Forward School

Program Code: J620-002-4:2020

Program Name: FRONT-END SOFTWARE DEVELOPMENT

Title : Covid 19 Project

Name: Phua Yan Han

IC Number: 050824070059

Date : 28/6/23

Introduction : sorting and filtering data frame

Conclusion : learn different functions for sorting and filtering data frame while plotting graph

Covid 19 Python Project (use all your knowledge thus far to solve this)

From Wikipedia,

Coronavirus disease 2019 (COVID-19) is an infectious disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). The disease was first identified in 2019 in Wuhan, China, and has since spread globally, resulting in the 2019–20 coronavirus pandemic. Common symptoms include fever, cough and shortness of breath. Muscle pain, sputum production and sore throat are less common. The rate of deaths per number of diagnosed cases is on average 3.4%, ranging from 0.2% in those less than 20 to approximately 15% in those over 80 years old.

Data Source (Date wise) : 2019 Novel Coronavirus COVID-19 (2019-nCoV) Data Repository by Johns Hopkins CSSE

Data Source: https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_daily_reports (https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_daily_reports)

File naming convention

MM-DD-YYYY.csv in UTC.

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). Last Update: MM/DD/YYYY HH:mm (24 hour format, in UTC). Confirmed: the number of confirmed cases. For Hubei Province: from Feb 13 (GMT +8), we report both clinically diagnosed and lab-confirmed cases. For lab-confirmed cases only (Before Feb 17), please refer to who_covid_19_situation_reports. For Italy, diagnosis standard might be changed since Feb 27 to "slow the growth of new case numbers." (Source) Deaths: the number of deaths. Recovered: the number of recovered cases. Update frequency: Files after Feb 1 (UTC): once a day around

In [48]:
```
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
```

Q1. Write Python code to display first 5 rows from COVID-19 dataset. Also print the dataset information and check the missing values.

In [12]:
```
df = pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/mas
df.head()
```

Out[12]:

	FIPS	Admin2	Province_State	Country_Region	Last_Update	Lat	Long_	Confirmed
0	NaN	NaN	NaN	Afghanistan	2020-10-11 04:23:46	33.93911	67.709953	39789
1	NaN	NaN	NaN	Albania	2020-10-11 04:23:46	41.15330	20.168300	15231
2	NaN	NaN	NaN	Algeria	2020-10-11 04:23:46	28.03390	1.659600	52940
3	NaN	NaN	NaN	Andorra	2020-10-11 04:23:46	42.50630	1.521800	2696
4	NaN	NaN	NaN	Angola	2020-10-11 04:23:46	-11.20270	17.873900	6246

In [49]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3992 entries, 0 to 3991
Data columns (total 14 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   FIPS                 3262 non-null   float64
 1   Admin2               3267 non-null   object 
 2   Province_State       3816 non-null   object 
 3   Country_Region       3992 non-null   object 
 4   Last_Update          3992 non-null   object 
 5   Lat                  3907 non-null   float64
 6   Long_                3907 non-null   float64
 7   Confirmed            3992 non-null   int64  
 8   Deaths               3992 non-null   int64  
 9   Recovered            3992 non-null   int64  
 10  Active               3992 non-null   int64  
 11  Combined_Key         3992 non-null   object 
 12  Incidence_Rate       3907 non-null   float64
 13  Case-Fatality_Ratio  3937 non-null   float64
dtypes: float64(5), int64(4), object(5)
memory usage: 436.8+ KB
```

In [50]:
```
df.isnull().sum()
```

Out[50]:
```
FIPS                   730
Admin2                 725
Province_State         176
Country_Region           0
Last_Update              0
Lat                     85
Long_                   85
Confirmed                0
Deaths                   0
Recovered                0
Active                   0
Combined_Key             0
Incidence_Rate          85
Case-Fatality_Ratio     55
dtype: int64
```

Q2. Write a Python program to get the latest number of confirmed, deaths, recovered and active cases of Novel Coronavirus (COVID-19) Country wise

In [36]:
```
df.groupby(["Country_Region"])[["Confirmed","Deaths","Recovered","Active"]].sum
```

Out[36]:

Country_Region	Confirmed	Deaths	Recovered	Active
Afghanistan	39789	1477	33064	5248
Albania	15231	416	9406	5409
Algeria	52940	1795	37170	13975
Andorra	2696	55	1814	827
Angola	6246	218	2716	3312
...	...	...	...	...
West Bank and Gaza	43945	378	37240	6327
Winter Olympics 2022	0	0	0	0
Yemen	2051	595	1329	127
Zambia	15415	337	14541	537
Zimbabwe	8010	230	6492	1288

197 rows × 4 columns

Q3. Write a Python program to get the Chinese province wise cases of confirmed, deaths and recovered cases of Novel Coronavirus (COVID-19)

In [52]:
```
df[df['Country_Region'] == 'China'].groupby('Province_State')[['Confirmed', 'De
```

Out[52]:

Province_State	Confirmed	Deaths	Recovered
Anhui	991	6	985
Beijing	936	9	927
Chongqing	585	6	578
Fujian	415	1	400
Gansu	170	2	168
Guangdong	1858	8	1823
Guangxi	260	2	256
Guizhou	147	2	145
Hainan	171	6	165
Hebei	365	6	358
Heilongjiang	948	13	935
Henan	1281	22	1255
Hong Kong	5175	105	4914
Hubei	68139	4512	63627
Hunan	1019	4	1015
Inner Mongolia	268	1	261
Jiangsu	667	0	664
Jiangxi	935	1	934
Jilin	157	2	155
Liaoning	276	2	269
Macau	46	0	46
Ningxia	75	0	75
Qinghai	18	0	18
Shaanxi	428	3	397
Shandong	832	7	824
Shanghai	1048	7	980
Shanxi	206	0	203
Sichuan	721	3	673
Tianjin	244	3	236
Tibet	1	0	1
Unknown	5201	0	0
Xinjiang	902	3	899
Yunnan	211	2	200
Zhejiang	1283	1	1272

Q4. Write a Python program to get the latest country wise deaths cases of Novel Coronavirus (COVID-19)

In [55]:
```
df.groupby(["Country_Region"])["Deaths"].sum().reset_index()
```

Out[55]:

	Country_Region	Deaths
0	Afghanistan	1477
1	Albania	416
2	Algeria	1795
3	Andorra	55
4	Angola	218
...	...	...
192	West Bank and Gaza	378
193	Winter Olympics 2022	0
194	Yemen	595
195	Zambia	337
196	Zimbabwe	230

197 rows × 2 columns

Q5. Write a Python program to list countries with no cases of Novel Coronavirus (COVID-19) recovered

In [224]:
```
no_recovered = df.groupby('Country_Region')['Recovered'].sum()
no_recovered[no_recovered == 0].reset_index()
```

Out[224]:

	Country_Region	Recovered
0	Antarctica	0
1	Kiribati	0
2	Korea, North	0
3	MS Zaandam	0
4	Nauru	0
5	Palau	0
6	Samoa	0
7	Serbia	0
8	Summer Olympics 2020	0
9	Sweden	0
10	Tonga	0
11	Tuvalu	0
12	Winter Olympics 2022	0

Q6. Write a Python program to get the latest number of confirmed deaths and recovered people of Novel Coronavirus (COVID-19) cases Country/Region - Province/State wise.

In [43]:
```
df.groupby(["Country_Region","Province_State"])[["Confirmed","Deaths","Recovere
```

Out[43]:

		Confirmed	Deaths	Recovered
Country_Region	**Province_State**			
Australia	**Australian Capital Territory**	113	3	110
	New South Wales	4278	53	0
	Northern Territory	33	0	33
	Queensland	1161	6	1152
	South Australia	473	4	466
...	**...**	...	...	...
United Kingdom	**Saint Helena, Ascension and Tristan da Cunha**	2	0	2
	Scotland	38042	4339	0
	Turks and Caicos Islands	696	6	672
	Unknown	0	74	0
	Wales	29654	2700	0

601 rows × 3 columns

Q7. Write a Python program to list countries with all cases of Novel Coronavirus (COVID-19) died

In [222]:
```
d_cases = df.groupby(['Country_Region'])['Confirmed', 'Deaths', 'Recovered', 'A
d_cases = d_cases[d_cases['Deaths'] == d_cases['Confirmed']]
d_cases = d_cases[d_cases['Confirmed'] > 0]
d_cases = d_cases[d_cases['Active']==0]
d_cases = d_cases[d_cases['Recovered']==0]
print(d_cases)
```

```
Empty DataFrame
Columns: [Country_Region, Confirmed, Deaths, Recovered, Active]
Index: []

C:\Users\Asus\AppData\Local\Temp\ipykernel_21376\1491657567.py:1: FutureWarni
ng: Indexing with multiple keys (implicitly converted to a tuple of keys) wil
l be deprecated, use a list instead.
  d_cases = df.groupby(['Country_Region'])['Confirmed', 'Deaths', 'Recovere
d', 'Active'].sum().reset_index()
```

Q8.Write a Python program to list countries with all cases of Novel Coronavirus (COVID-19) recovered.

In [223]:
```
d_cases = df.groupby(['Country_Region'])['Confirmed', 'Deaths', 'Recovered',"Ac
d_cases = d_cases[d_cases['Recovered'] == d_cases['Confirmed']]
d_cases = d_cases[d_cases['Confirmed'] > 0]
d_cases = d_cases[d_cases['Active']==0]
d_cases = d_cases[d_cases['Deaths']==0]
print(d_cases)
```

```
                      Country_Region  Confirmed  Deaths  Recovered  Active
70                           Grenada         24       0         24       0
76                          Holy See         12       0         12       0
149  Saint Vincent and the Grenadines         64       0         64       0
176                     Timor-Leste         28       0         28       0

C:\Users\Asus\AppData\Local\Temp\ipykernel_21376\2304194200.py:1: FutureWarni
ng: Indexing with multiple keys (implicitly converted to a tuple of keys) wil
l be deprecated, use a list instead.
  d_cases = df.groupby(['Country_Region'])['Confirmed', 'Deaths', 'Recovere
d',"Active"].sum().reset_index()
```

Q9. Write a Python program to get the top 10 countries data (Last Update, Country/Region, Confirmed, Deaths, Recovered) of Novel Coronavirus (COVID-19).

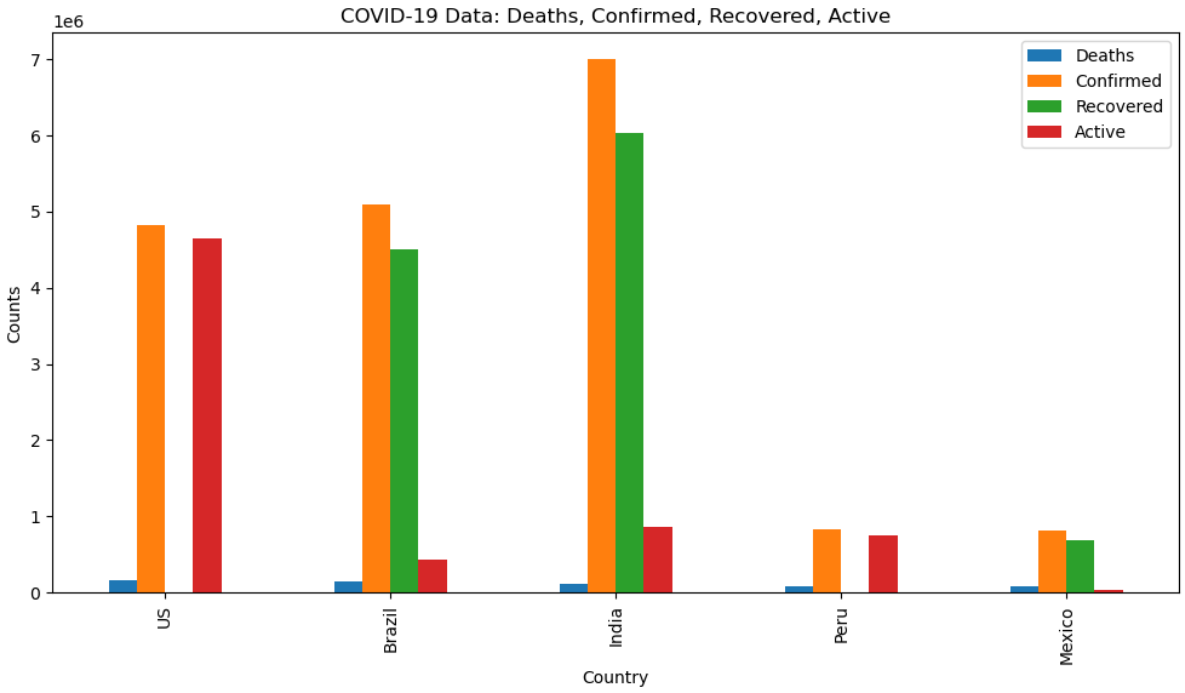
In [172]:
```
df[['Last_Update', 'Country_Region', 'Confirmed', 'Deaths', 'Recovered']].nlarg
```

Out[172]:

	Last_Update	Country_Region	Confirmed	Deaths	Recovered
255	2020-10-11 04:23:46	India	1517434	40040	1255779
54	2020-10-11 04:23:46	Brazil	1034816	37223	898416
6	2020-10-11 04:23:46	Argentina	883882	23581	709464
236	2020-10-11 04:23:46	India	750517	6194	697699
202	2020-10-11 04:23:46	France	731155	32439	81739
250	2020-10-11 04:23:46	India	700786	9891	569947
576	2020-10-11 04:23:46	South Africa	690896	17673	622153
265	2020-10-11 04:23:46	India	651370	10187	597033
3938	2020-10-11 04:23:46	United Kingdom	504056	51595	0
273	2020-10-11 04:23:46	Iran	496253	28293	403950

Q10. Write a Python program to create a plot (lines) of total deaths, confirmed, recovered and active cases Country wise where deaths greater than 150.

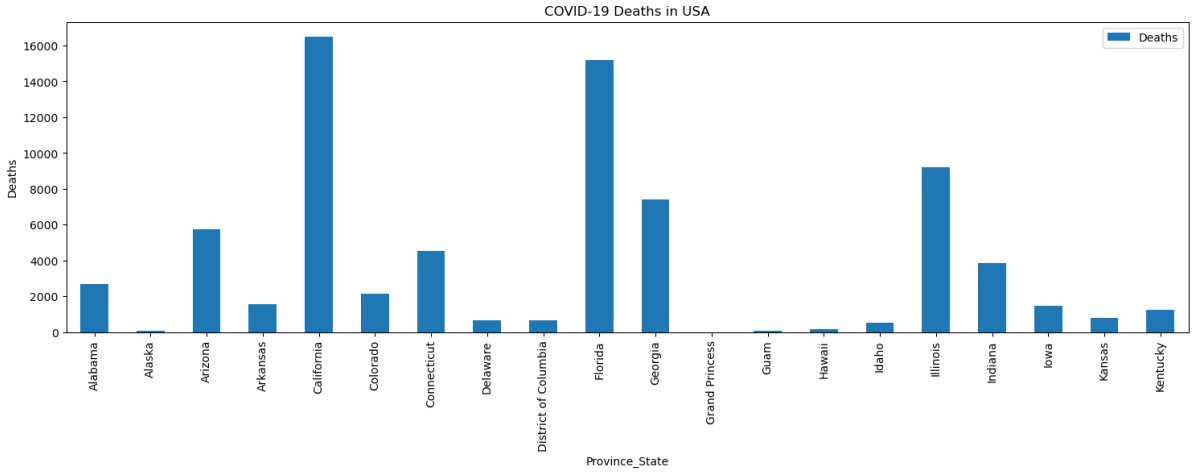
In [178]:
```
filteredData = df[df['Deaths'] > 150].groupby('Country_Region')[['Deaths', 'Cor
fig, ax = plt.subplots(figsize=(12, 6))

# Plot the data as a line graph
filteredData.nlargest(5,['Deaths','Confirmed','Recovered','Active']).plot(kind=

# Set the labels and title
ax.set_xlabel('Country')
ax.set_ylabel('Counts')
ax.set_title('COVID-19 Data: Deaths, Confirmed, Recovered, Active')

# Rotate the x-axis labels for better readability if needed
plt.xticks(rotation=90)

# Show the plot
plt.show()
```



Q.11 Write a Python program to visualize the state/province wise death cases of Novel Coronavirus (COVID-19) in USA

In [196]:
```
# temp=df.groupby(['Country_Region','Province_State'])['Deaths'].sum().reset_ir
# temp=temp[(temp['Country_Region'] =="US") & df['Deaths']>0]
# ax=temp.head(20).plot(x='Province_State',y=['Deaths'],kind='bar', figsize=(18
# plt.set_xlabel("Province_State")
# plt.set_ylabel("Deaths")
# plt.set_title("COVID-19 Deaths in USA")
new_cases=df.groupby(['Country_Region','Province_State'])['Deaths'].sum().reset
us_cases = new_cases[(new_cases['Country_Region']== 'US') & (new_cases['Deaths'
ax = us_cases.head(20).plot(x='Province_State',y=['Deaths'],kind='bar',figsize=
plt.xlabel("Province_State")
plt.ylabel("Deaths")
plt.title("COVID-19 Deaths in USA")
```

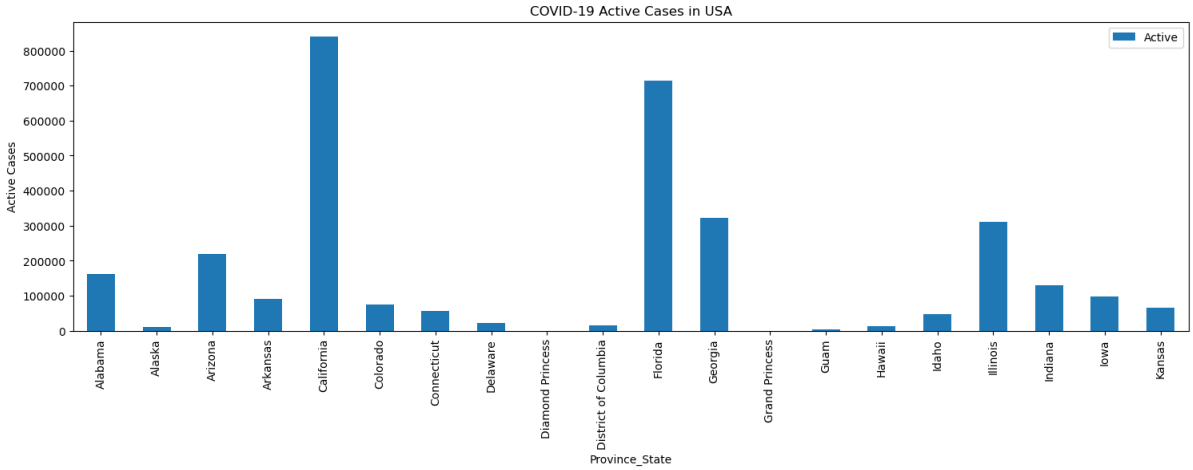
Out[196]: Text(0.5, 1.0, 'COVID-19 Deaths in USA')



Q.12 Write a Python program to visualize the state/province wise Active cases of Novel Coronavirus (COVID-19) in USA

