In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
# Task 2:
# Task 2.1: importing covid19 dataset

corona_dataset_csv = pd.read_csv("Covid project/Datasets/covid19_Confirmed_dataset.csv")
corona_dataset_csv.head(10)
```

Out[2]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	0
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	0
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	0
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	0
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	3

10 rows × 104 columns

In [3]:

```
corona_dataset_csv.shape
```

Out[3]:

```
(266, 104)
```

In [4]:

```
df = corona_dataset_csv.drop(["Lat","Long"],axis=1)
df.head()
```

Out[4]:

	Province/State	Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20
0	NaN	Afghanistan	0	0	0	0	0	0	0
1	NaN	Albania	0	0	0	0	0	0	0
2	NaN	Algeria	0	0	0	0	0	0	0
3	NaN	Andorra	0	0	0	0	0	0	0
4	NaN	Angola	0	0	0	0	0	0	0

5 rows × 102 columns

In [5]:

```
# Task 2.3: Aggregating the rows by the country

corona_dataset_aggregated = df.groupby("Country/Region").sum()
corona_dataset_aggregated.head()
```

Out[5]:

Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29/20	1/30/20
Afghanistan	0	0	0	0	0	0	0	0	0
Albania	0	0	0	0	0	0	0	0	0
Algeria	0	0	0	0	0	0	0	0	0
Andorra	0	0	0	0	0	0	0	0	0
Angola	0	0	0	0	0	0	0	0	0

5 rows × 100 columns

In [6]:

```
corona_dataset_aggregated.shape
```

Out[6]:

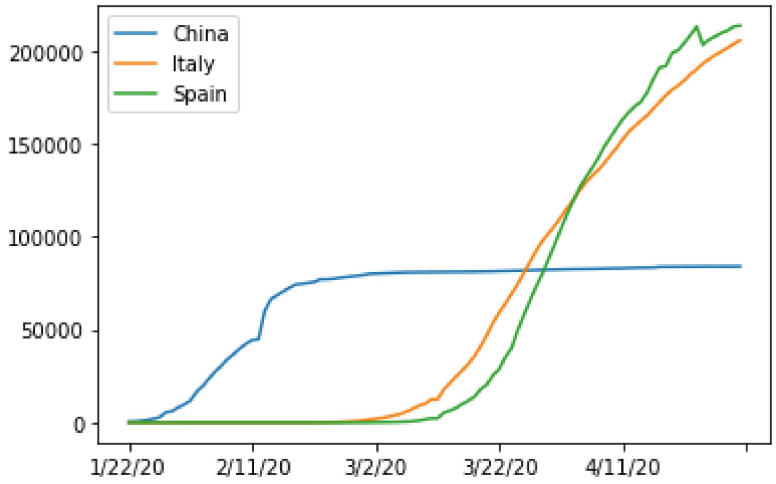
```
(187, 100)
```

In [7]:

```
# task 2.4: Visualizing data related to a country for example china

corona_dataset_aggregated.loc['China'].plot()
corona_dataset_aggregated.loc['Italy'].plot()
corona_dataset_aggregated.loc['Spain'].plot()
plt.legend()
```

Out[7]:

```
<matplotlib.legend.Legend at 0x1f33db353d0>
```

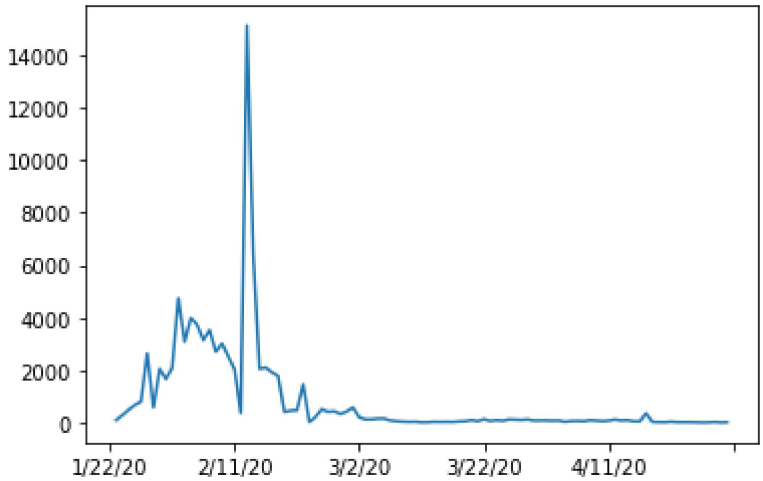


In [8]:

```
# calculating and plotting the first derivative of the curve

corona_dataset_aggregated.loc['China'].diff().plot()
```

Out[8]:

```
<AxesSubplot:>
```



In [9]:

```
# find maximum inflection rate for China, Italy and Spain

corona_dataset_aggregated.loc['China'].diff().max()
```

Out[9]:

```
15136.0
```

In [10]:

```
corona_dataset_aggregated.loc['Italy'].diff().max()
```

Out[10]:

```
6557.0
```

In [11]:

```
# find maximum infection rate for all the of the countries
countries = list(corona_dataset_aggregated.index)
max_infection_rate = []
for c in countries:
    max_infection_rate.append(corona_dataset_aggregated.loc[c].diff().max())
corona_dataset_aggregated['max_infection_rate'] = max_infection_rate
```

In [12]:

```
corona_dataset_aggregated.head()
```

Out[12]:

Country/Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29/20	1/30/20	1/
Afghanistan	0	0	0	0	0	0	0	0	0	
Albania	0	0	0	0	0	0	0	0	0	
Algeria	0	0	0	0	0	0	0	0	0	
Andorra	0	0	0	0	0	0	0	0	0	
Angola	0	0	0	0	0	0	0	0	0	

5 rows × 101 columns

In [13]:

```
# create a new dataframe with only needed column

corona_data = pd.DataFrame(corona_dataset_aggregated['max_infection_rate'])
```

In [14]:

```
corona_data.head()
```

Out[14]:

Country/Region	max_infection_rate
Afghanistan	232.0
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0

In [15]:

```
# importing the dataset
happiness_report = pd.read_csv('Covid project/Datasets/worldwide_happiness_report.csv')
```

In [16]:

```
happiness_report.head()
```

Out[16]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298

In [17]:

```
useless_cols = ['Overall rank','Score','Generosity','Perceptions of corruption']
happiness_report.drop(useless_cols,axis=1,inplace=True)
happiness_report.head()
```

Out[17]:

	Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
0	Finland	1.340	1.587	0.986	0.596
1	Denmark	1.383	1.573	0.996	0.592
2	Norway	1.488	1.582	1.028	0.603
3	Iceland	1.380	1.624	1.026	0.591
4	Netherlands	1.396	1.522	0.999	0.557

In [18]:

```
# changing the indices of the dataframe

happiness_report.set_index('Country or region', inplace=True)
```

In [19]:

```
happiness_report.head()
```

Out[19]:

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557

In [20]:

```
# now let's join two dataset we have prepared
#corona dataset
corona_data.head()
```

Out[20]:

Country/Region	max_infection_rate
Afghanistan	232.0
Albania	34.0
Algeria	199.0
Andorra	43.0
Angola	5.0

In [21]:

```
corona_data.shape
```

Out[21]:

```
(187, 1)
```

In [22]:

```
happiness_report.head()
```

Out[22]:

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557

In [23]:

```
happiness_report.shape
```

Out[23]:

```
(156, 4)
```

In [24]:

```
data = corona_data.join(happiness_report, how='inner')
data.head()
```

Out[24]:

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	232.0	0.350	0.517	0.361	0.000
Albania	34.0	0.947	0.848	0.874	0.383
Algeria	199.0	1.002	1.160	0.785	0.086
Argentina	291.0	1.092	1.432	0.881	0.471
Armenia	134.0	0.850	1.055	0.815	0.283

In [25]:

```
data.shape
```

Out[25]:

```
(143, 5)
```

In [26]:

```
# correlation matrix
data.corr()
```

Out[26]:

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
max_infection_rate	1.000000	0.250118	0.191958	0.289263	0.078196
GDP per capita	0.250118	1.000000	0.759468	0.863062	0.394603
Social support	0.191958	0.759468	1.000000	0.765286	0.456246
Healthy life expectancy	0.289263	0.863062	0.765286	1.000000	0.427892
Freedom to make life choices	0.078196	0.394603	0.456246	0.427892	1.000000

In [27]:

```
# Visualization of the results
data.head()
```

Out[27]:

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	232.0	0.350	0.517	0.361	0.000
Albania	34.0	0.947	0.848	0.874	0.383
Algeria	199.0	1.002	1.160	0.785	0.086
Argentina	291.0	1.092	1.432	0.881	0.471
Armenia	134.0	0.850	1.055	0.815	0.283

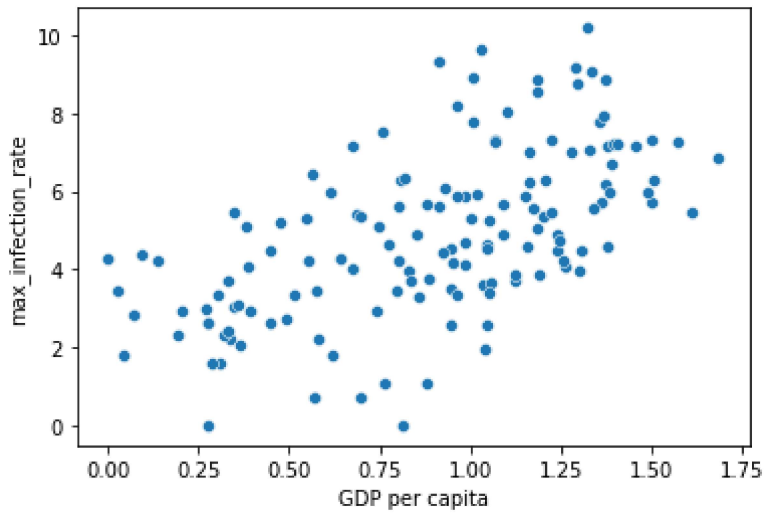
In [28]:

```
# plotting GDP vsmaximum Infection rate

x = data["GDP per capita"]
y = data["max_infection_rate"]

sns.scatterplot(x,np.log(y))
```

```
C:\Users\ichah\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: Future
Warning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[28]:

```
<AxesSubplot:xlabel='GDP per capita', ylabel='max_infection_rate'>
```



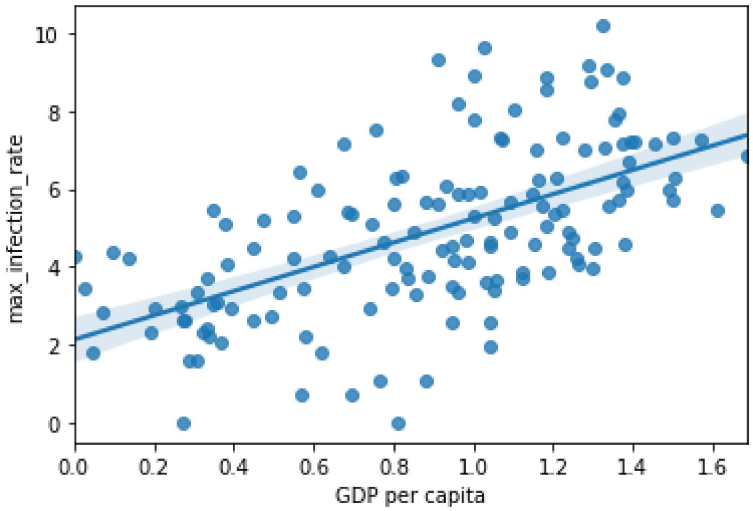
In [29]:

```
sns.regplot(x,np.log(y))
```

```
C:\Users\ichah\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: Future
Warning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[29]:

```
<AxesSubplot:xlabel='GDP per capita', ylabel='max_infection_rate'>
```



In [ ]:

In [ ]:

In [ ]: