```
Requirement already satisfied: pycountry_convert in /usr/local/lib/python3.6/dist-packages (0.7.2)
Requirement already satisfied: pprintpp>=0.3.0 in /usr/local/lib/python3.6/dist-packages (from pycountry_convert) (0.4.0)
Requirement already satisfied: pycountry>=16.11.27.1 in /usr/local/lib/python3.6/dist-packages (from pycountry_convert) (19.8.1
Requirement already satisfied: pytest>=3.4.0 in /usr/local/lib/python3.6/dist-packages (from pycountry_convert) (3.6.4)
Requirement already satisfied: pytest-cov>=2.5.1 in /usr/local/lib/python3.6/dist-packages (from pycountry_convert) (2.10.0)
Requirement already satisfied: repoze.lru>=0.7 in /usr/local/lib/python3.6/dist-packages (from pycountry_convert) (0.7)
```

```
import pandas as pd
import pycountry_convert as pc
cf = pd.read_csv('time_series_covid19_confirmed_global.csv', delimiter = ',')
dt = pd.read_csv('time_series_covid19_deaths_global.csv', delimiter = ',')
rc = pd.read_csv('time_series_covid19_recovered_global.csv', delimiter = ',')
```

```
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from pytest>=3.4.0->pycountry_convert) (4
```

```
cf['Country/Region'] = cf['Country/Region'].replace(
                            ['Congo (Brazzaville)', 'Congo (Kinshasa)', "Cote d'Ivoire", 'Burma', 'Korea, South', 'Taiwan*', 'US'
                            ['Congo', 'Congo', 'Ivory Coast', 'Myanmar', 'South Korea', 'Taiwan', 'United States']
                            )

rc['Country/Region'] = rc['Country/Region'].replace(
                            ['Congo (Brazzaville)', 'Congo (Kinshasa)', "Cote d'Ivoire", 'Burma', 'Korea, South', 'Taiwan*', 'US'
                            ['Congo', 'Congo', 'Ivory Coast', 'Myanmar', 'South Korea', 'Taiwan', 'United States']
                            )

dt['Country/Region'] = dt['Country/Region'].replace(
                            ['Congo (Brazzaville)', 'Congo (Kinshasa)', "Cote d'Ivoire", 'Burma', 'Korea, South', 'Taiwan*', 'US'
                            ['Congo', 'Congo', 'Ivory Coast', 'Myanmar', 'South Korea', 'Taiwan', 'United States']
                            )
```

```
country_list = cf['Country/Region'].unique()
len(country_list)
```

```
184
```

```
convert = {}

for country in country_list:
    try :
        convert[country] = pc.convert_continent_code_to_continent_name(pc.country_alpha2_to_continent_code(pc.country_name_to_cou
    except :
        convert[country] = "Other"

len(convert)
```

```
184
```

```
cf['Continents'] = cf['Country/Region'].map(convert)
dt['Continents'] = dt['Country/Region'].map(convert)
rc['Continents'] = rc['Country/Region'].map(convert)
```

```
rc_result = rc.drop(axis=1, columns=['Province/State', 'Lat', 'Long'])
rc_max = rc_result.groupby(['Country/Region']).agg('sum')
rc_max = rc_max['4/12/20'].to_frame('Max_Recover')
cf_result = cf.drop(axis=1, columns=['Province/State', 'Lat', 'Long'])
cf_max = cf_result.groupby(['Country/Region']).agg('sum')
cf_max = cf_max['4/12/20'].to_frame('Max_Confirmed')
dt_result = dt.drop(axis=1, columns=['Province/State', 'Lat', 'Long'])
dt_max = dt_result.groupby(['Country/Region']).agg('sum')
dt_max = dt_max['4/12/20'].to_frame('Max_Death')
```

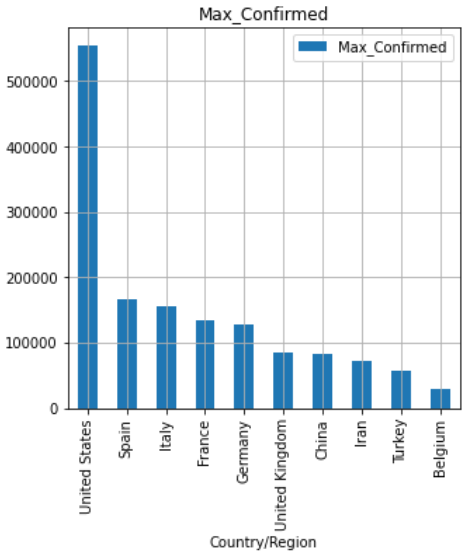
```
eu_list = ['Austria', 'Belgium', 'Bulgaria', 'Croatia', 'Cyprus', 'Czechia', 'Denmark', 'Estonia', 'Finland', 'France', 'Germany'
x = rc_max.reset_index()
rc_eu = x[x['Country/Region'].isin(eu_list)]
y = cf_max.reset_index()
cf_eu = y[y['Country/Region'].isin(eu_list)]

eu = pd.merge(rc_eu, cf_eu, how='inner', left_on='Country/Region', right_on='Country/Region')
eu['Union'] ='European Union'
eu = eu.groupby(['Union']).agg('sum')
eu['Percent_recovered'] = eu['Max_Recover'] / eu['Max_Confirmed'] * 100

print(eu['Percent_recovered'])
print ("\nอัตรการรักษาหายของผู้ป่วยในสหภาพยุโรปเท่ากับ " + " " + str(eu.loc['European Union', 'Percent_recovered']) + "%")
```

```
Union
European Union    28.003272
Name: Percent_recovered, dtype: float64

อัตรการรักษาหายของผู้ป่วยในสหภาพยุโรปเท่ากับ  28.0032721114162168%
```

2. จงหา Country/Region ที่มียอดจำนวนผู้ติดเชื้อล่าสุดมากที่สุด 10 อันดับพร้อมพล็อตกราฟแสดงจำนวนผู้ติดเชื้อของแต่ละ Country/Region โดยเรียงลำดับจากมากไปน้อย (ไม่ต้องแยกเป็น Province/State)

```
#Solution
top10_cf = cf_result.groupby(['Country/Region']).agg('sum')
top10_cf = top10_cf.nlargest(10, '4/12/20')
top10_cf = top10_cf['4/12/20'].to_frame('Max_Confirmed')

print('Country/Region ที่มียอดจำนวนผู้ติดเชื้อล่าสุดมากที่สุด 10 อันดับ\n')
print(top10_cf)
print("\n")

top10_cf.plot(kind='bar', subplots=True, figsize=(5, 5), legend=True, grid=True)
```

```
Country/Region ที่มียอดจำนวนผู้ติดเชื้อล่าสุดมากที่สุด 10 อันดับ

                Max_Confirmed
Country/Region
United States          555313
Spain                  166831
Italy                  156363
France                 133670
Germany                127854
United Kingdom          85206
China                   83134
Iran                    71686
Turkey                  56956
Belgium                 29647


array([<matplotlib.axes._subplots.AxesSubplot object at 0x7fb79dac2780>],
      dtype=object)
```

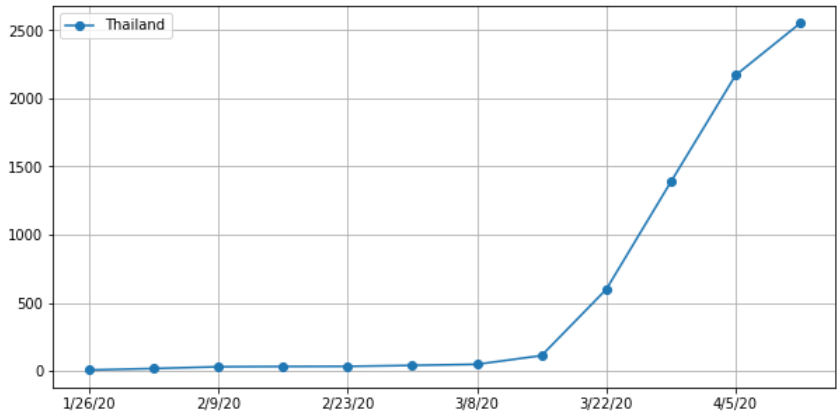


3. จงพล็อตกราฟเส้นเพื่อแสดง trend ยอดผู้ติดเชื้อในประเทศไทยในแต่ละสัปดาห์(กำหนดให้นับข้อมูลทุกวันอาทิตย์)

```
#Solution
thai = cf_result.groupby('Country/Region').agg('sum')
thai = thai.T['Thailand'].to_frame('Thailand')
thai = thai.reset_index()
datetime = pd.to_datetime(thai['index'])
dayname = datetime.dt.day_name()
thai['Day_Name'] = dayname

#print(thai)
#print("\n")
thai_sunday = thai[thai['Day_Name'] == 'Sunday']

#print(thai_sunday)
#print("\n")
print('กราฟเส้นแสดง trend ยอดผู้ติดเชื้อในประเทศไทยแต่ละสัปดาห์ (ทุกวันอาทิตย์)\n')

line_chart = thai_sunday.drop(axis=1, columns=['Day_Name'])
line_chart.plot.line(x='index', y='Thailand', figsize=(10, 5), legend=True, grid=True, linestyle='-', marker='o')
```

กราฟเส้นแสดง trend ยอดผู้ติดเชื้อในประเทศไทยแต่ละสัปดาห์ (ทุกวันอาทิตย์)

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb79db2c748>
```



4. จงหายอดผู้ติดเชื้อของแต่ละวันบนเรือไดม่อนพริ้นเซสพร้อมพล็อตกราฟ แล้วระบุวันที่มีการติดเชื้อมากที่สุด

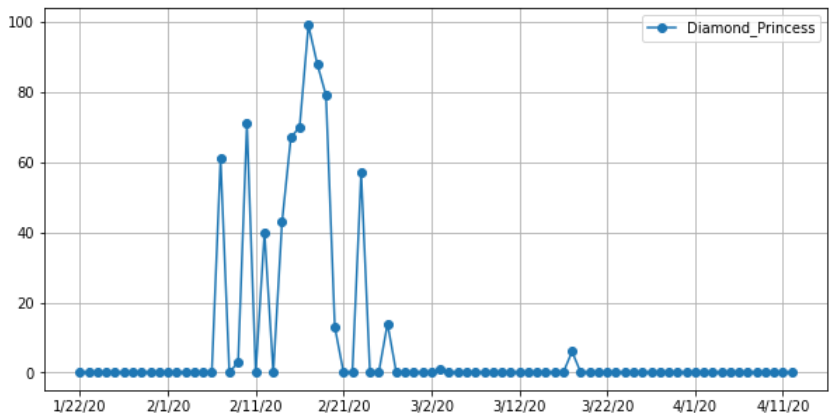
```
#Solution
diamond = cf_result.groupby('Country/Region').agg('sum')
diamond = diamond.T['Diamond Princess'].to_frame('Diamond_Princess')

#print(diamond)
#print('\n')


plot_diamond = diamond.diff()
plot_diamond['Diamond_Princess'] = plot_diamond['Diamond_Princess'].replace([None], [0])

print('ยอดผู้ติดเชื้อของแต่ละวันบนเรือไดม่อนพริ้นเซส\n')
print(plot_diamond)
print('\n')
print('กราฟแสดงยอดผู้ติดเชื้อของแต่ละวันบนเรือไดม่อนพริ้นเซส\n')

plot_diamond.plot(kind='line', figsize=(10, 5), legend=True, grid=True, linestyle='-', marker='o')
```

ยอดผู้ติดเชื้อของแต่ละวันบนเรือไดม่อนพริ้นเซส

```
         Diamond_Princess
1/22/20               0.0
1/23/20               0.0
1/24/20               0.0
1/25/20               0.0
1/26/20               0.0
...                   ...
4/8/20                0.0
4/9/20                0.0
4/10/20               0.0
4/11/20               0.0
4/12/20               0.0

[82 rows x 1 columns]
```

กราฟแสดงยอดผู้ติดเชื้อของแต่ละวันบนเรือไดม่อนพริ้นเซส

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb79d88ac88>
```




```
plot_diamond = plot_diamond.idxmax(axis=0, skipna=True)
print('วันที่มีการติดเชื้อมากที่สุด คือ วันที่ ' + plot_diamond)
```

```
Diamond_Princess    วันที่มีการติดเชื้อมากที่สุด คือ วันที่ 2/17/20
dtype: object
```

5. จงหา Country/Region ที่ล่าสุดมีเปอร์เซ็นต์อัตราการเสียชีวิตมากที่สุด 20 อันดับ พร้อมระบุว่า จาก Country/Region ในกลุ่มดังกล่าว ส่วนใหญ่อยู่ในทวีปอะไร

```
#Solution
max_dt = dt_result.groupby(['Country/Region']).agg('sum')
max_cf = cf_result.groupby(['Country/Region']).agg('sum')

mdt = max_dt['4/12/20'].to_frame('Max_Death')
mcf = max_cf['4/12/20'].to_frame('Max_Confirmed')

mdt['Max_Confirmed'] = mcf['Max_Confirmed']

#print(mdt)
#print('\n')
print('Country/Region ที่มีอัตราเสียชีวิตมากที่สุด 20 อันดับ\n')

mdt['Percent_Death'] = mdt['Max_Death'] / mdt['Max_Confirmed'] * 100

mdt = mdt.reset_index()
mdt['Continents'] = mdt['Country/Region'].map(convert)
mdt

top20_dt = mdt.nlargest(20, 'Percent_Death')
print(top20_dt[['Country/Region', 'Continents', 'Percent_Death']])
```

```
Country/Region ที่มีอัตราเสียชีวิตมากที่สุด 20 อันดับ

     Country/Region     Continents  Percent_Death
99       MS Zaandam          Other      22.222222
183        Zimbabwe         Africa      21.428571
11          Bahamas  North America      17.391304
101          Malawi         Africa      15.384615
2           Algeria         Africa      15.308255
17           Belize  North America      14.285714
106      Mauritania         Africa      14.285714
69           Guyana  South America      13.333333
81            Italy         Europe      12.726156
28       Cabo Verde         Africa      12.500000
173  United Kingdom         Europe      12.474474
16          Belgium         Europe      12.142881
60           Gambia         Africa      11.111111
120       Nicaragua  North America      11.111111
58           France         Europe      10.781776
118     Netherlands         Europe      10.669619
4            Angola         Africa      10.526316
157           Sudan         Africa      10.526316
155           Spain         Europe      10.315229
94          Liberia         Africa      10.000000
```

```
count_continents = top20_dt.pivot_table(index=['Continents'], aggfunc='size')
count_continents = count_continents.idxmax(axis=0, skipna=True)
print('ทวีปที่มีอัตราการเสียชีวิตมากที่สุด คือ ' + count_continents)
```

```
ทวีปที่มีอัตราการเสียชีวิตมากที่สุด คือ Africa
```

6. จงหายอดผู้ป่วยที่กำลังรักษาตัวล่าสุด (สมมติว่ายังไม่เสียชีวิตหรือหายดี) ของแต่ละ Province/State ในประเทศจีน พร้อมพล็อตกราฟแสดงจำนวนในแต่ละ Province/State

```
#Solution
chinarc = rc.groupby(['Country/Region', 'Province/State'])[['Country/Region', 'Province/State', '4/12/20']].agg('sum')
chinarc = chinarc.reset_index()
chinarc = chinarc[(chinarc['Country/Region'] == 'China')][["Province/State", 'Country/Region', '4/12/20']]


chinacf = cf.groupby(['Country/Region', 'Province/State'])[['Country/Region', 'Province/State', '4/12/20']].agg('sum')
chinacf = chinacf.reset_index()
chinacf = chinacf[(chinacf['Country/Region'] == 'China')][["Province/State", 'Country/Region', '4/12/20']]


chinadt = dt.groupby(['Country/Region', 'Province/State'])[['Country/Region', 'Province/State', '4/12/20']].agg('sum')
chinadt = chinadt.reset_index()
chinadt = chinadt[(chinadt['Country/Region'] == 'China')][["Province/State", 'Country/Region', '4/12/20']]


#chinarc.shape
#chinacf.shape
#chinadt.shape

```

```
chinarc = chinarc.drop(axis=1, columns=['Country/Region'])
chinadt = chinadt.drop(axis=1, columns=['Country/Region'])

china = pd.merge(chinacf, chinarc, how='inner', left_on='Province/State', right_on='Province/State')
china = pd.merge(china, chinadt, how='inner', left_on='Province/State', right_on='Province/State')
china.rename(columns={'4/12/20_x' : 'Confirmed','4/12/20_y' : 'Recovered', '4/12/20' : 'Death'}, inplace=True)
china['In_Process'] = china['Confirmed'] - china['Recovered'] - china['Death']
china_inprocess = china[['Province/State', 'In_Process']]

print('ยอดผู้ป่วยที่กำลังรักษาตัวล่าสุด ของแต่ละ Province/State ในประเทศจีน\n')
print(china_inprocess)
print('\n')

china_inprocess.plot.bar(x='Province/State', y='In_Process', figsize=(15, 5), legend=True, grid=True)
```

ยอดผู้ป่วยที่กำลังรักษาตัวล่าสุด ของแต่ละ Province/State ในประเทศจีน

	Province/State	In_Process
0	Anhui	1
1	Beijing	102
2	Chongqing	3
3	Fujian	30
4	Gansu	2
5	Guangdong	101
6	Guangxi	0
7	Guizhou	0
8	Hainan	0
9	Hebei	8
10	Heilongjiang	201
11	Henan	1
12	Hong Kong	640
13	Hubei	303
14	Hunan	1
15	Inner Mongolia	105
16	Jiangsu	15
17	Jiangxi	0
18	Jilin	2
19	Liaoning	7
20	Macau	32
21	Ningxia	0
22	Qinghai	0
23	Shaanxi	5
24	Shandong	20
25	Shanghai	154
26	Shanxi	37
27	Sichuan	10
28	Tianjin	19
29	Tibet	0
30	Xinjiang	0
31	Yunnan	8
32	Zhejiang	28

<matplotlib.axes._subplots.AxesSubplot at 0x7fb79e6d7a58>