In [199]:
```
# fig, ax = plt.subplots(figsize=(12,5))
# df[df['Country_Region'] =="US"].groupby(['Province_State'])['Active'].sum().h
# ax.set_xlabel("Province_State")
# ax.set_ylabel("Active")
# ax.set_title("COVID-19 Active Case in USA")
new_cases=df.groupby(['Country_Region','Province_State'])['Active'].sum().reset
us_cases = new_cases[(new_cases['Country_Region']== 'US') & (new_cases['Active'
ax = us_cases.head(20).plot(x='Province_State',y=['Active'],kind='bar',figsize=
plt.xlabel("Province_State")
plt.ylabel("Active Cases")
plt.title("COVID-19 Active Cases in USA")
```

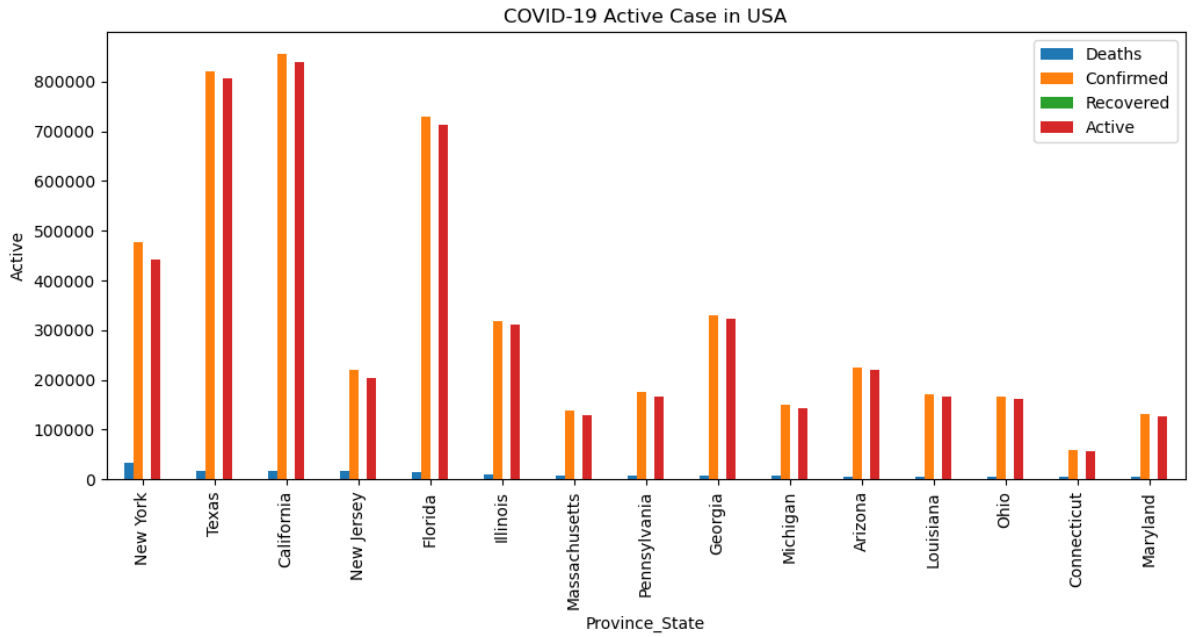
Out[199]:
```
Text(0.5, 1.0, 'COVID-19 Active Cases in USA')
```



Q.13 Write a Python program to visualize the state/province wise combine number of confirmed, deaths, recovered, active Novel Coronavirus (COVID-19) cases in USA.

In [201]:
```
fig, ax = plt.subplots(figsize=(12,5))
temp=df[df['Country_Region'] =="US"].groupby('Province_State')[['Deaths', 'Conf
temp.sort_values('Deaths',ascending=False).head(15).plot(kind='bar', ax=ax)
ax.set_xlabel("Province_State")
ax.set_ylabel("Active")
ax.set_title("COVID-19 Active Case in USA")
```

Out[201]: Text(0.5, 1.0, 'COVID-19 Active Case in USA')



Q.14 Write a Python program to visualize Worldwide Confirmed Novel Coronavirus (COVID-19) cases over time

In [ ]:

In [ ]: