In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import pandas as pd
import random
import math
import time
```

In [2]:

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import RandomizedSearchCV, train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error,r2_score
import datetime
plt.style.use('ggplot')
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

In [3]:

```
import os

os.chdir('C:\\Users\\Dell\\Documents\\Assignments_Placement\\1.Vumonic\\datasets')
```

Loading all datasets

In [4]:

```
recoveries_df = pd.read_csv("1.time_series_covid19_recovered_global.csv")
deaths_df = pd.read_csv("2.time_series_covid19_deaths_global.csv")
confirmed_df = pd.read_csv("3.time_series_covid19_confirmed_global.csv")
latest_data = pd.read_csv("5.csse_daily.csv")
medical_data =pd.read_csv("07-12-2020.csv")
```

In [5]:

```
latest_data.head()
```

Out[5]:

	FIPS	Admin2	Province_State	Country_Region	Last_Update	Lat	Long_	C
0	45001.0	Abbeville	South Carolina	US	2020-07-13 04:43:04	34.223334	-82.461707	
1	22001.0	Acadia	Louisiana	US	2020-07-13 04:43:04	30.295065	-92.414197	
2	51001.0	Accomack	Virginia	US	2020-07-13 04:43:04	37.767072	-75.632346	
3	16001.0	Ada	Idaho	US	2020-07-13 04:43:04	43.452658	-116.241552	
4	19001.0	Adair	Iowa	US	2020-07-13 04:43:04	41.330756	-94.471059	

In [6]:

```
confirmed_df.head()
```

Out[6]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	
1	NaN	Albania	41.15330	20.168300	0	0	0	0	
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	

5 rows × 178 columns

In [7]:

```
medical_data.head()
```

Out[7]:

	Province_State	Country_Region	Last_Update	Lat	Long_	Confirmed	Deaths	Reco
0	Alabama	US	2020-07-13 04:43:16	32.3182	-86.9023	53587	1121	25
1	Alaska	US	2020-07-13 04:43:16	61.3707	-152.4044	1479	17	
2	American Samoa	US	2020-07-13 04:43:16	-14.2710	-170.1320	0	0	
3	Arizona	US	2020-07-13 04:43:16	33.7298	-111.4312	122467	2237	14
4	Arkansas	US	2020-07-13 04:43:16	34.9697	-92.3731	28367	321	21

In [8]:

```
cols = confirmed_df.keys()
```

Loading only dates in dataframes

In [9]:

```
confirmed = confirmed_df.loc[:, cols[4]:]
deaths = deaths_df.loc[:, cols[4]:]
recoveries = recoveries_df.loc[:, cols[4]:]
```

In [10]:

```
dates = confirmed.keys()
world_cases = []
total_deaths = []
mortality_rate = []
recovery_rate = []
total_recovered = []
total_active = []
```

Get all cases hike from dates for the outbreak

In [11]:

```
for i in dates:
    confirmed_sum = confirmed[i].sum()
    death_sum = deaths[i].sum()
    recovered_sum = recoveries[i].sum()

    # confirmed, deaths, recovered, and active
    world_cases.append(confirmed_sum)
    total_deaths.append(death_sum)
    total_recovered.append(recovered_sum)
    total_active.append(confirmed_sum-death_sum-recovered_sum)

    # calculate rates
    mortality_rate.append(death_sum/confirmed_sum)
    recovery_rate.append(recovered_sum/confirmed_sum)
```

Daily count of cases

In [12]:

```
def daily_increase(data):
    d = []
    for i in range(len(data)):
        if i == 0:
            d.append(data[0])
        else:
            d.append(data[i]-data[i-1])
    return d
```

Moving Average of cases

In [13]:

```
def moving_average(data, step_size):
    moving_average = []
    for i in range(len(data)):
        if i + step_size < len(data):
            moving_average.append(np.mean(data[i:i+ step_size]))
        else:
            moving_average.append(np.mean(data[i:len(data)]))
    return moving_average
```

In [14]:

```
# step size from week will be 7
step = 7

# confirmed cases
world_daily_increase = daily_increase(world_cases)
world_confirmed_avg= moving_average(world_cases, step)
world_daily_increase_avg = moving_average(world_daily_increase, step)
```

In [15]:

```
# deaths
world_daily_death = daily_increase(total_deaths)
world_death_avg = moving_average(total_deaths, step)
world_daily_death_avg = moving_average(world_daily_death, step)
```

In [16]:

```
# recoveries
world_daily_recovery = daily_increase(total_recovered)
world_recovery_avg = moving_average(total_recovered, step)
world_daily_recovery_avg = moving_average(world_daily_recovery, step)


# active
world_active_avg = moving_average(total_active, step)
```

Reshaping values for ease of graphs

In [17]:

```
days_since_1_22 = np.array([i for i in range(len(dates))]).reshape(-1, 1)
world_cases = np.array(world_cases).reshape(-1, 1)
total_deaths = np.array(total_deaths).reshape(-1, 1)
total_recovered = np.array(total_recovered).reshape(-1, 1)
```

Creating Forcasting days

In [18]:

```
days_in_future = 10
future_forcast = np.array([i for i in range(len(dates)+days_in_future)]).reshape(-1, 1)
available_dates = future_forcast[:-10]
```

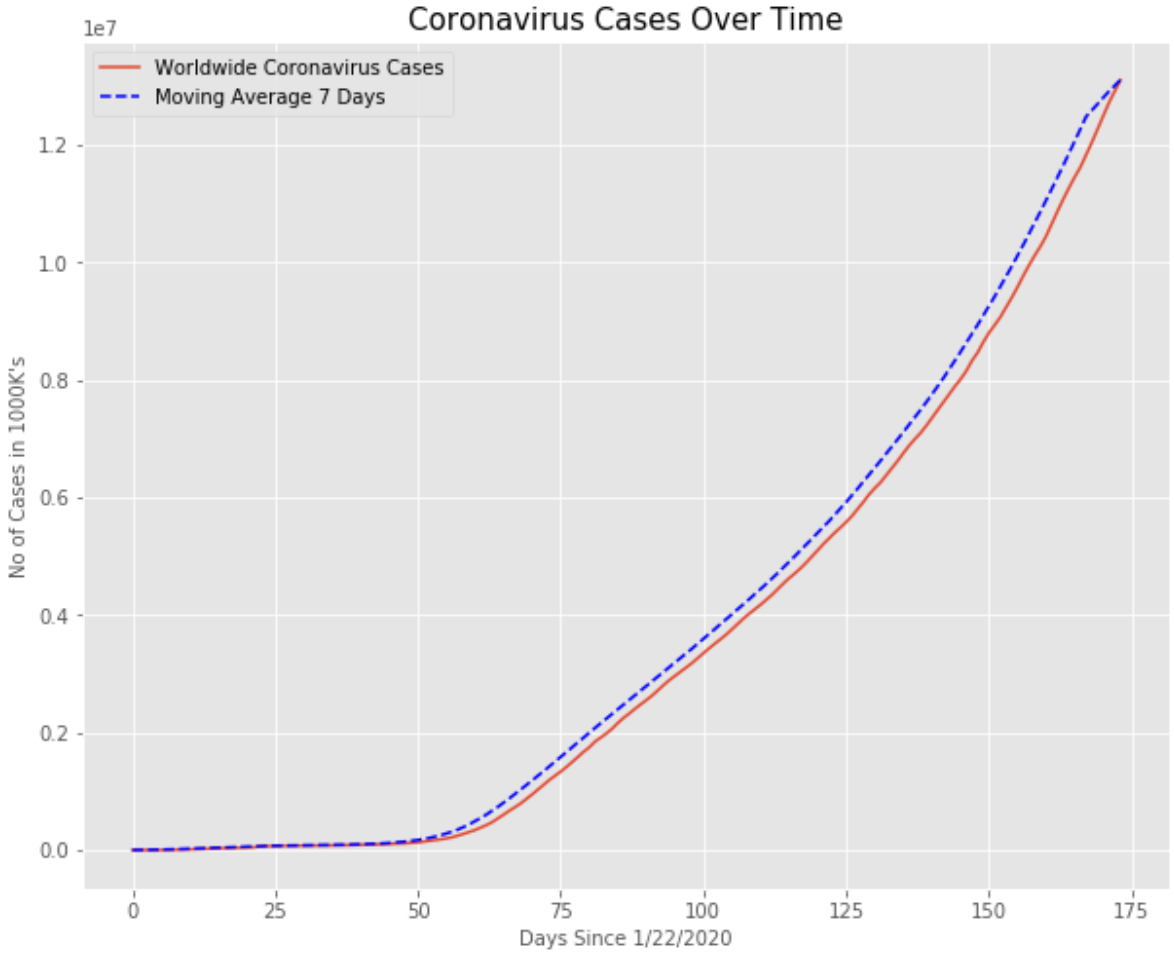
Graphing the number of Weekly Analysis of confirmed cases, active cases, deaths, recoveries

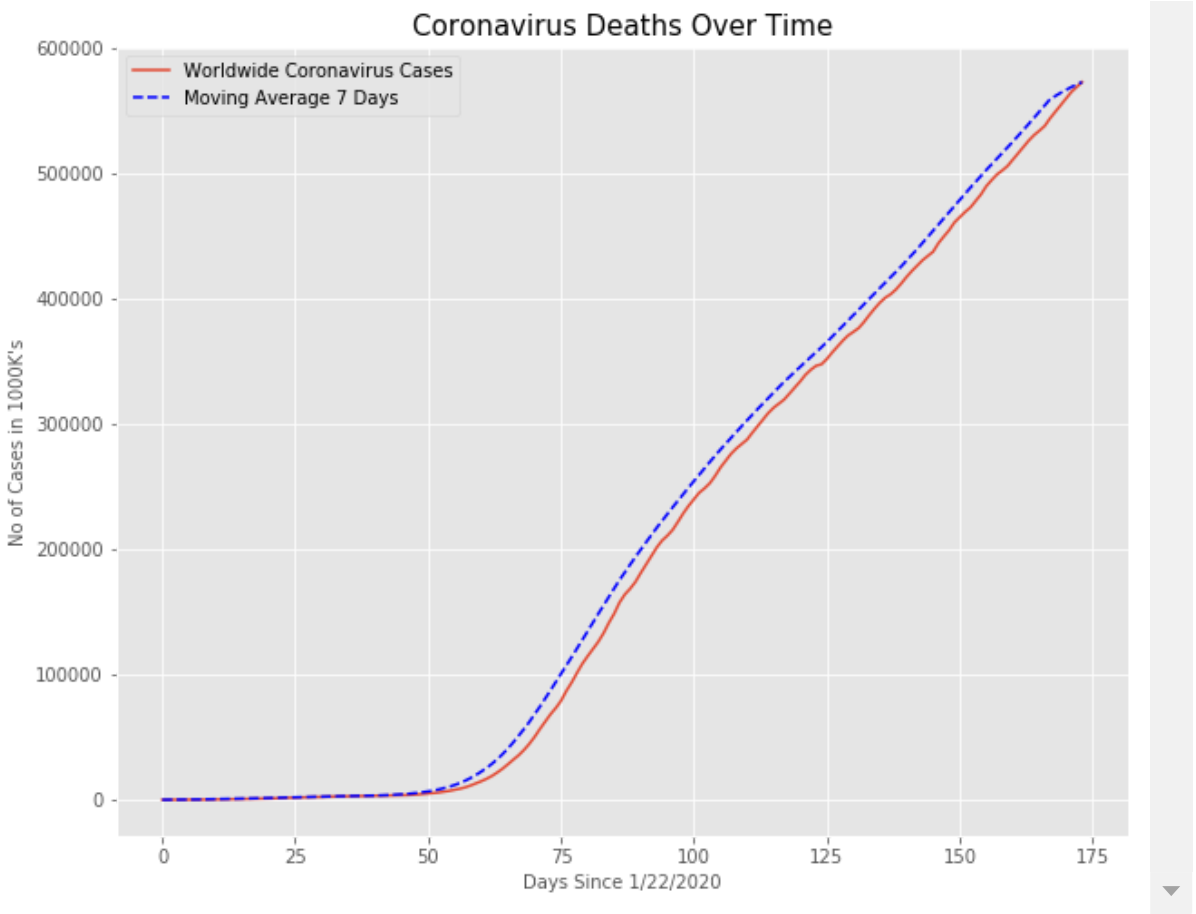
In [19]:

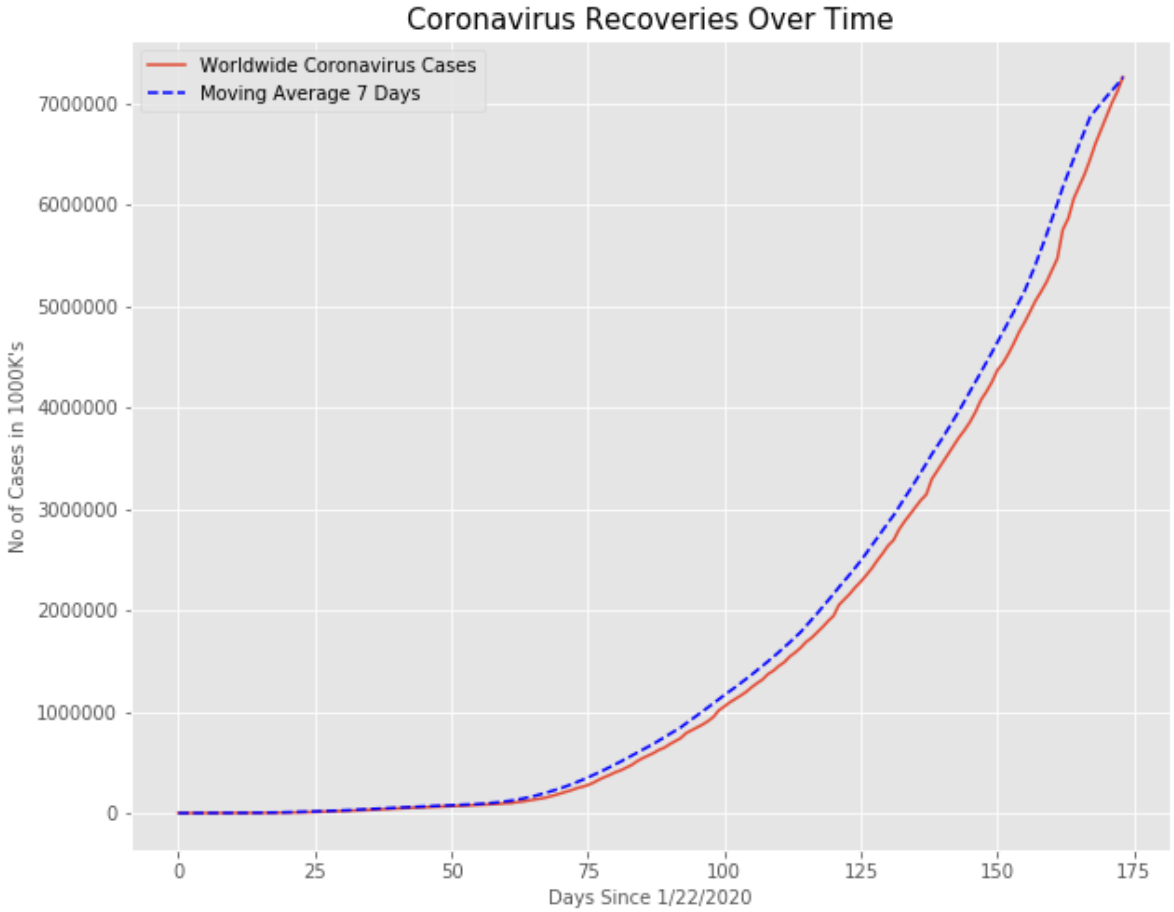
```
# Confirmed/Death/Recovery/Active Cases with moving avergae of it
def case_avg(days,case,avg,string):
    days = days.reshape(1, -1)[0]
    plt.figure(figsize=(10, 8))
    plt.plot(days, case)
    plt.plot(days, avg, '--b')
    plt.title('Coronavirus {} Over Time'.format(string), size=15)
    plt.xlabel('Days Since 1/22/2020', size=10)
    plt.ylabel("No of Cases in 1000K's", size=10)
    plt.legend(['Worldwide Coronavirus Cases', 'Moving Average {} Days'.format(step)], prop
    plt.show()
```

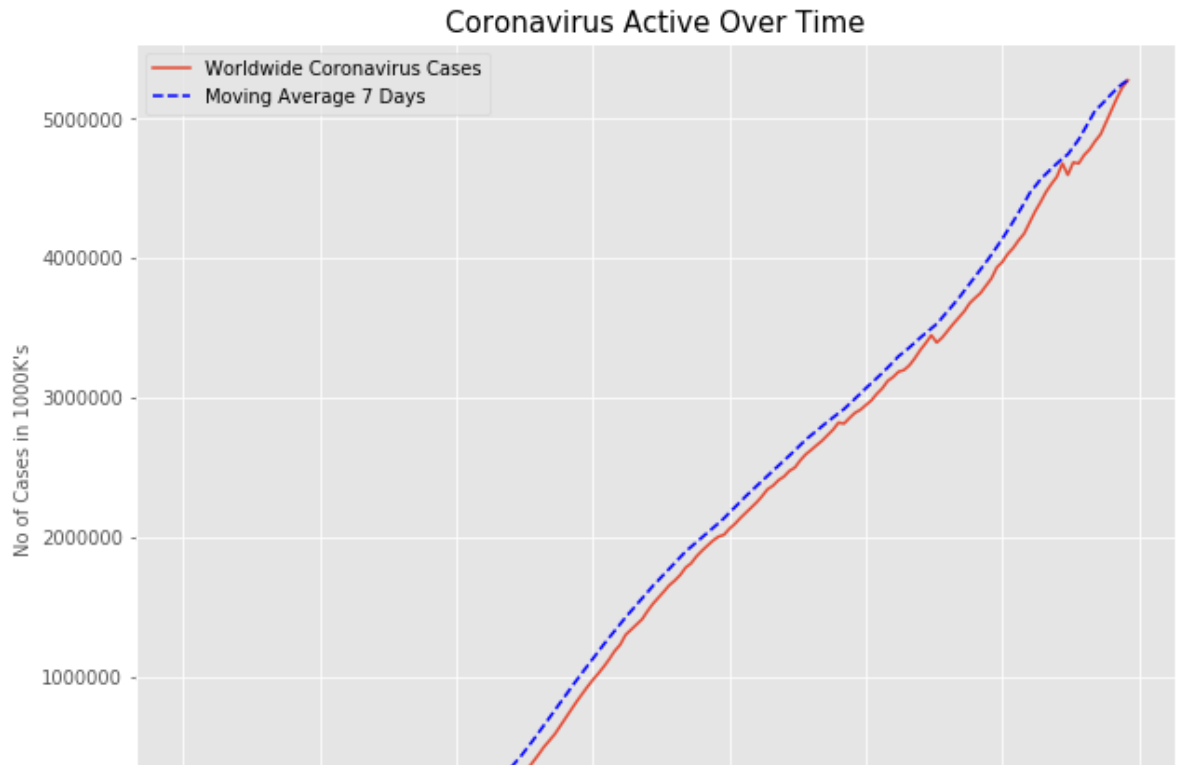
In [20]:

```
case_avg(available_dates, world_cases,world_confirmed_avg,'Cases')
case_avg(available_dates, total_deaths,world_death_avg,'Deaths')
case_avg(available_dates, total_recovered,world_recovery_avg,'Recoveries')
case_avg(available_dates, total_active,world_active_avg,'Active')
```




Coronavirus Deaths Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
No of Cases in 1000K's
Days Since 1/22/2020
0
100000
200000
300000
400000
500000
600000
0
25
50
75
100
125
150
175


Coronavirus Recoveries Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
No of Cases in 1000K's
Days Since 1/22/2020
0
1000000
2000000
3000000
4000000
5000000
6000000
7000000
0
25
50
75
100
125
150
175

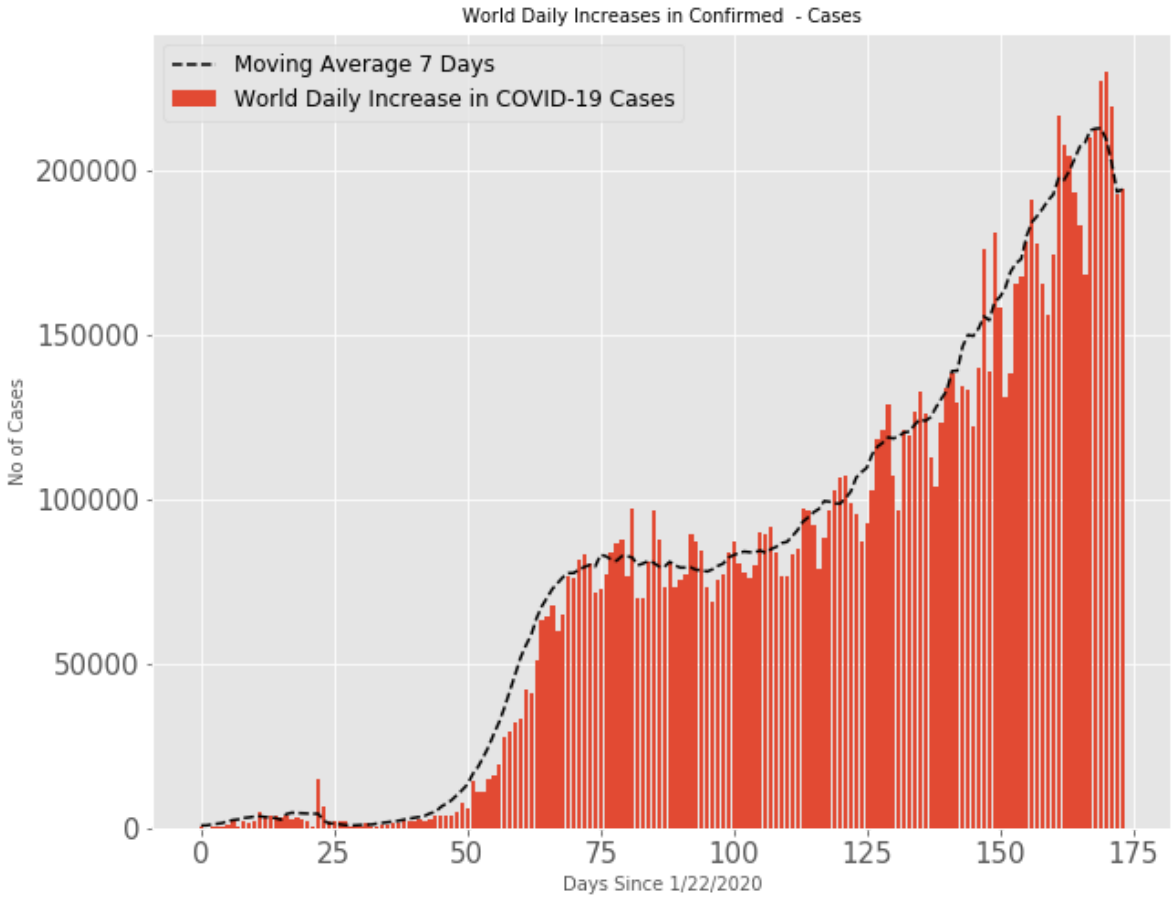


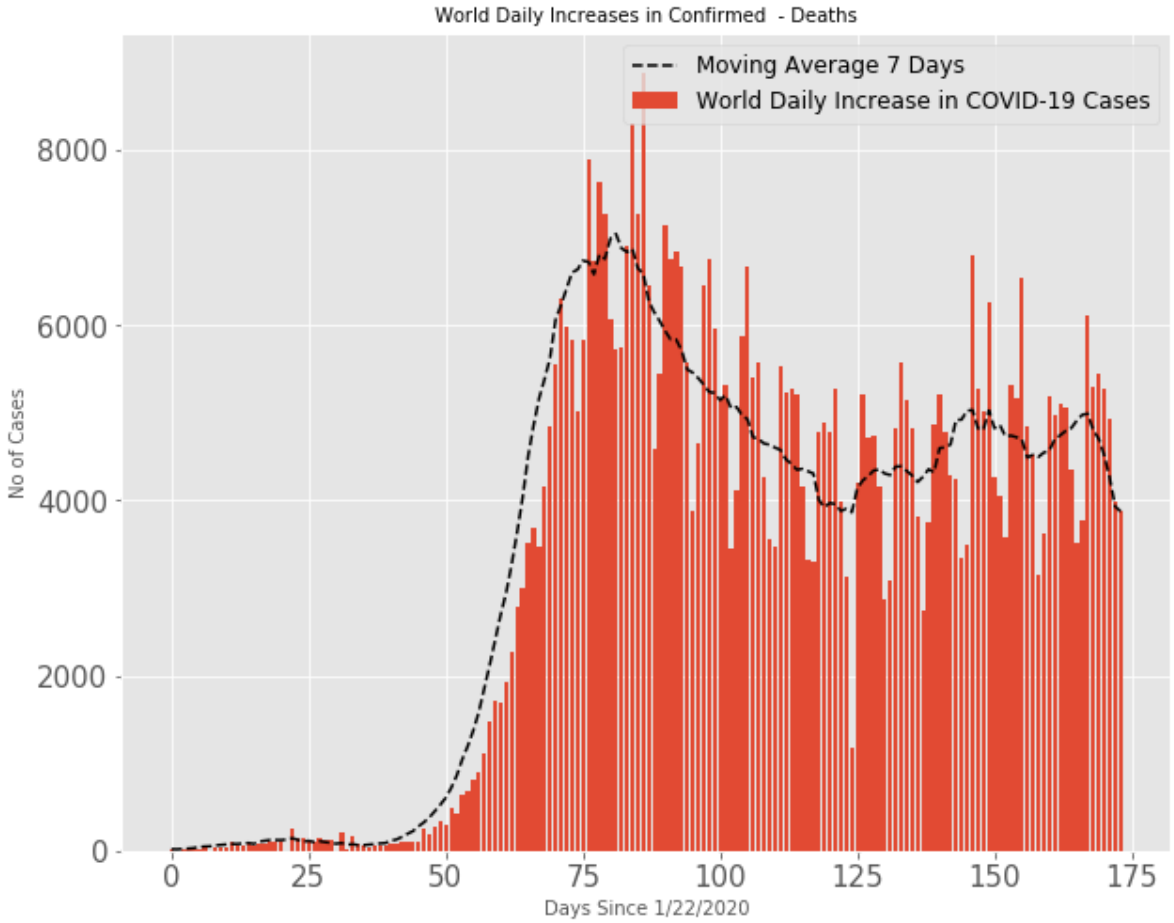
In [21]:

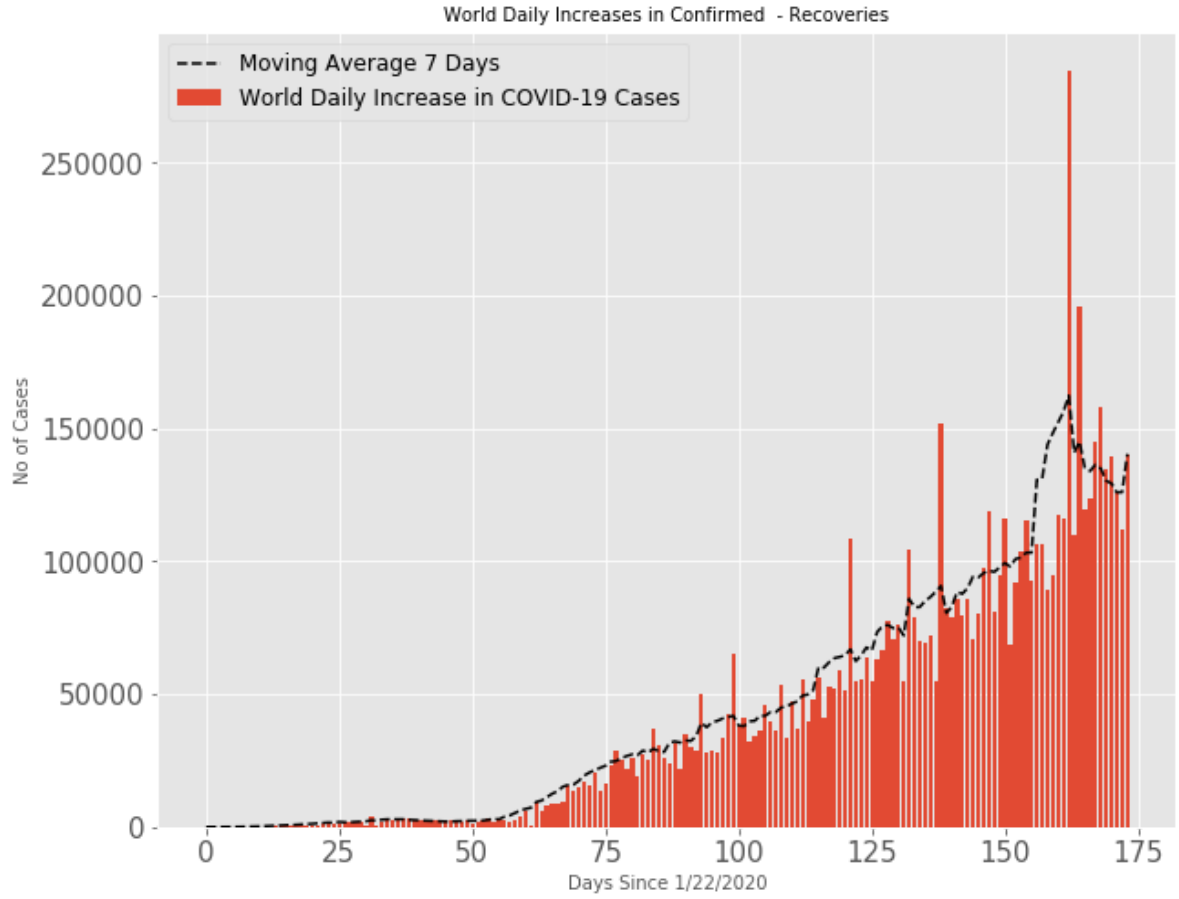
```
# Daily Analysis of Confirmed/Recovery/Death cases with moving average
def daily_graph(days,increase,increase_avg,case,string):
    days = days.reshape(1, -1)[0]
    plt.figure(figsize=(10, 8))
    plt.bar(days, increase)
    plt.plot(days, increase_avg, color='black', linestyle='dashed')
    plt.title('World Daily Increases in {}  - {}'.format(case,string), size=10)
    plt.xlabel('Days Since 1/22/2020', size=10)
    plt.ylabel('No of Cases', size=10)
    plt.legend(['Moving Average {} Days'.format(step), 'World Daily Increase in COVID-19 Ca
    plt.xticks(size=15)
    plt.yticks(size=15)
    plt.show()
```

In [22]:

```
daily_graph(available_dates,world_daily_increase,world_daily_increase_avg,'Confirmed','Case
daily_graph(available_dates,world_daily_death,world_daily_death_avg,'Confirmed','Deaths')
daily_graph(available_dates,world_daily_recovery,world_daily_recovery_avg,'Confirmed','Reco
```




World Daily Increases in Confirmed - Deaths
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
0
2000
4000
6000
8000
0
25
50
75
100
125
150
175
Days Since 1/22/2020



Country wise analysis

In [23]:

```
def country_plot(available_dates, country_cases, country_daily_increase, country_daily_deat
#def country_plot(x, y1, y2, y3, y4, country):
    # step is set as 14 in in the beginning of the notebook

    confirmed_avg = moving_average(country_cases, step)

    confirmed_increase_avg = moving_average(country_daily_increase, step)

    death_increase_avg = moving_average(country_daily_death, step)

    recovery_increase_avg = moving_average(country_daily_recovery, step)

    case_avg(available_dates,country_cases,confirmed_avg,str(country))

    daily_graph(available_dates,country_daily_increase,confirmed_increase_avg,'cases',str(c

    daily_graph(available_dates,country_daily_death,death_increase_avg,'deaths',str(country


    daily_graph(available_dates,country_daily_recovery,recovery_increase_avg,'recovery',str
```

In [24]:

```
# helper function for getting country's cases, deaths, and recoveries
def get_country_info(country_name):
    country_cases = []
    country_deaths = []
    country_recoveries = []

    for i in dates:
        country_cases.append(confirmed_df[confirmed_df['Country/Region']==country_name][i].
        country_deaths.append(deaths_df[deaths_df['Country/Region']==country_name][i].sum()
        country_recoveries.append(recoveries_df[recoveries_df['Country/Region']==country_na
    return (country_cases, country_deaths, country_recoveries)
```

In [25]:

```
def country_visualizations(country_name):
    country_info = get_country_info(country_name)
    country_cases = country_info[0]
    country_deaths = country_info[1]
    country_recoveries = country_info[2]

    country_daily_increase = daily_increase(country_cases)
    country_daily_death = daily_increase(country_deaths)
    country_daily_recovery = daily_increase(country_recoveries)

    country_plot(available_dates, country_cases, country_daily_increase, country_daily_deat
```

In [26]:

```
latest_data.head()
```

Out[26]:

	FIPS	Admin2	Province_State	Country_Region	Last_Update	Lat	Long_	C
0	45001.0	Abbeville	South Carolina	US	2020-07-13 04:43:04	34.223334	-82.461707	
1	22001.0	Acadia	Louisiana	US	2020-07-13 04:43:04	30.295065	-92.414197	
2	51001.0	Accomack	Virginia	US	2020-07-13 04:43:04	37.767072	-75.632346	
3	16001.0	Ada	Idaho	US	2020-07-13 04:43:04	43.452658	-116.241552	
4	19001.0	Adair	Iowa	US	2020-07-13 04:43:04	41.330756	-94.471059	

In [27]:

```
# Get top countries with cases
latest_data.groupby('Country_Region').sum().sort_values(by = 'Confirmed', ascending = False
```

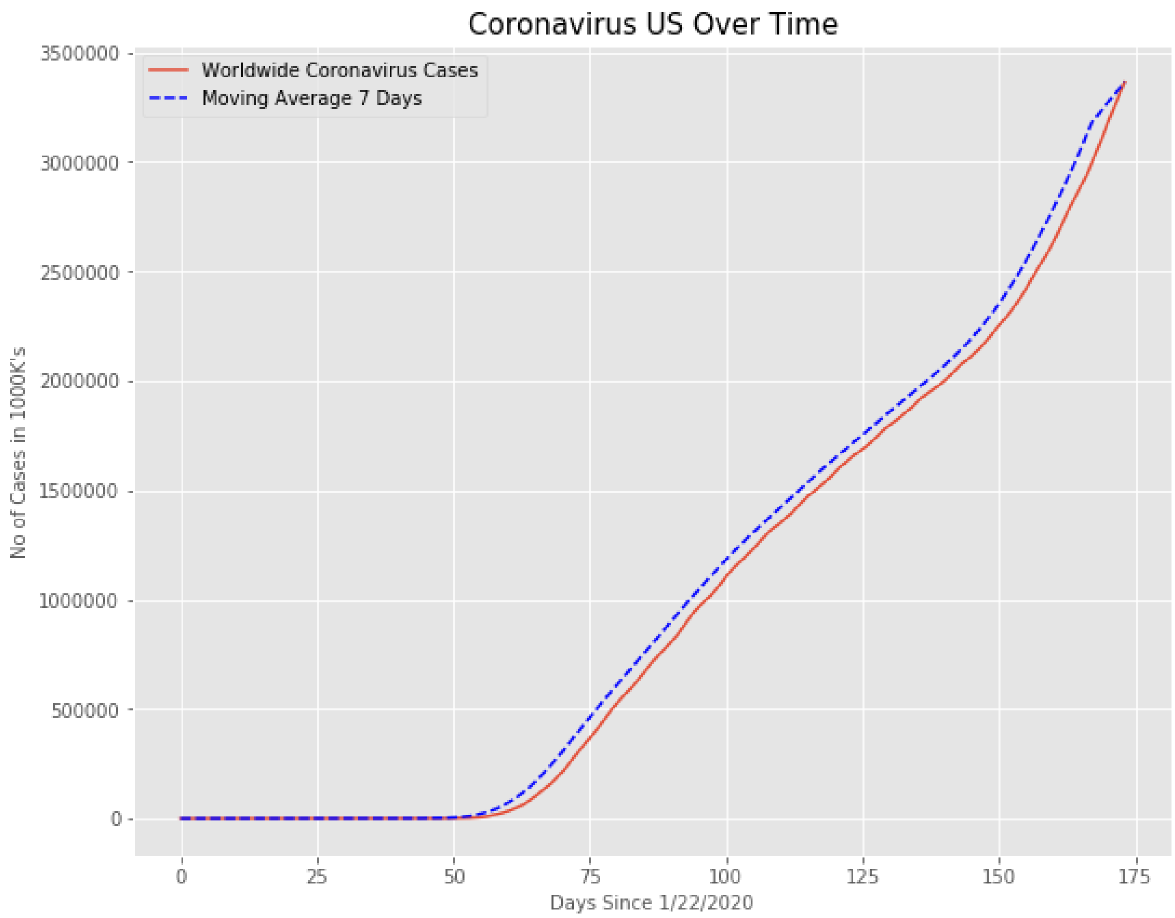
Out[27]:

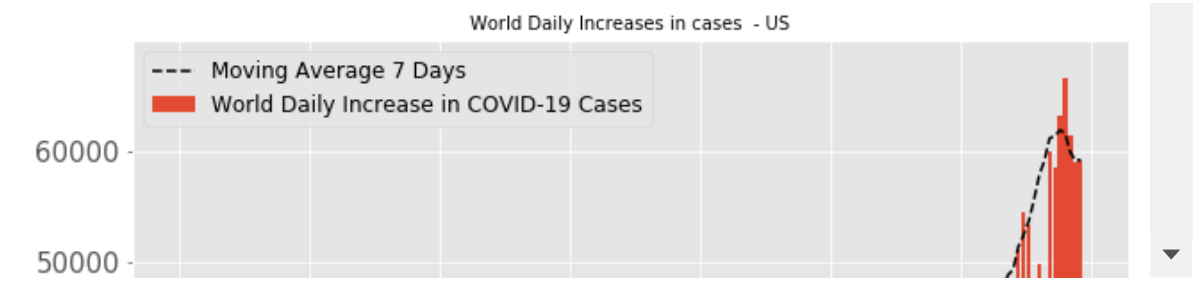
```
['US',
 'Brazil',
 'India',
 'Russia',
 'Peru',
 'Chile',
 'Mexico',
 'United Kingdom',
 'South Africa',
 'Iran']
```

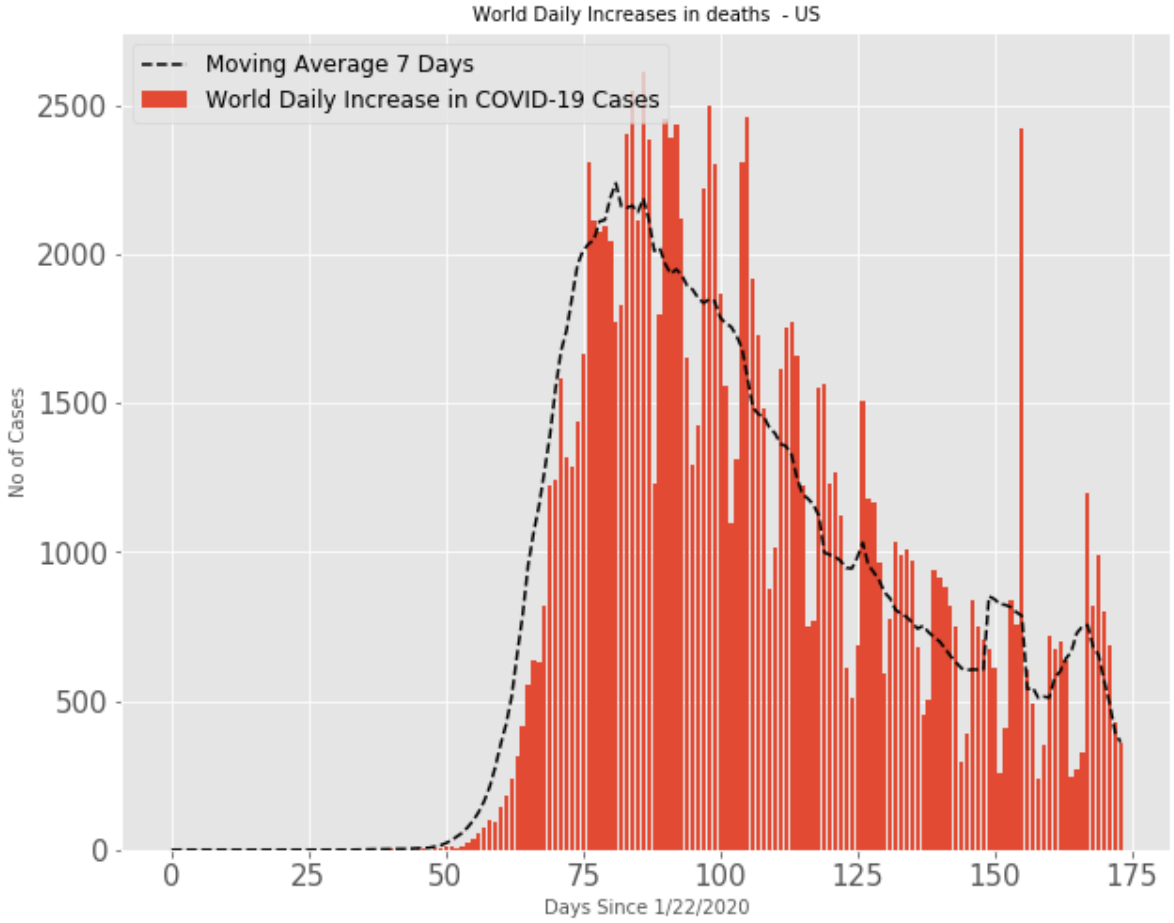
In [28]:

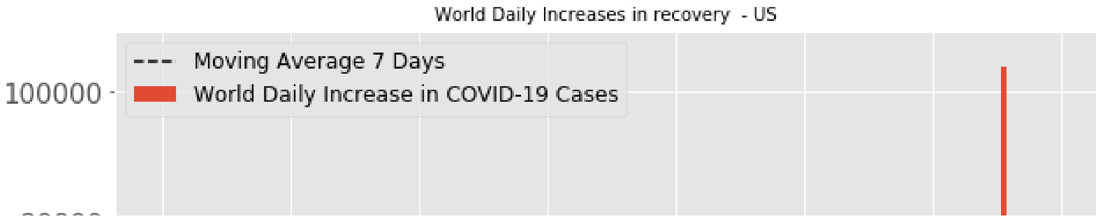
```
countries = latest_data.groupby('Country_Region').sum().sort_values(by = 'Confirmed', ascen
#countries = ['India']
for country in countries:
    print('-*-*'*10,end = '')
    print(country,end ='')
    print('-*-*'*10)
    country_visualizations(country)
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*US-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*
```

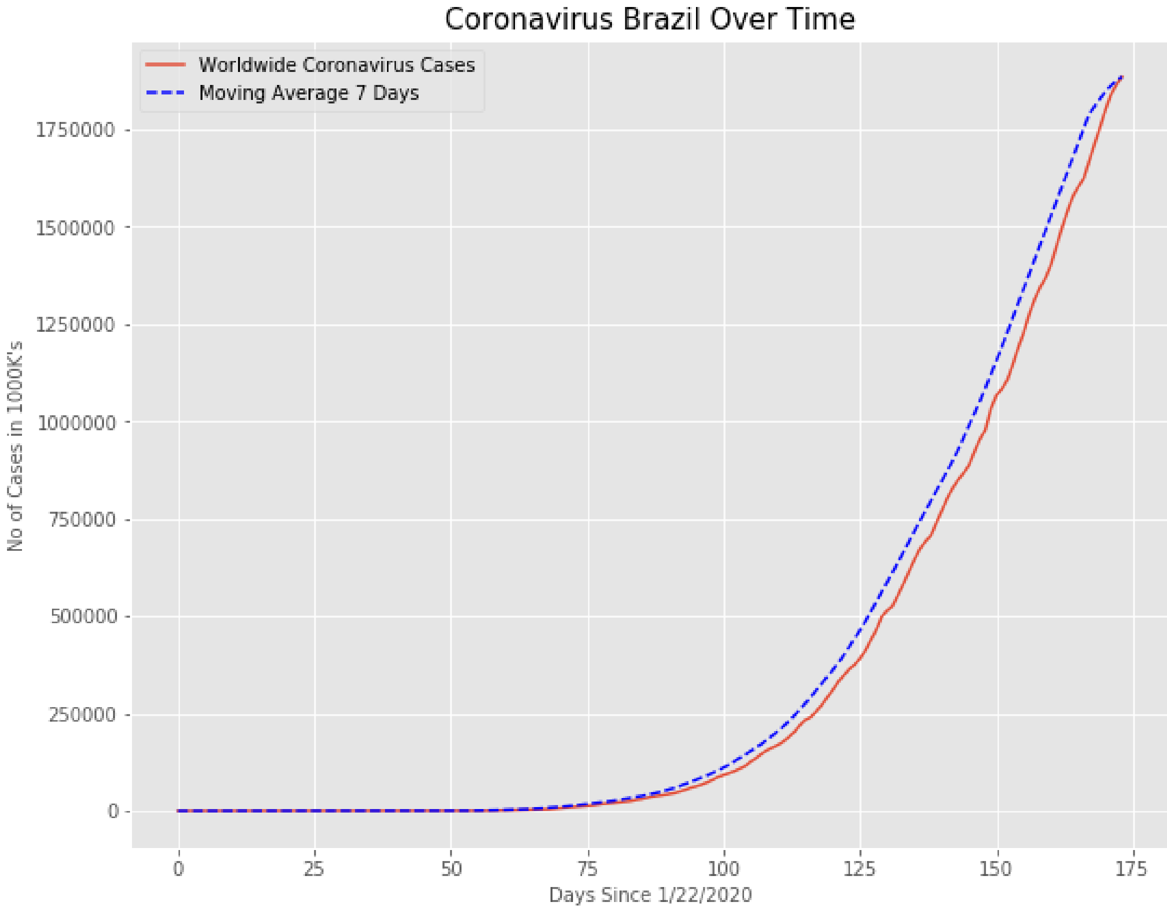


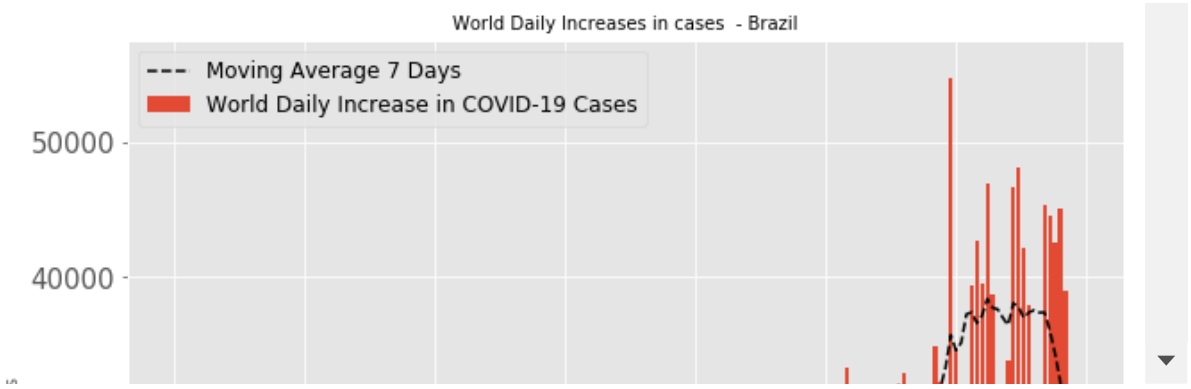

World Daily Increases in cases - US
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
60000
50000

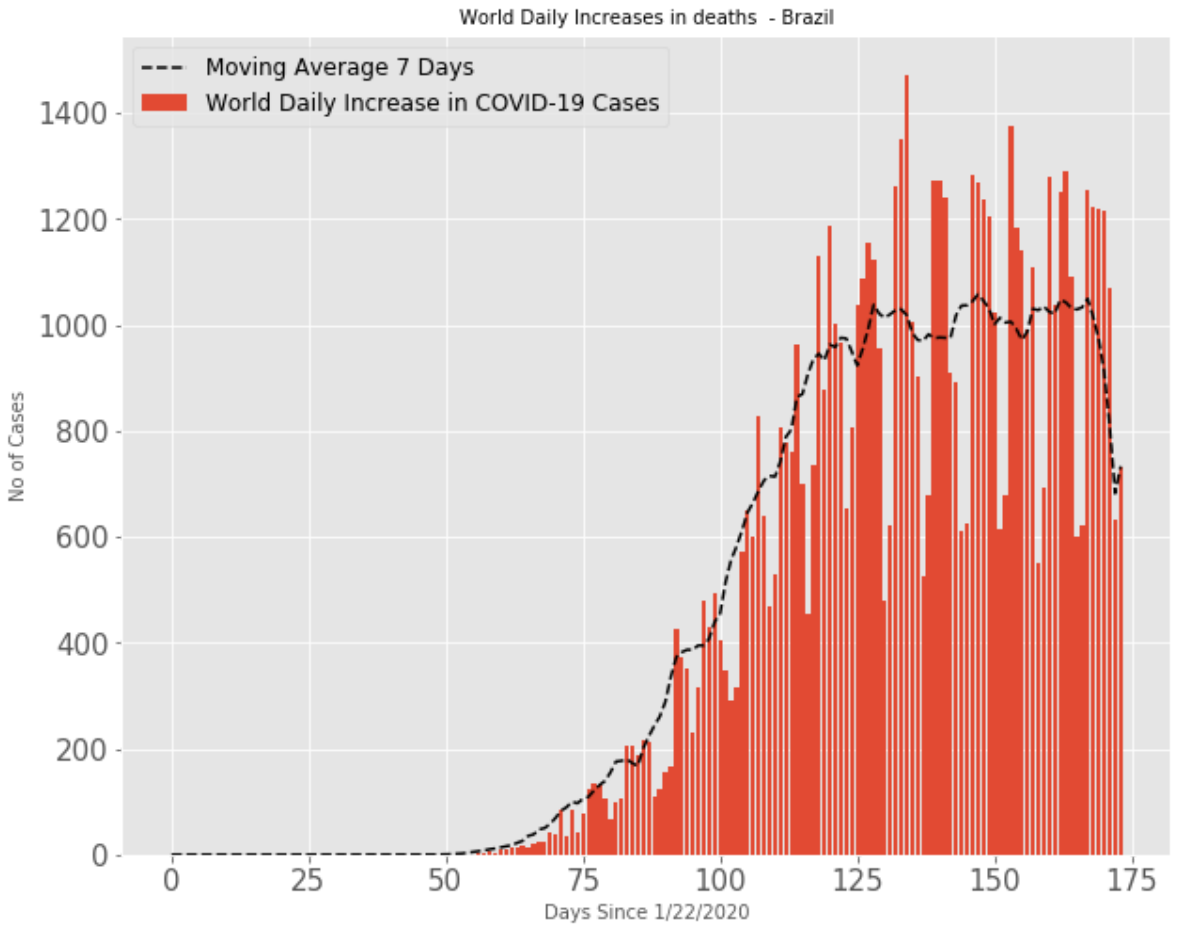

World Daily Increases in deaths - US
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
2500
2000
1500
1000
500
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

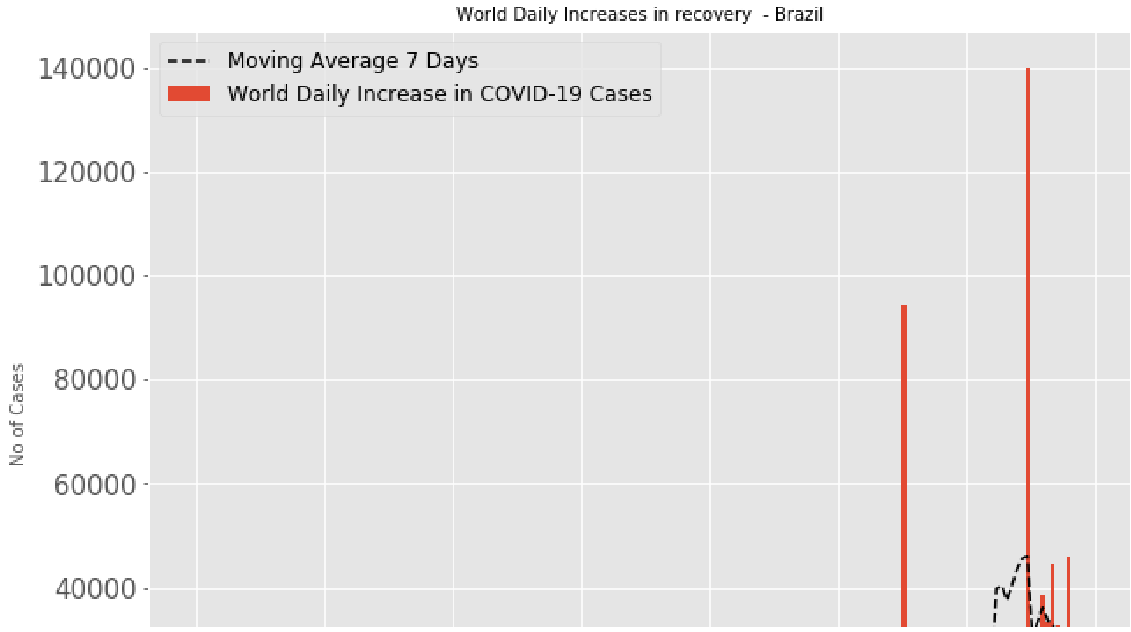

World Daily Increases in recovery - US
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
100000

-*Brazil-*

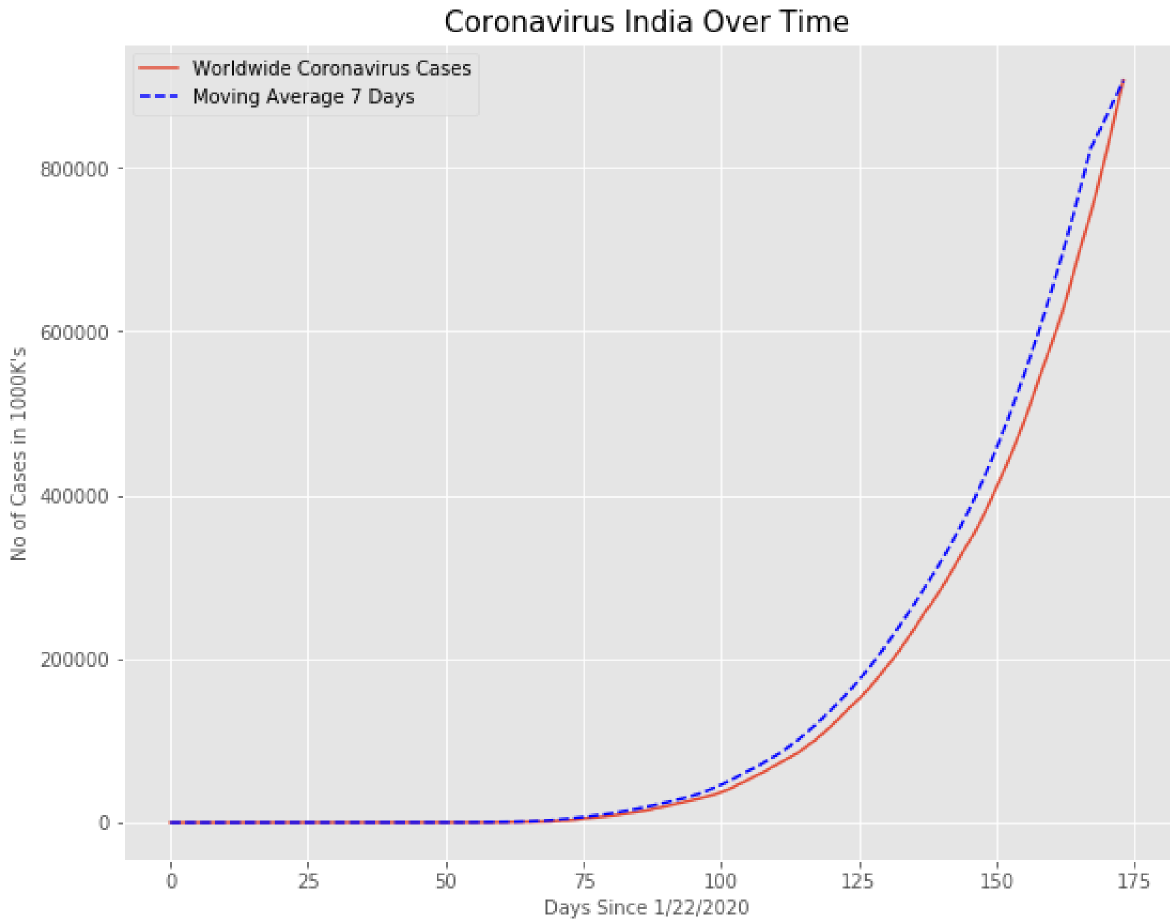

Coronavirus Brazil Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
No of Cases in 1000K's
Days Since 1/22/2020
0
250000
500000
750000
1000000
1250000
1500000
1750000
0
25
50
75
100
125
150
175

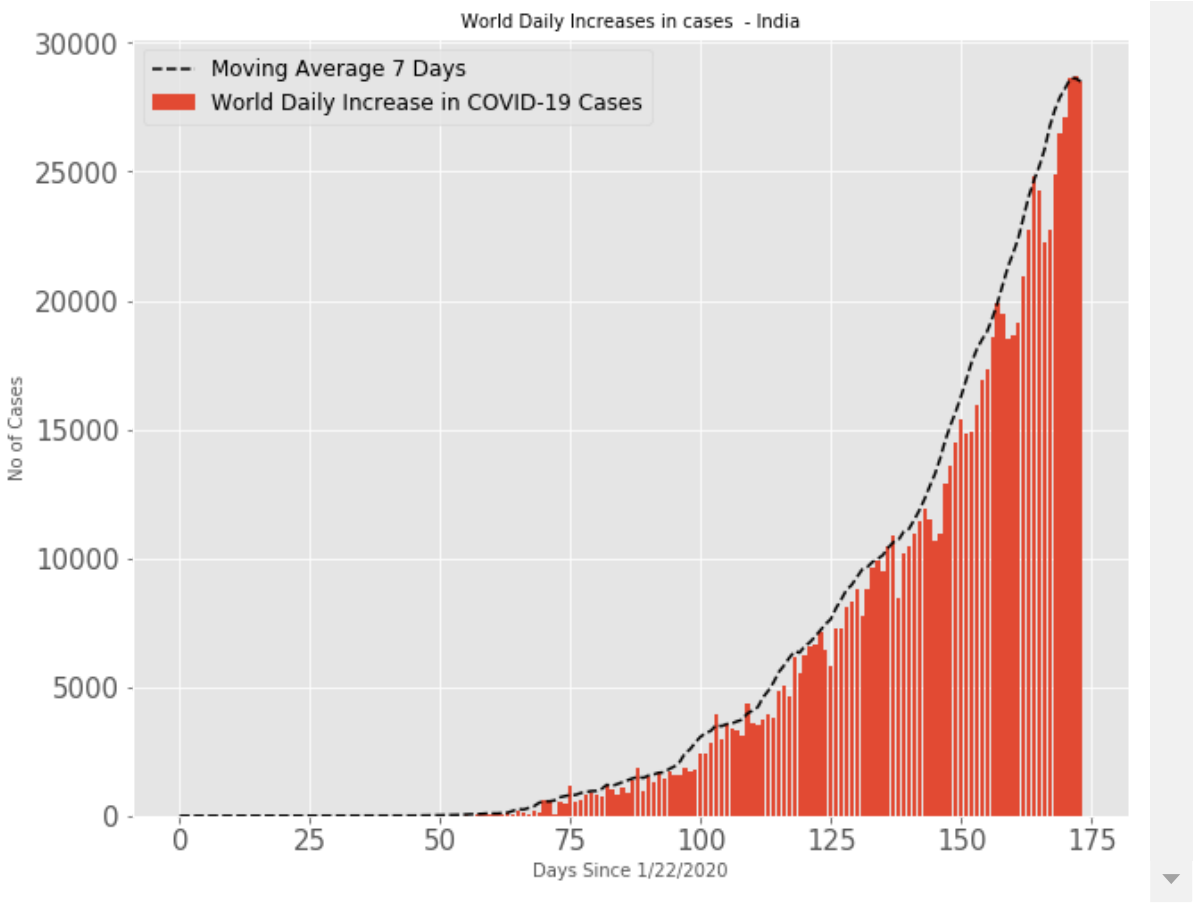

World Daily Increases in cases - Brazil
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
50000
40000

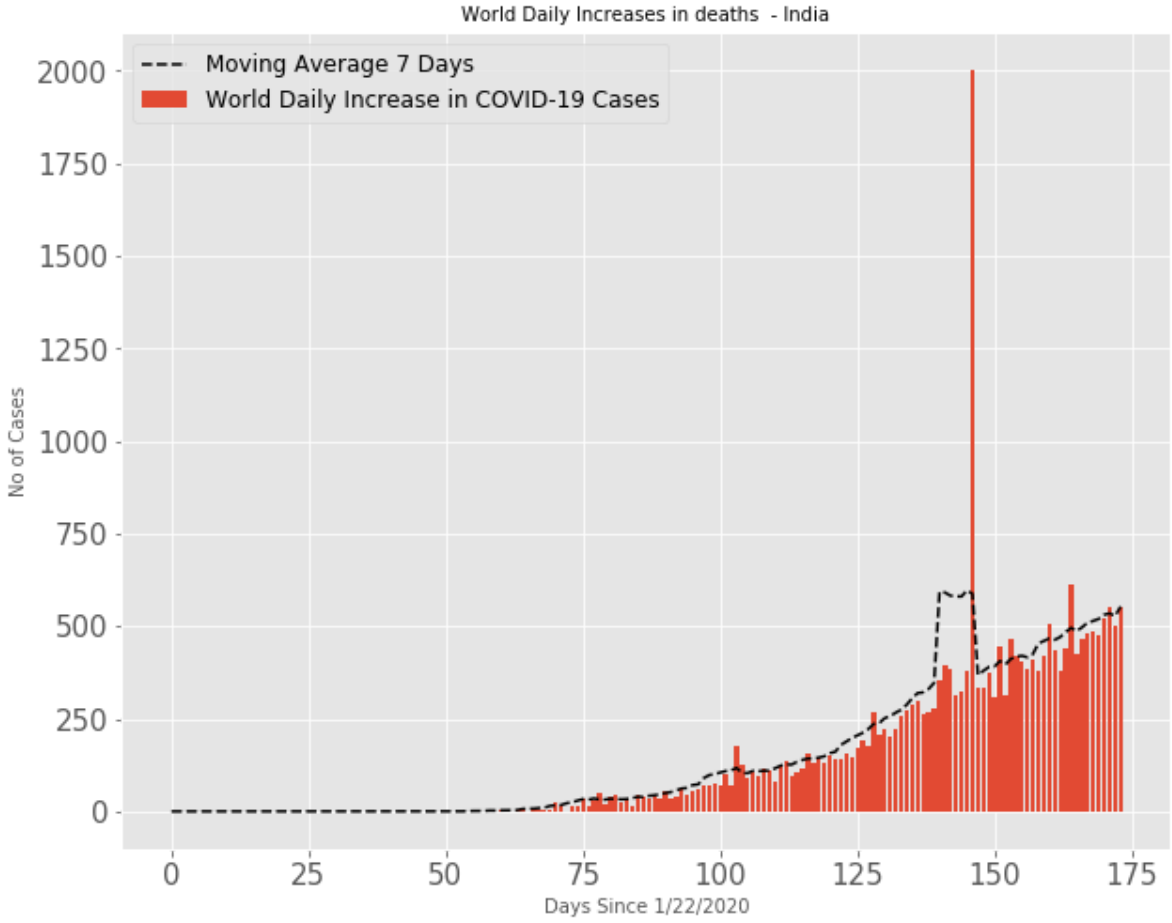

World Daily Increases in deaths - Brazil
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
1400
1200
1000
800
600
400
200
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

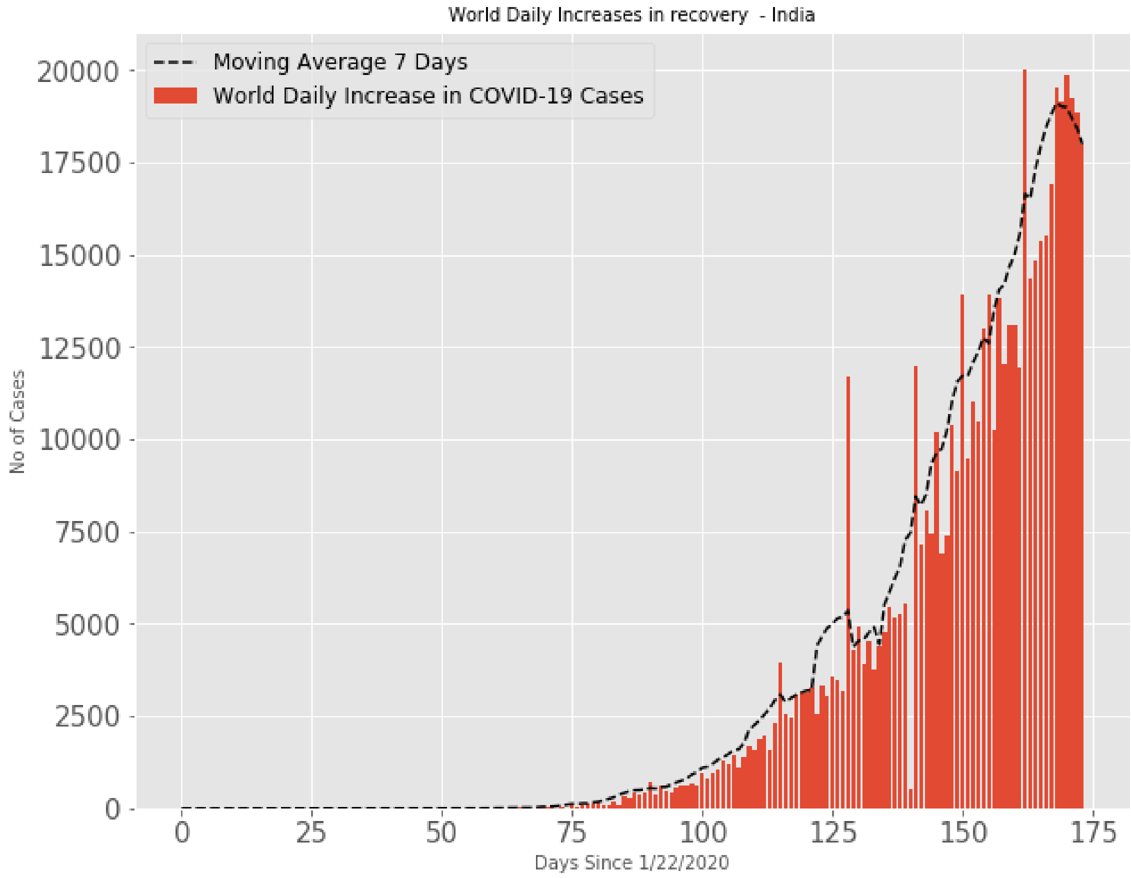

World Daily Increases in recovery - Brazil
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
140000
120000
100000
80000
60000
40000
No of Cases

-*India-*


Coronavirus India Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
800000
600000
400000
200000
0
No of Cases in 1000K's
0
25
50
75
100
125
150
175
Days Since 1/22/2020

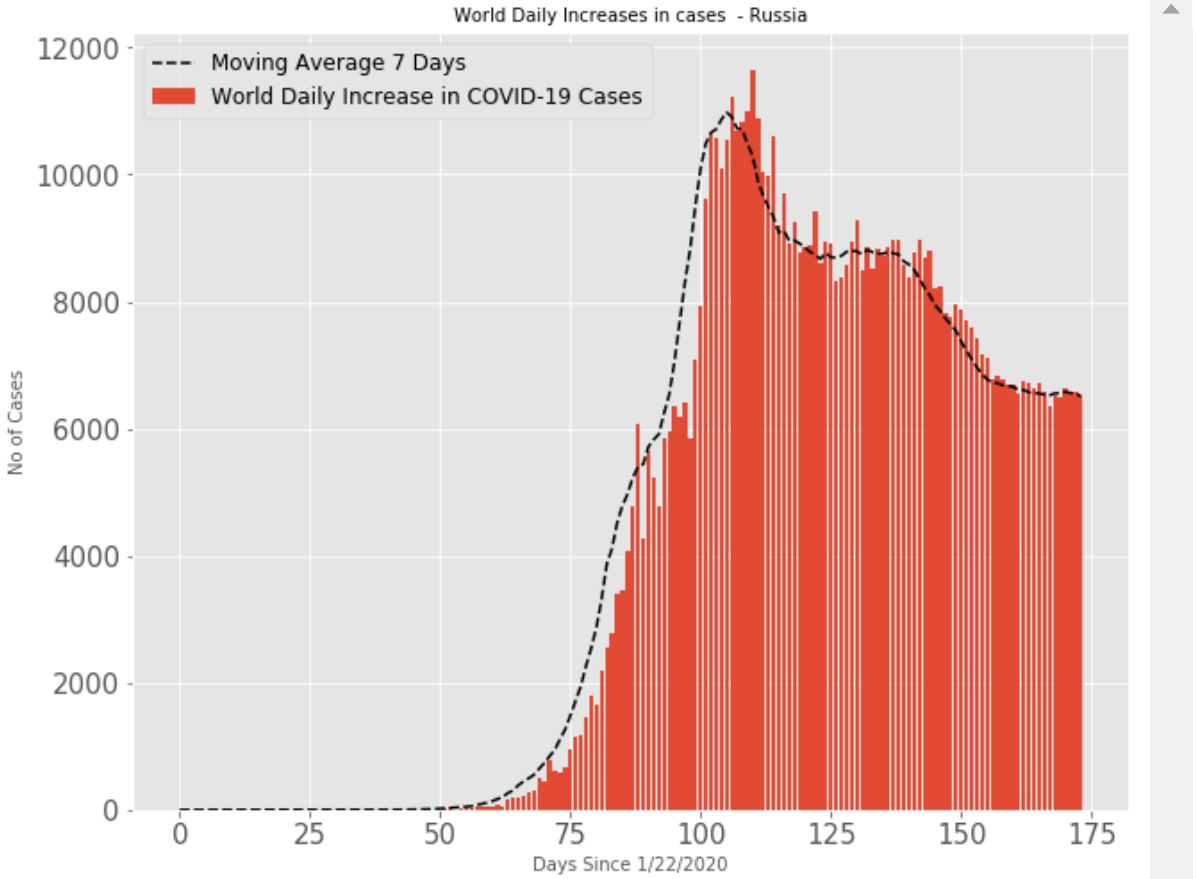

World Daily Increases in cases - India
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
Days Since 1/22/2020
0
5000
10000
15000
20000
25000
30000
0
25
50
75
100
125
150
175

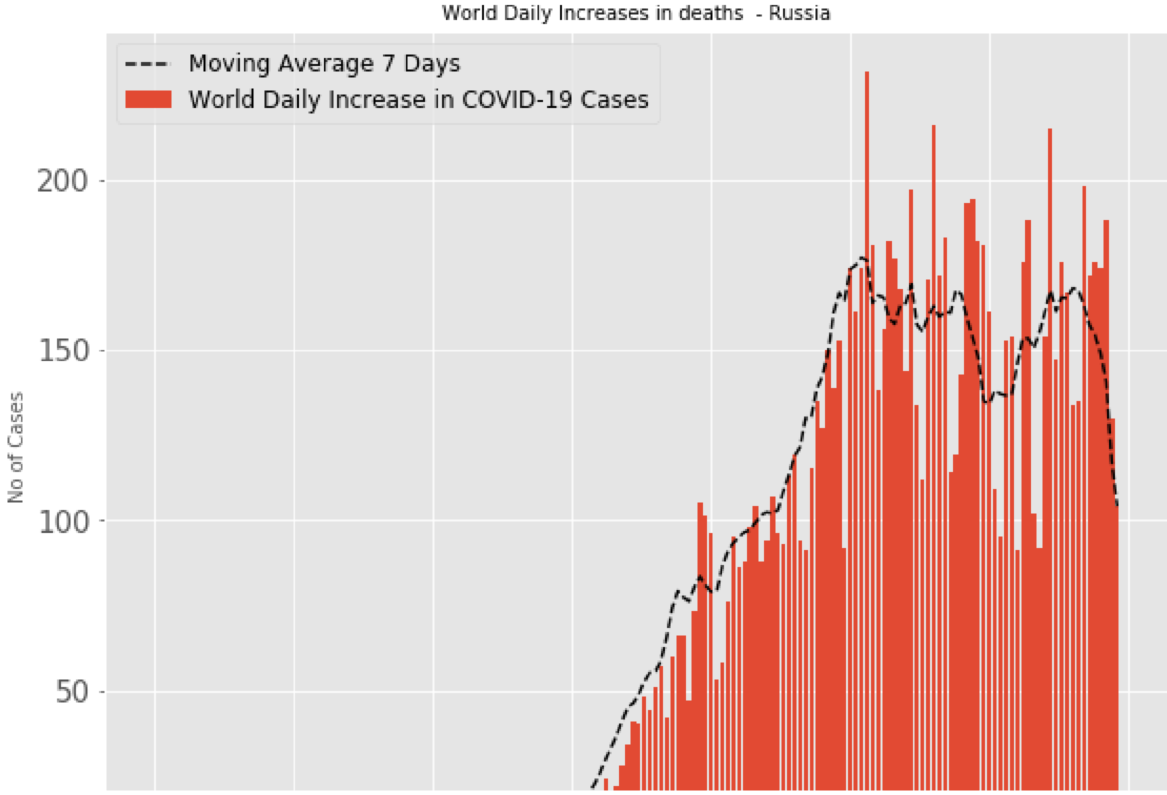

World Daily Increases in deaths - India
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
Days Since 1/22/2020
0
250
500
750
1000
1250
1500
1750
2000
0
25
50
75
100
125
150
175

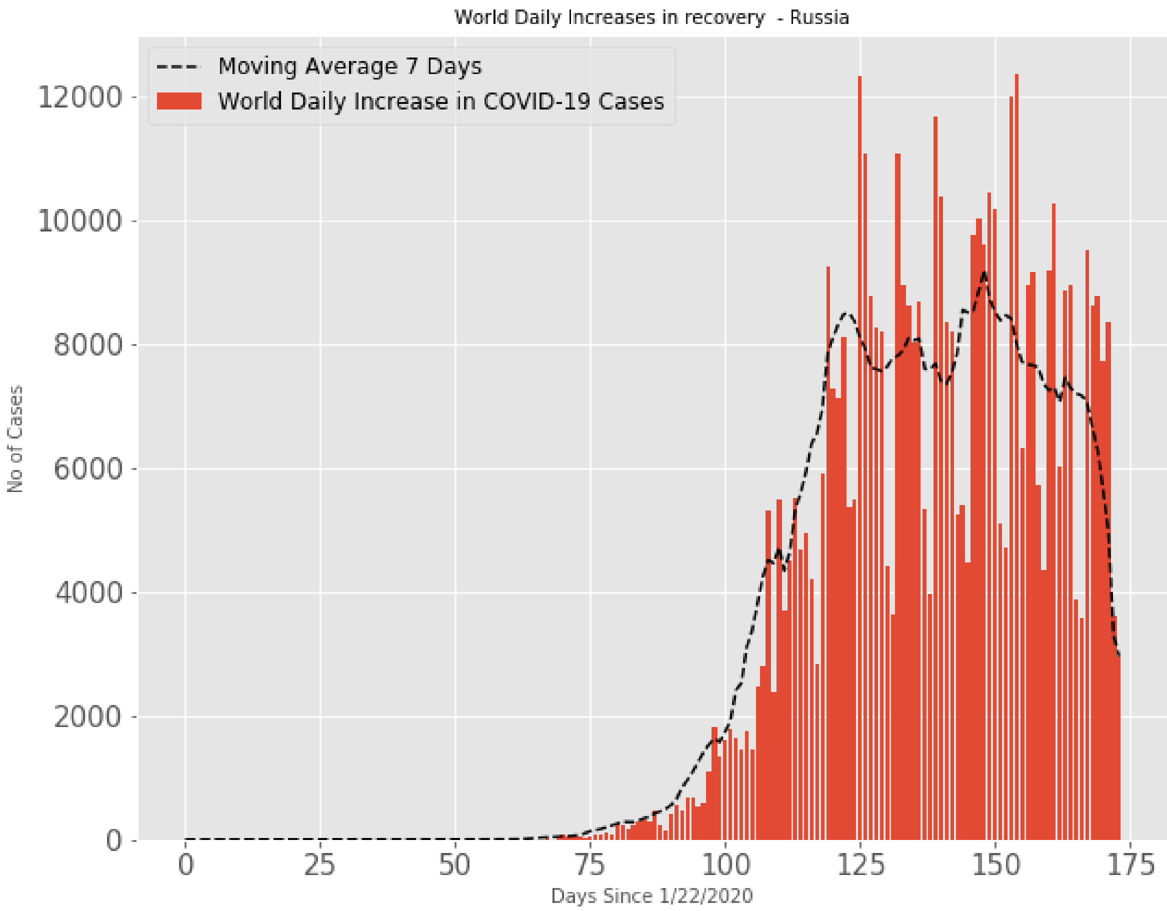

World Daily Increases in recovery - India
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
Days Since 1/22/2020
0
2500
5000
7500
10000
12500
15000
17500
20000
0
25
50
75
100
125
150
175

-*Russia-*

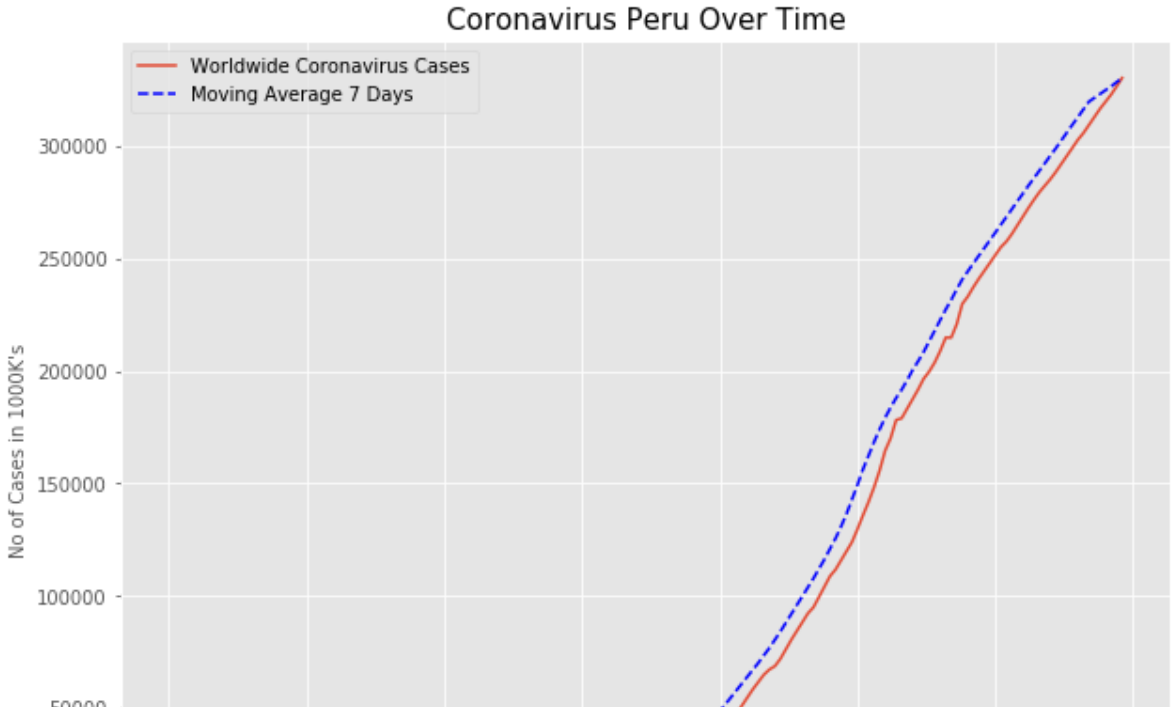

Coronavirus Russia Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
700000
600000


World Daily Increases in cases - Russia
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
12000
10000
8000
6000
4000
2000
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

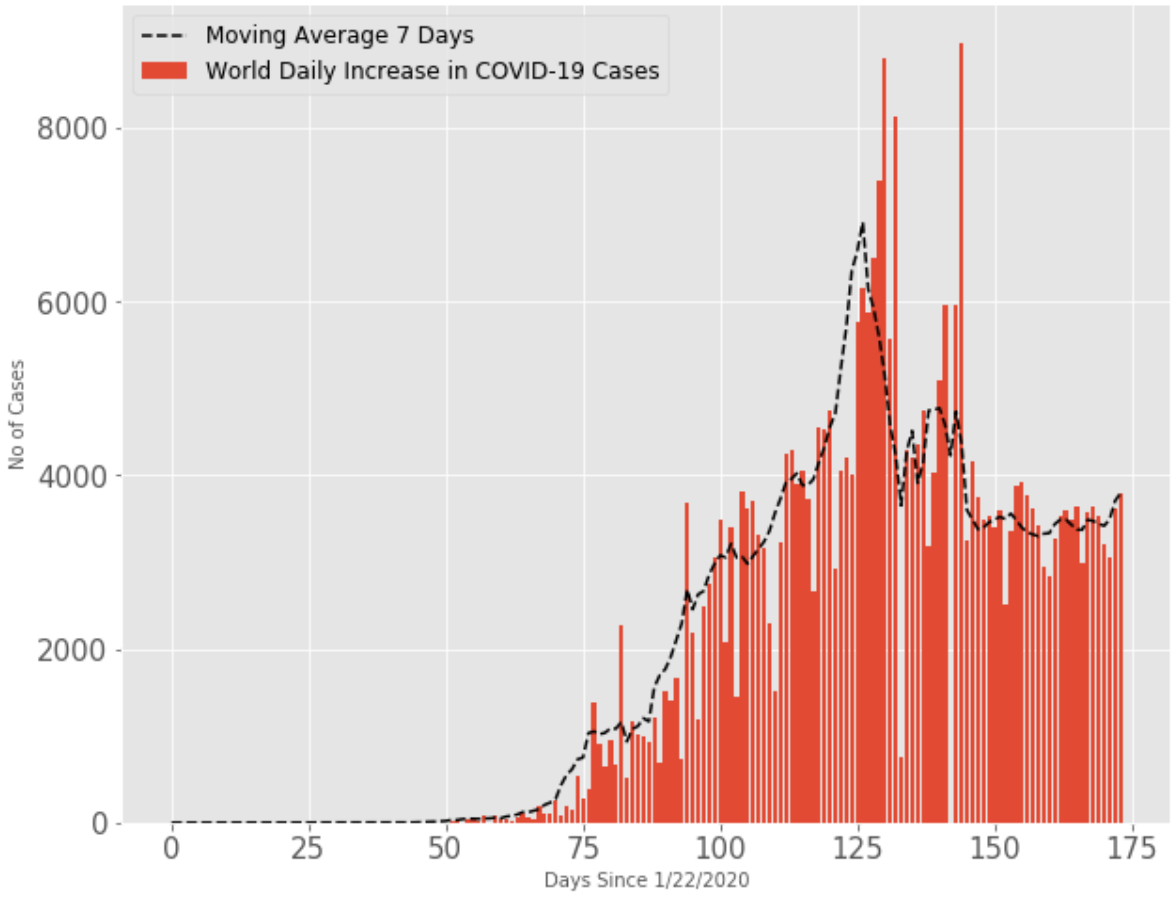

World Daily Increases in deaths - Russia
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
200
150
100
50

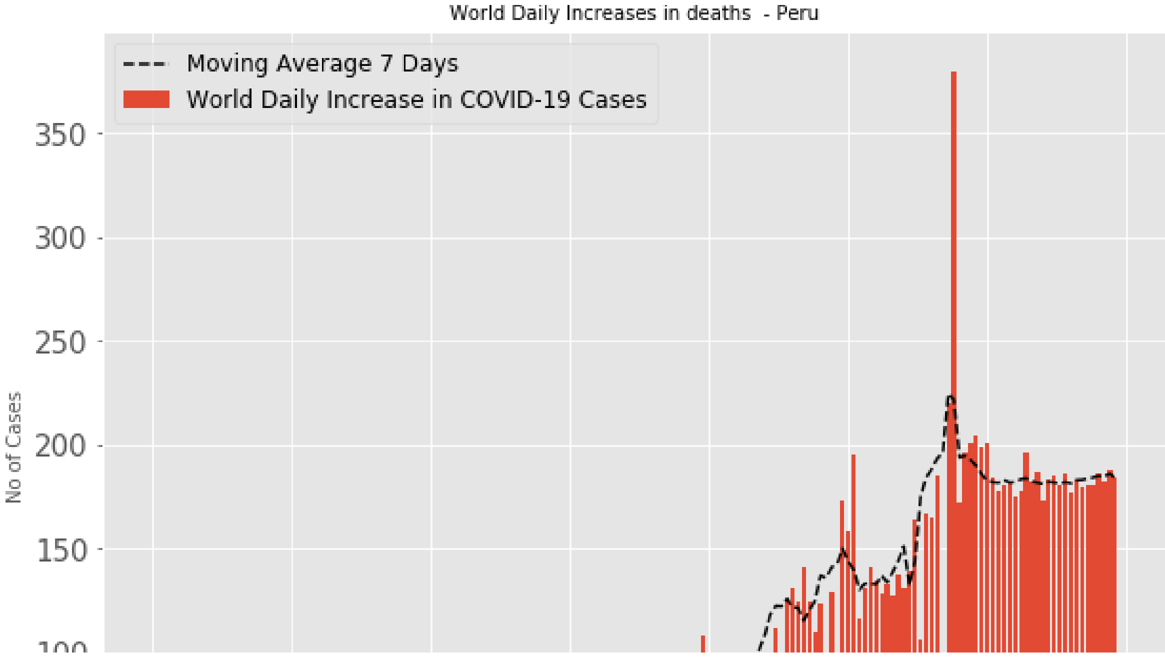

World Daily Increases in recovery - Russia
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
12000
10000
8000
6000
4000
2000
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

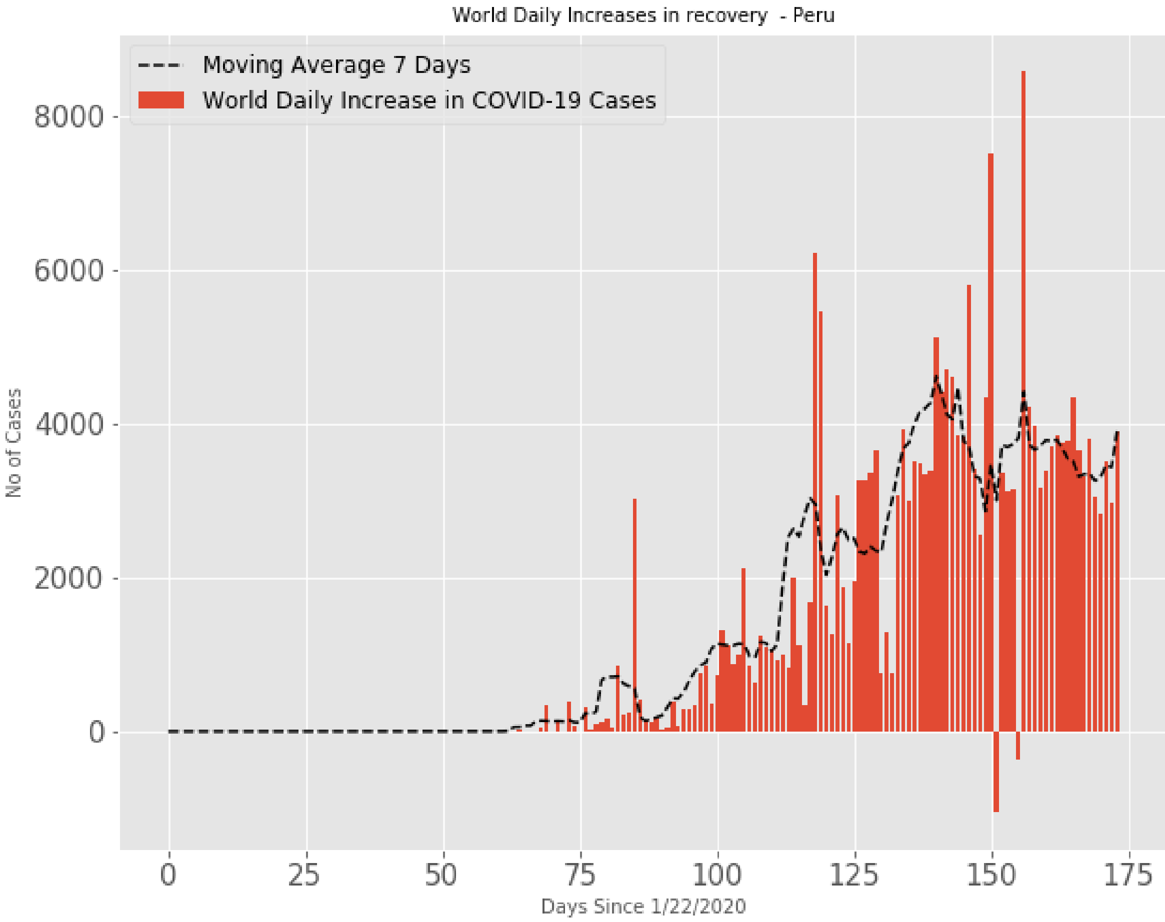
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Peru-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*
```



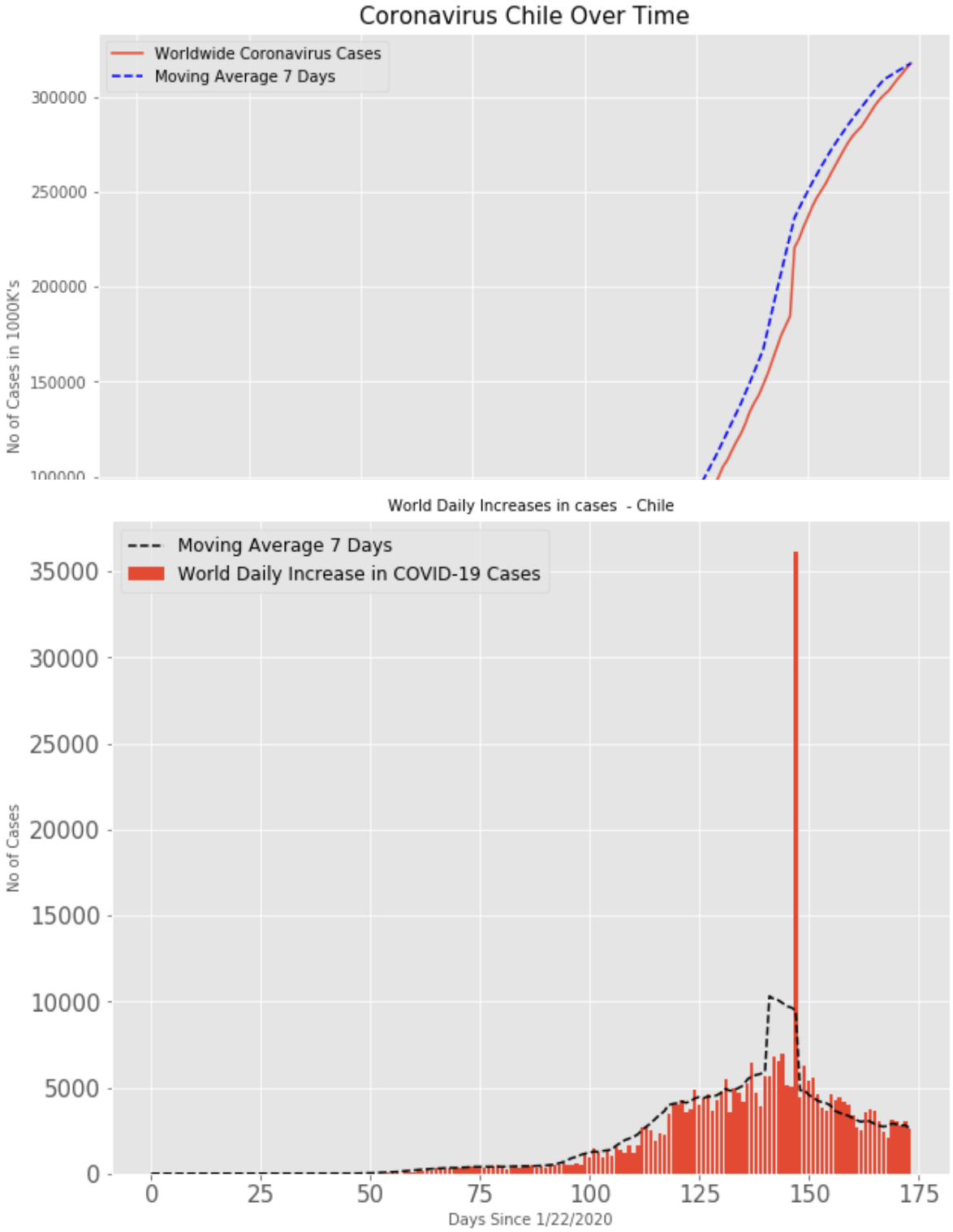
World Daily Increases in cases - Peru

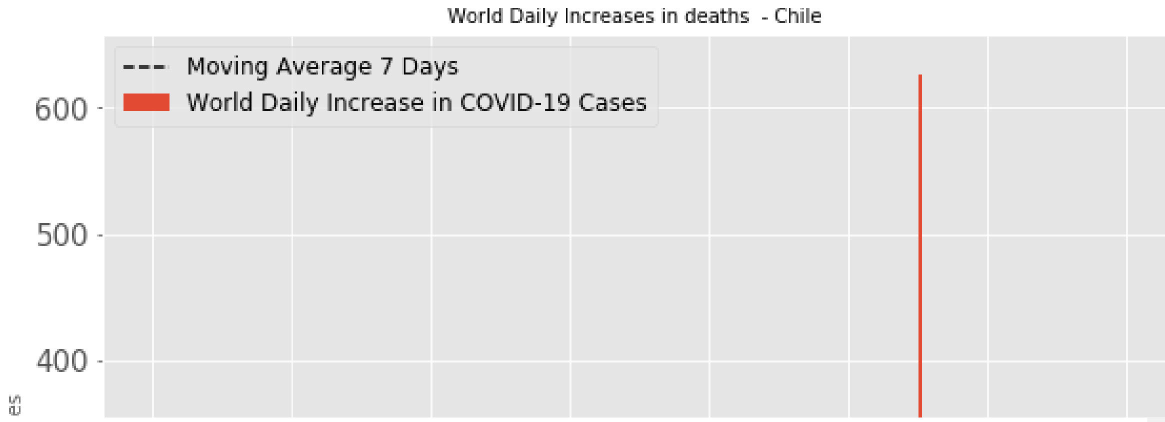


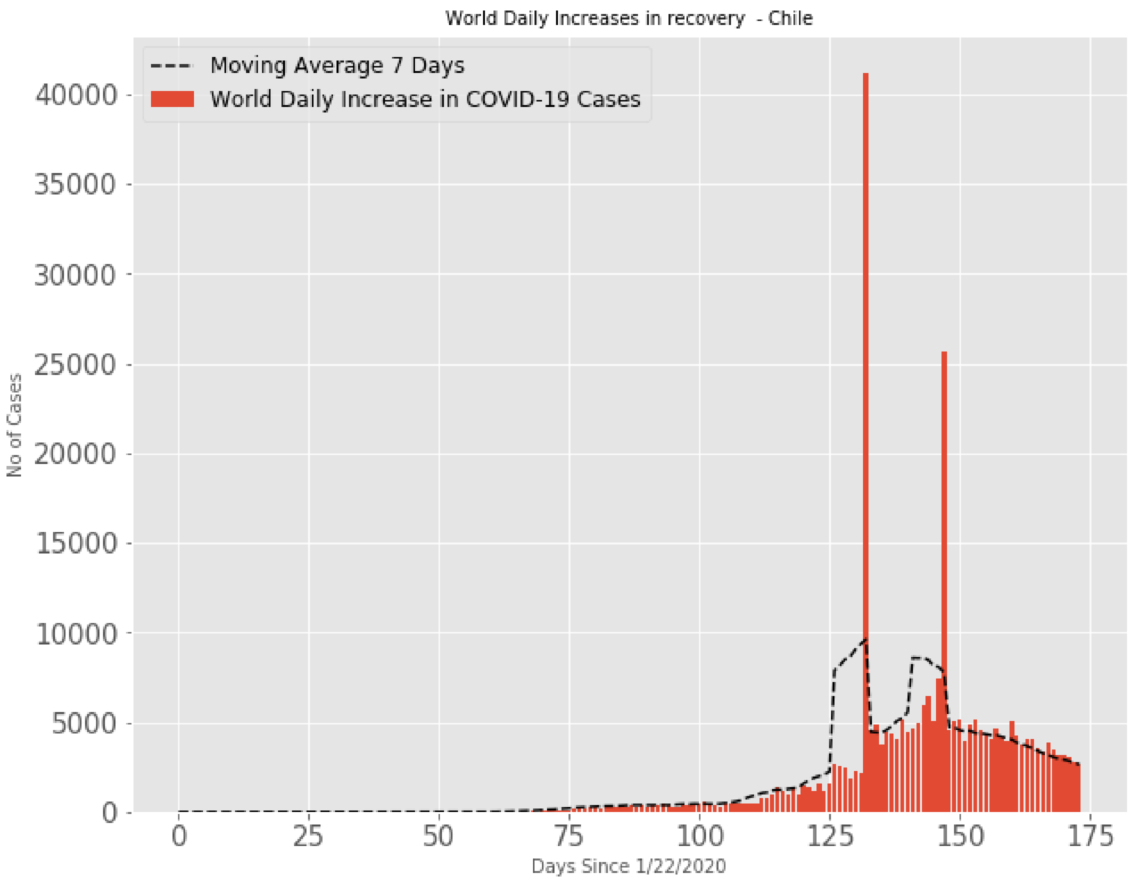

World Daily Increases in deaths - Peru
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
350
300
250
200
150


World Daily Increases in recovery - Peru
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
8000
6000
4000
2000
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Chile-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
*-*-*-*-*
```

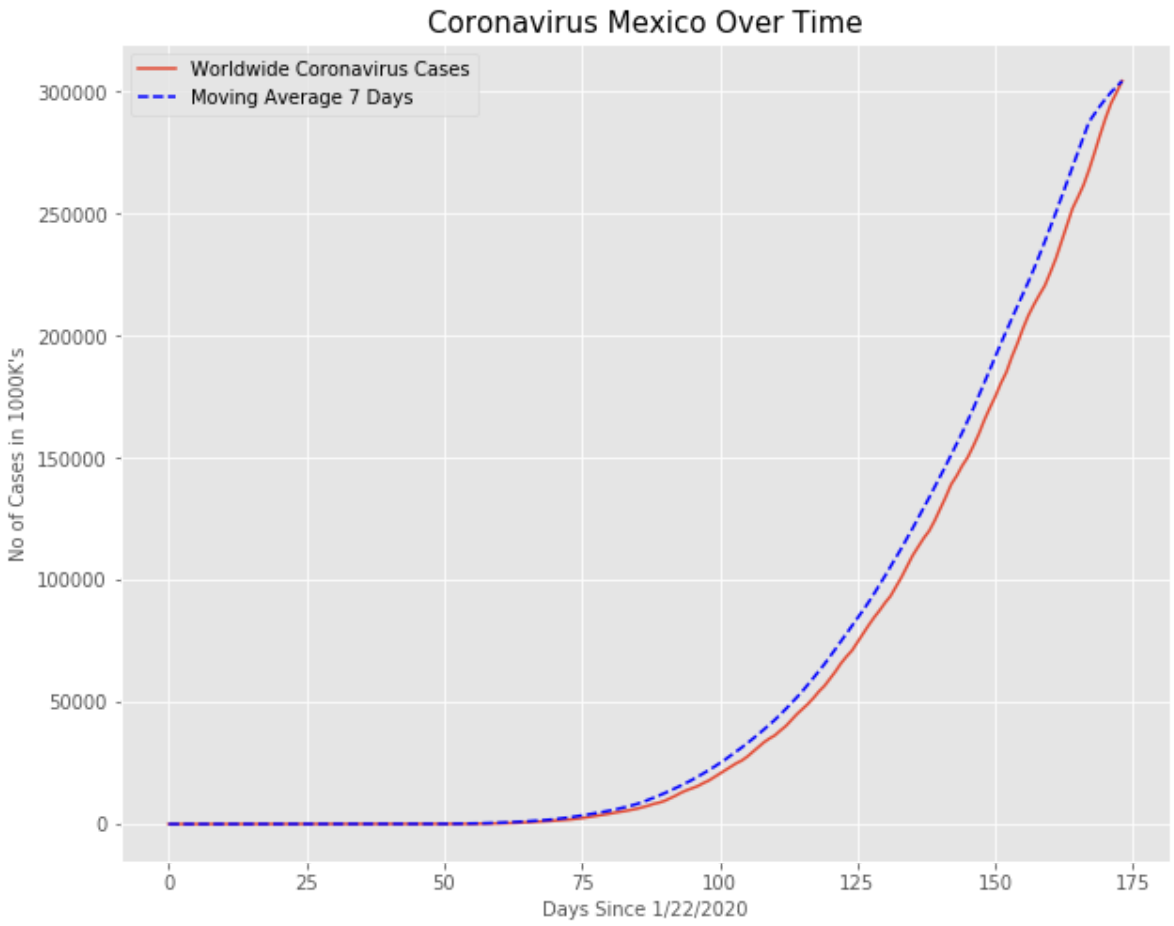

Coronavirus Chile Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
No of Cases in 1000K's
300000
250000
200000
150000
100000
World Daily Increases in cases - Chile
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
35000
30000
25000
20000
15000
10000
5000
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

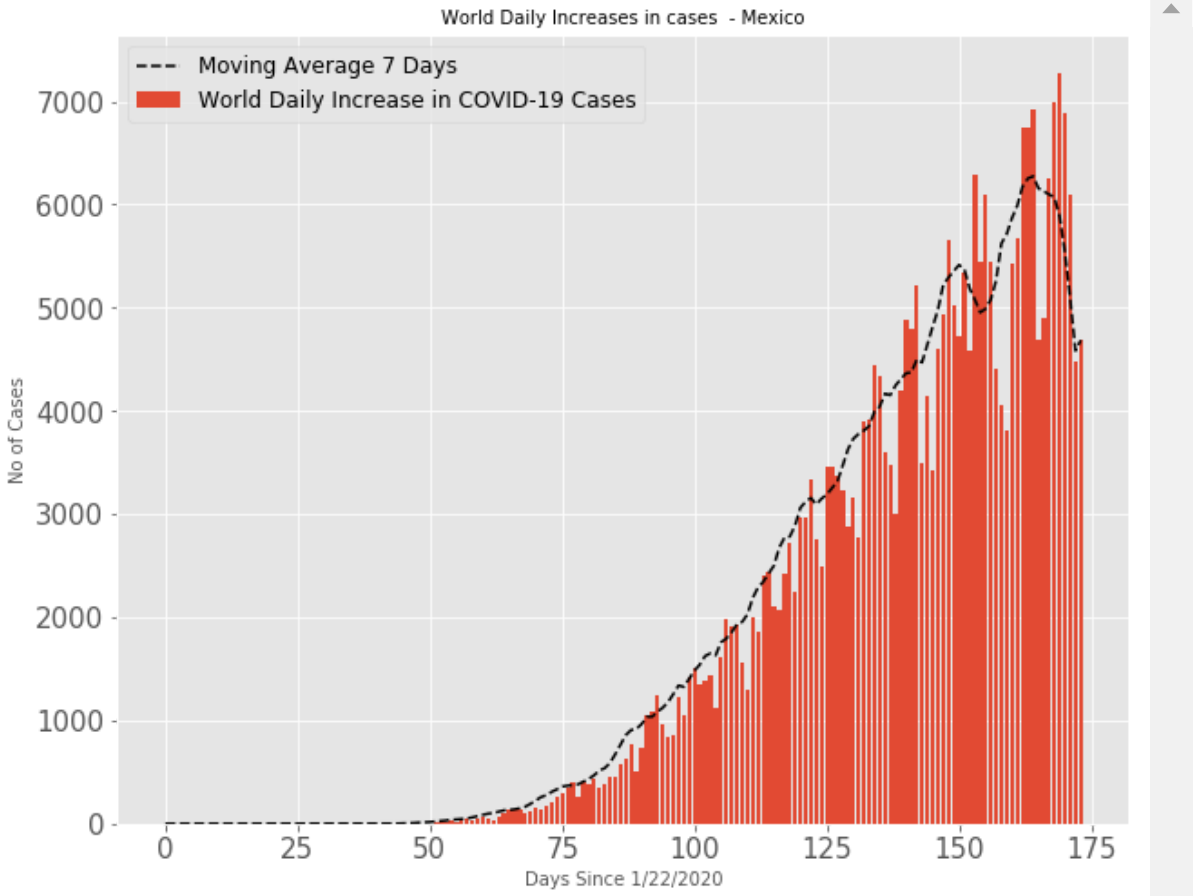

World Daily Increases in deaths - Chile
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
600
500
400
es

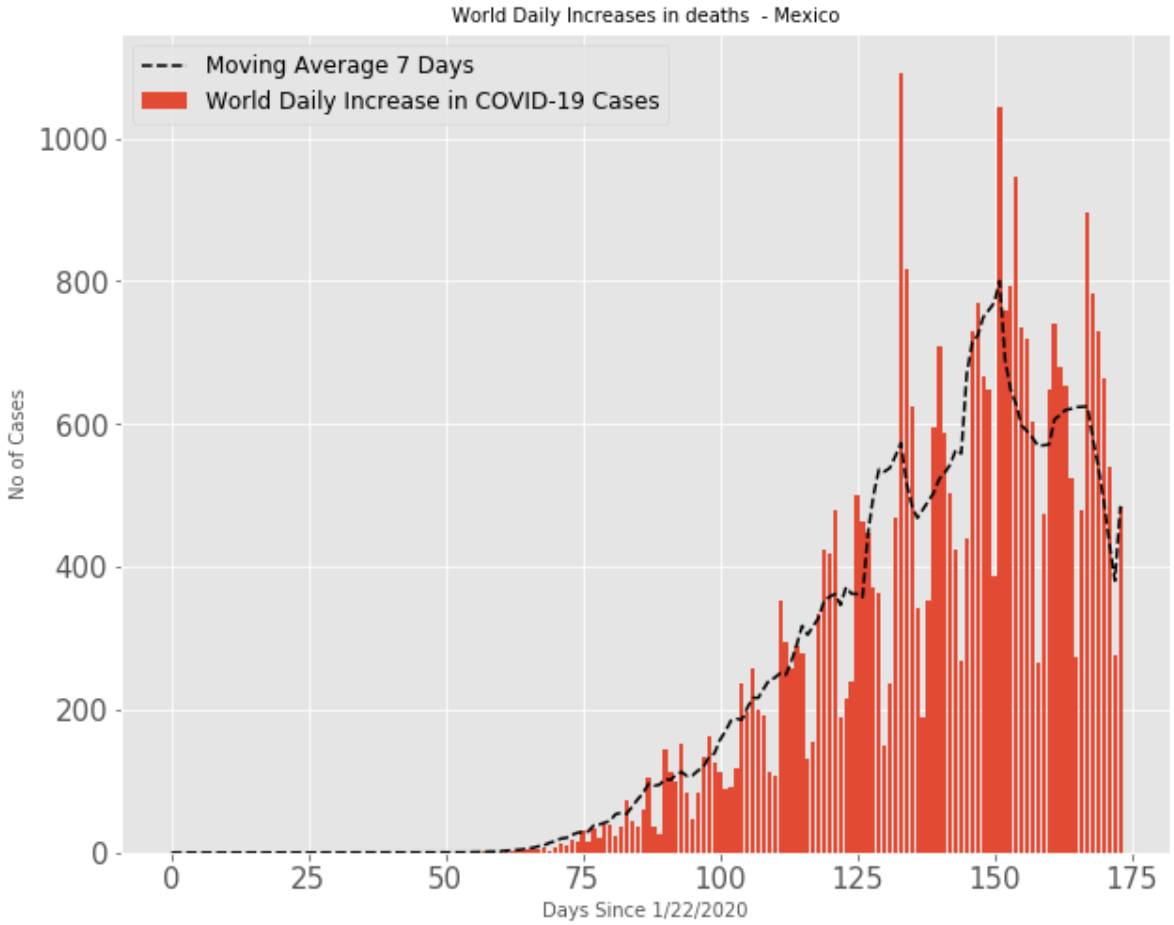

World Daily Increases in recovery - Chile
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
40000
35000
30000
25000
20000
15000
10000
5000
0
No of Cases
0
25
50
75
100
125
150
175
Days Since 1/22/2020

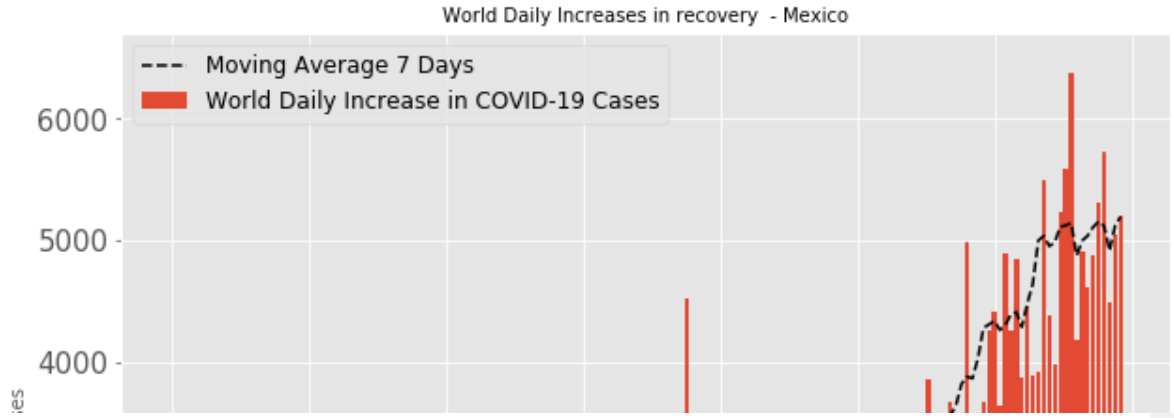
-*Mexico-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

-*-*-*-*-*

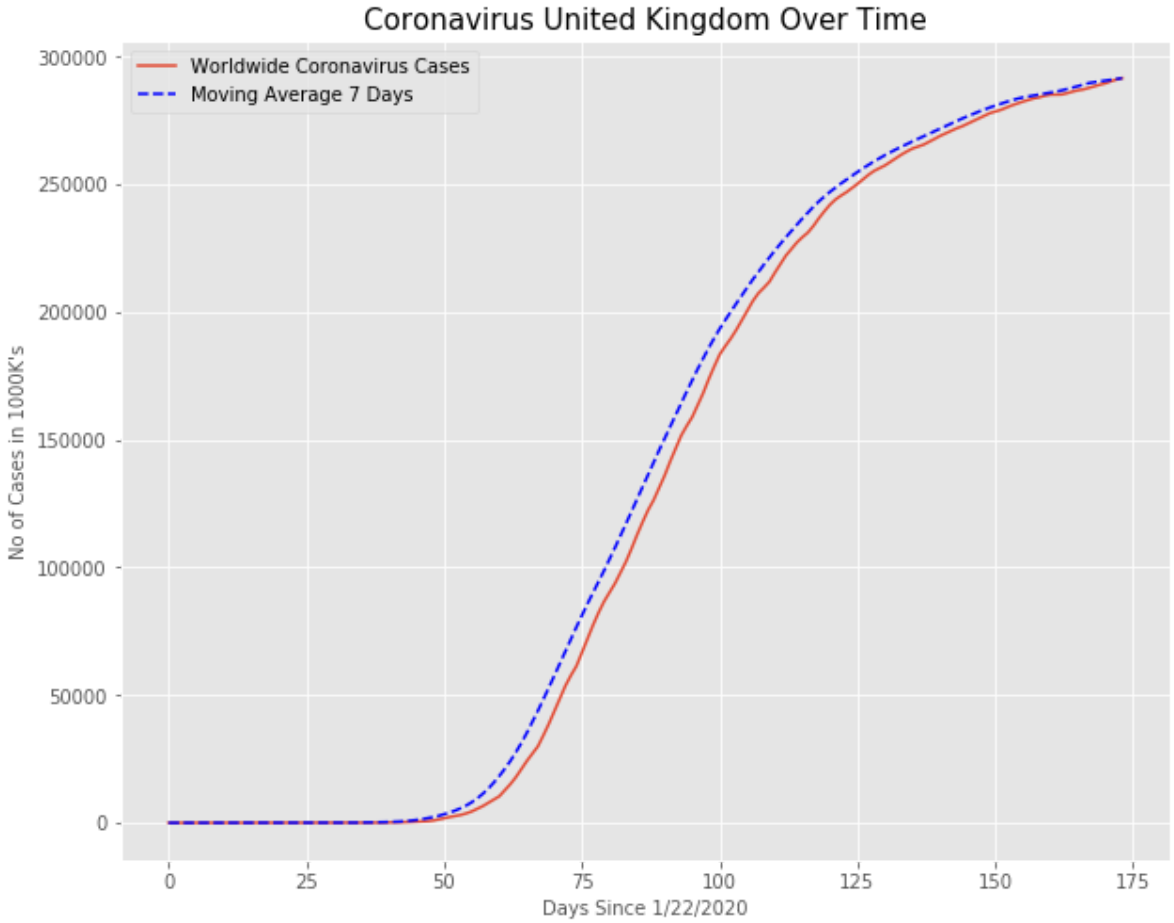

Coronavirus Mexico Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
No of Cases in 1000K's
300000
250000
200000
150000
100000
50000
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

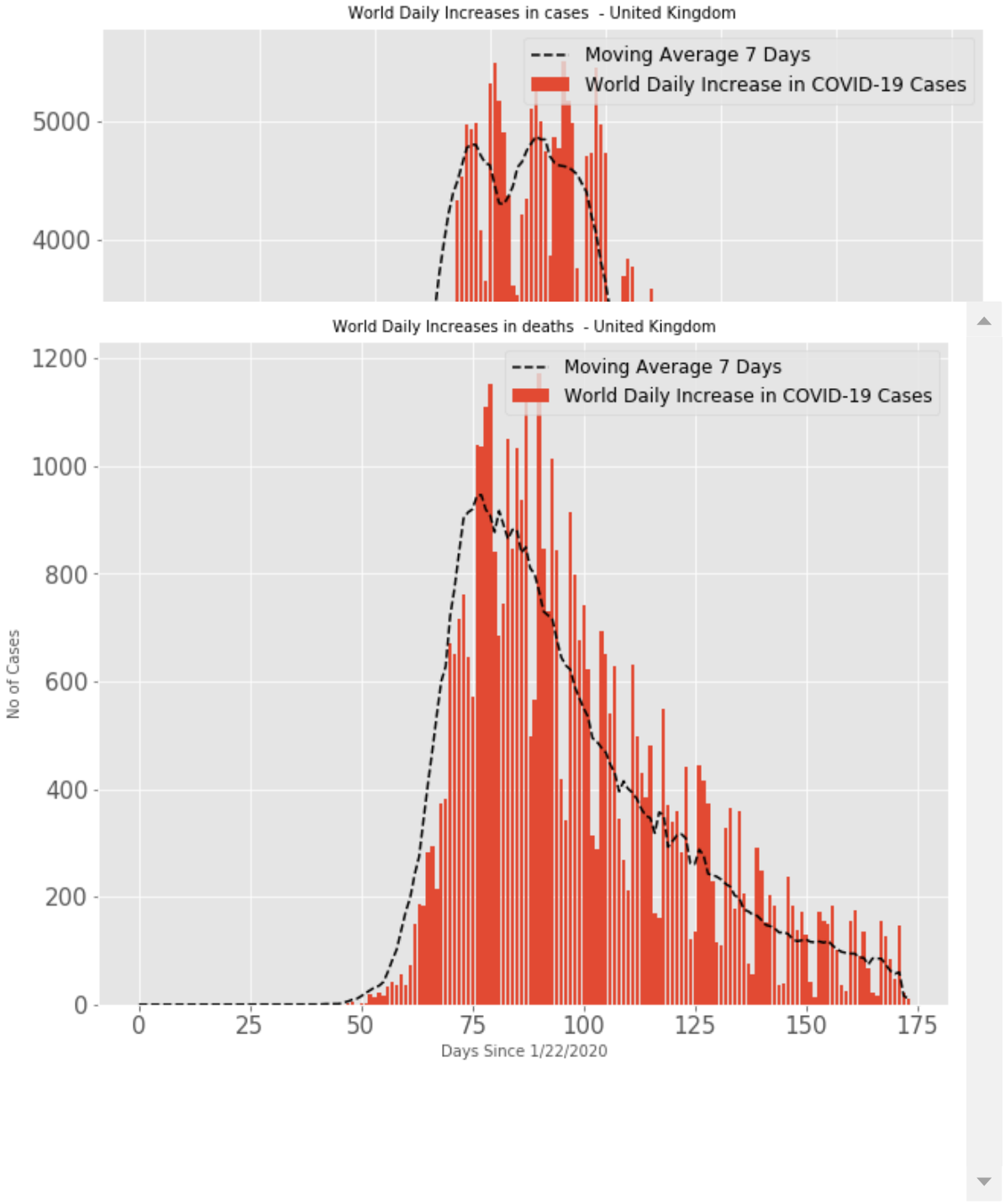

World Daily Increases in cases - Mexico
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
7000
6000
5000
4000
3000
2000
1000
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

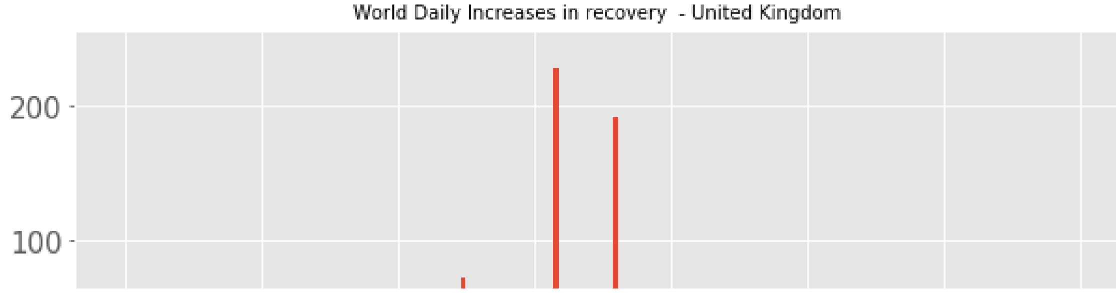

World Daily Increases in deaths - Mexico
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
0
200
400
600
800
1000
0
25
50
75
100
125
150
175
Days Since 1/22/2020



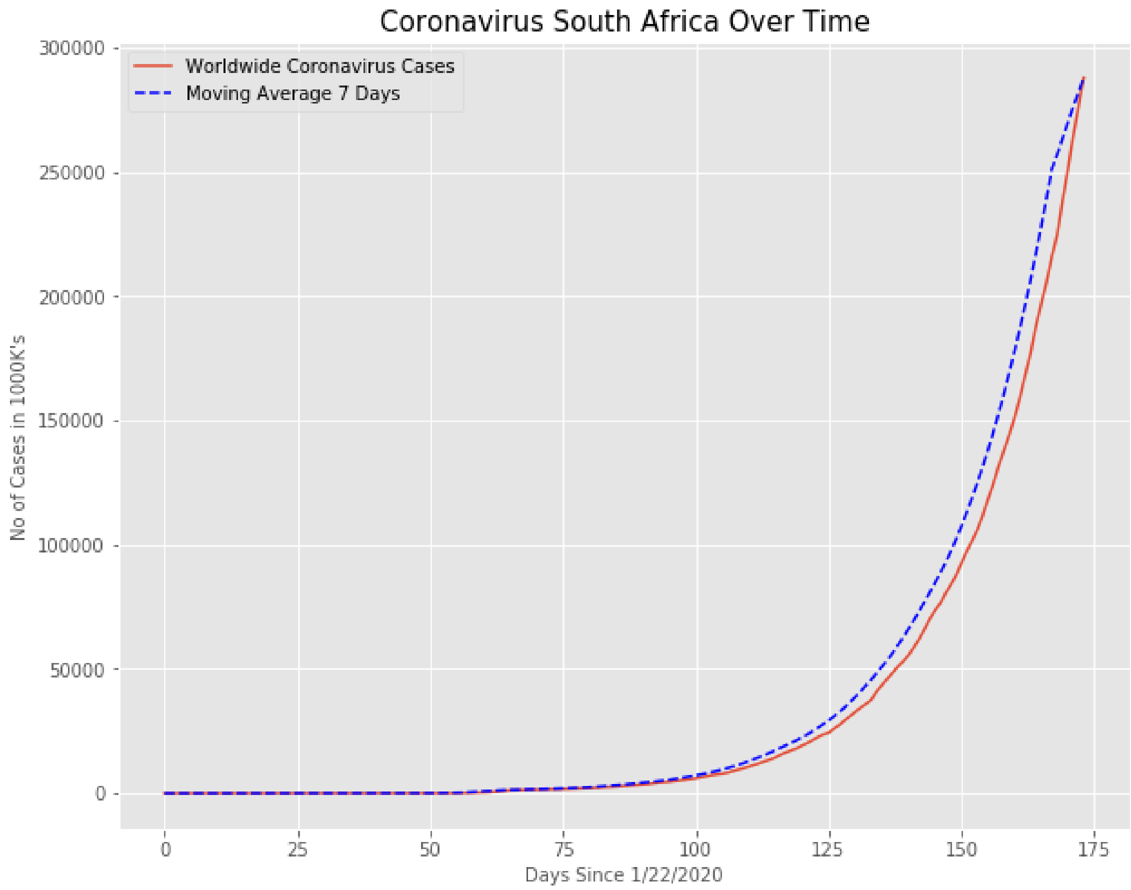

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*United Kingdom-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*-*-*-*-*-*-*
```

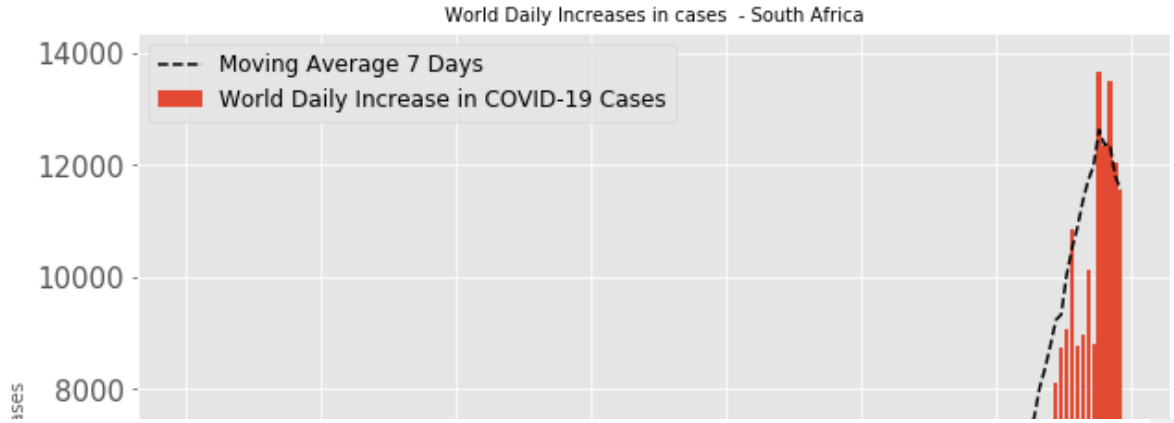


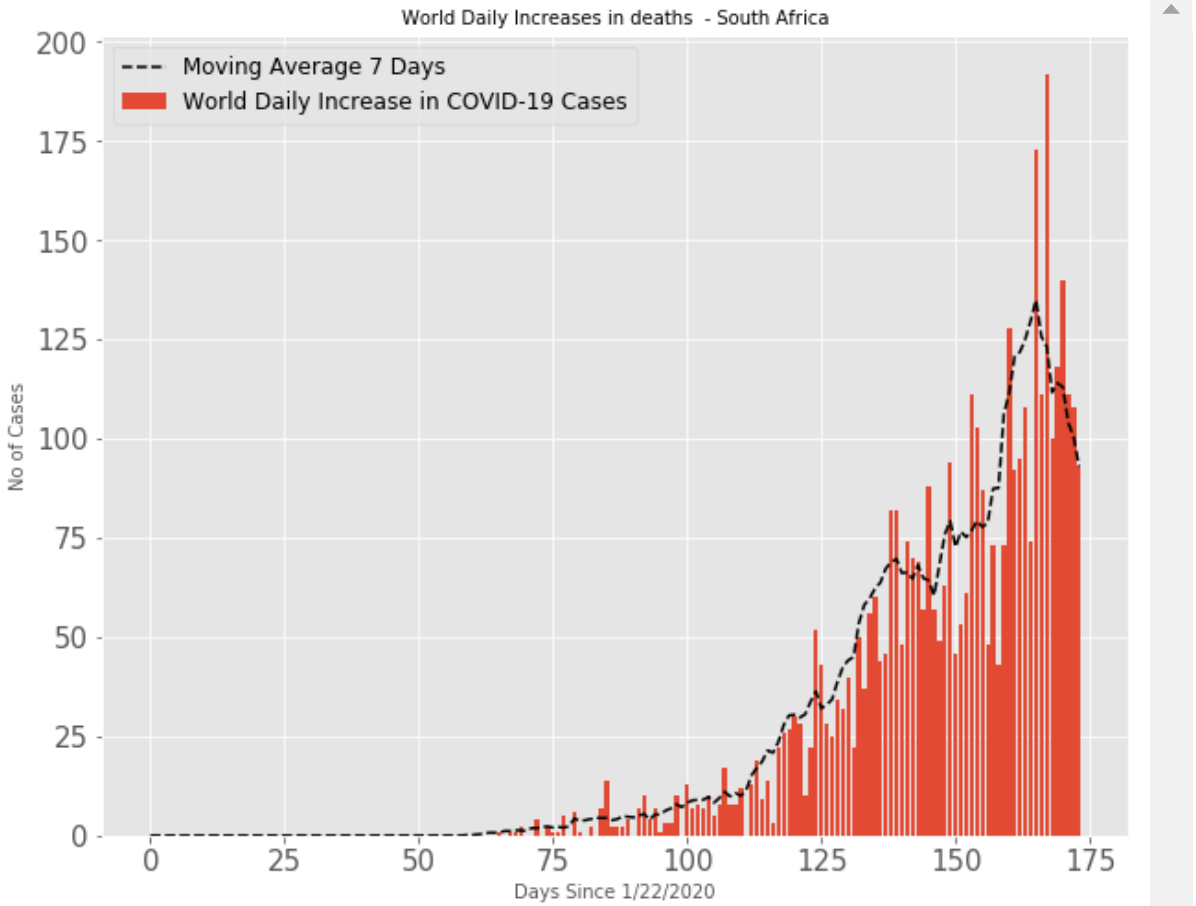

World Daily Increases in cases - United Kingdom
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
5000
4000
World Daily Increases in deaths - United Kingdom
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
1200
1000
800
600
400
200
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

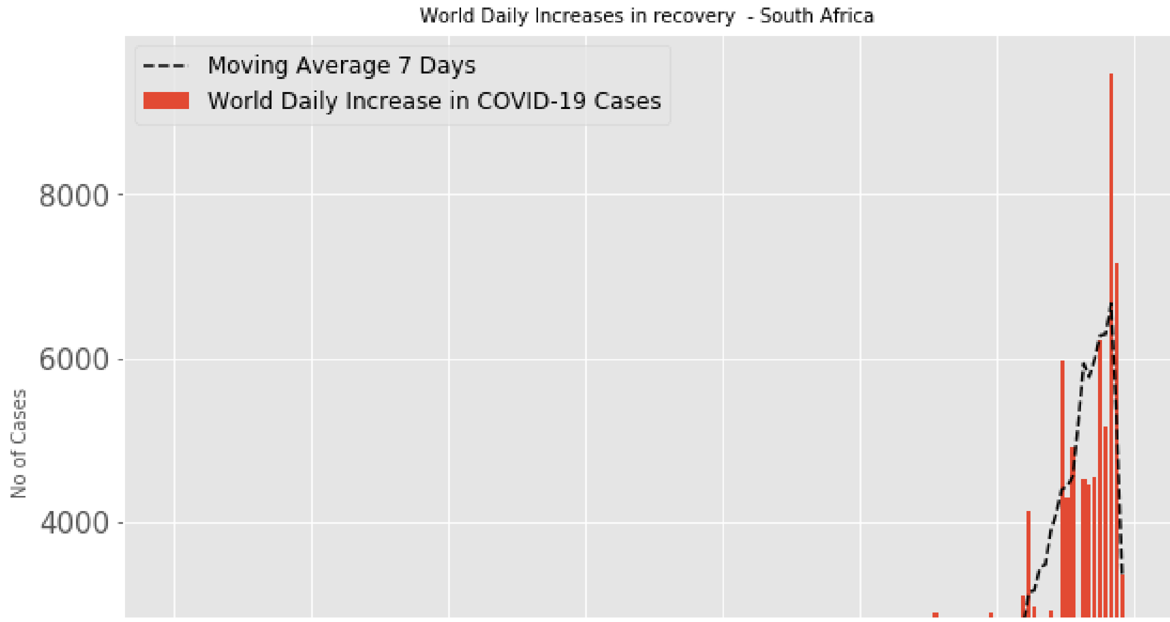


-*South Africa-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*-*-*-*

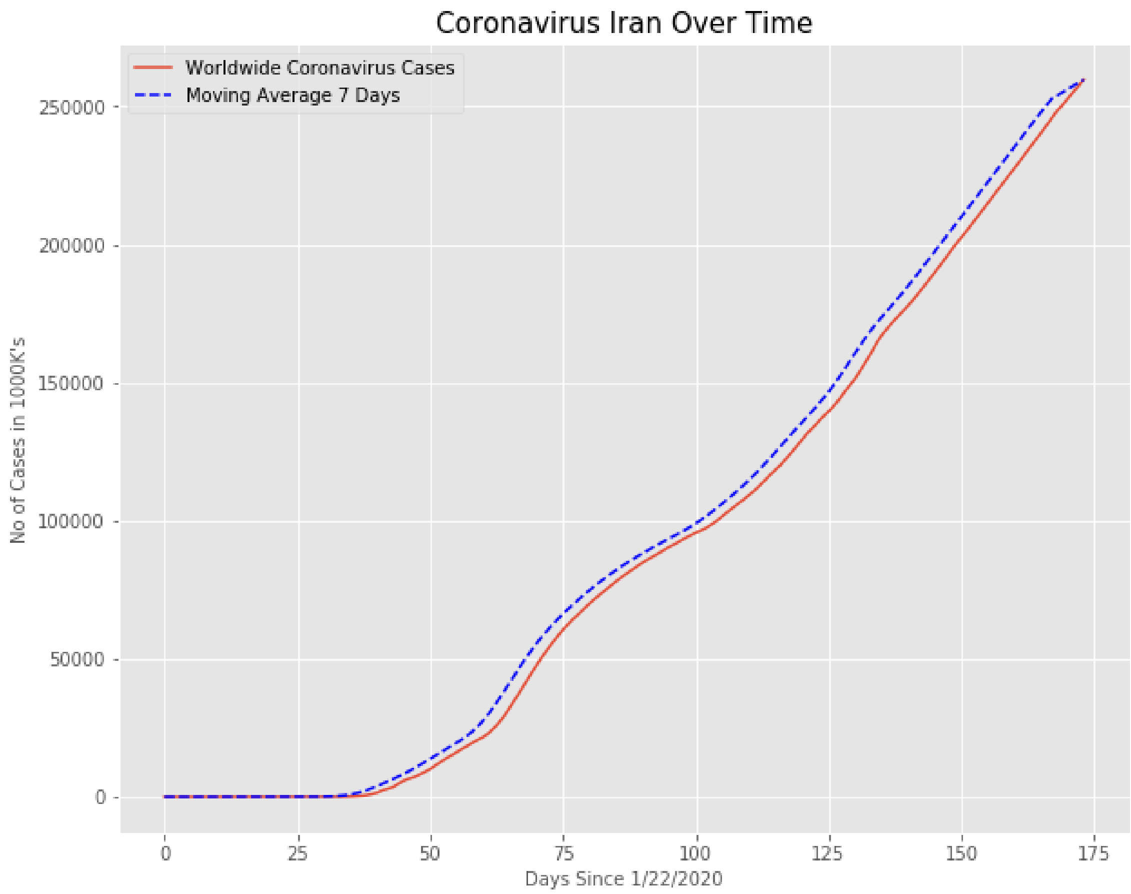


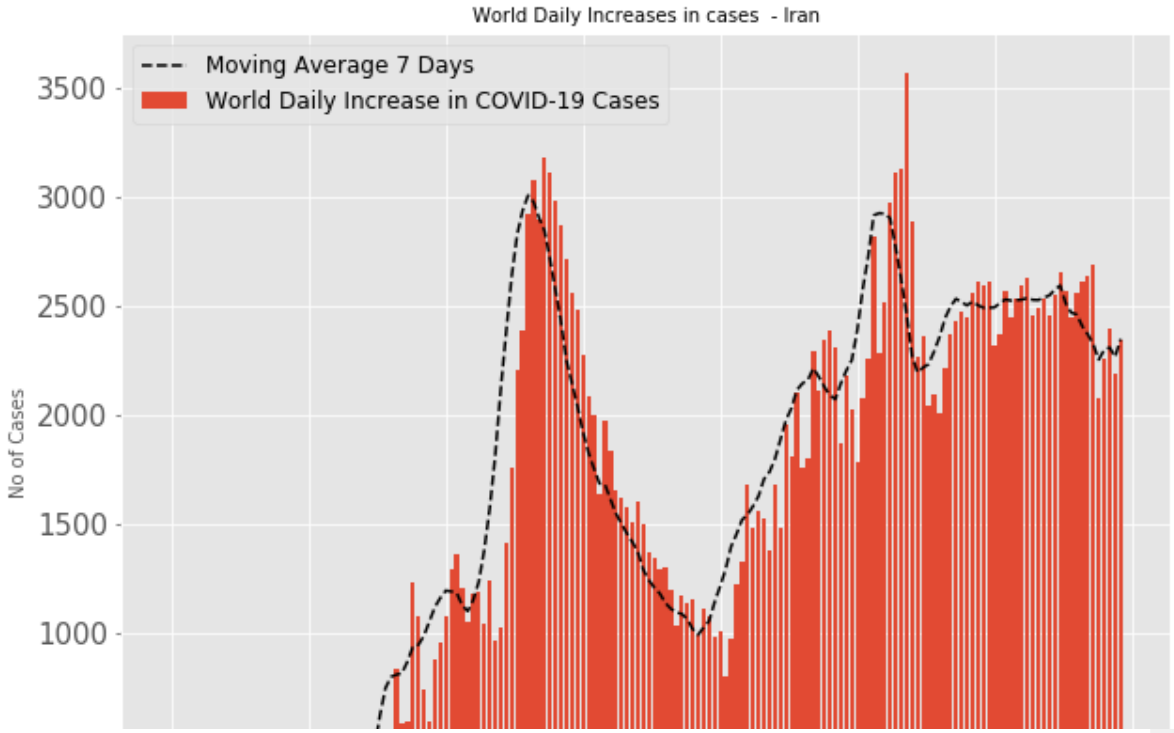

World Daily Increases in cases - South Africa
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
14000
12000
10000
8000

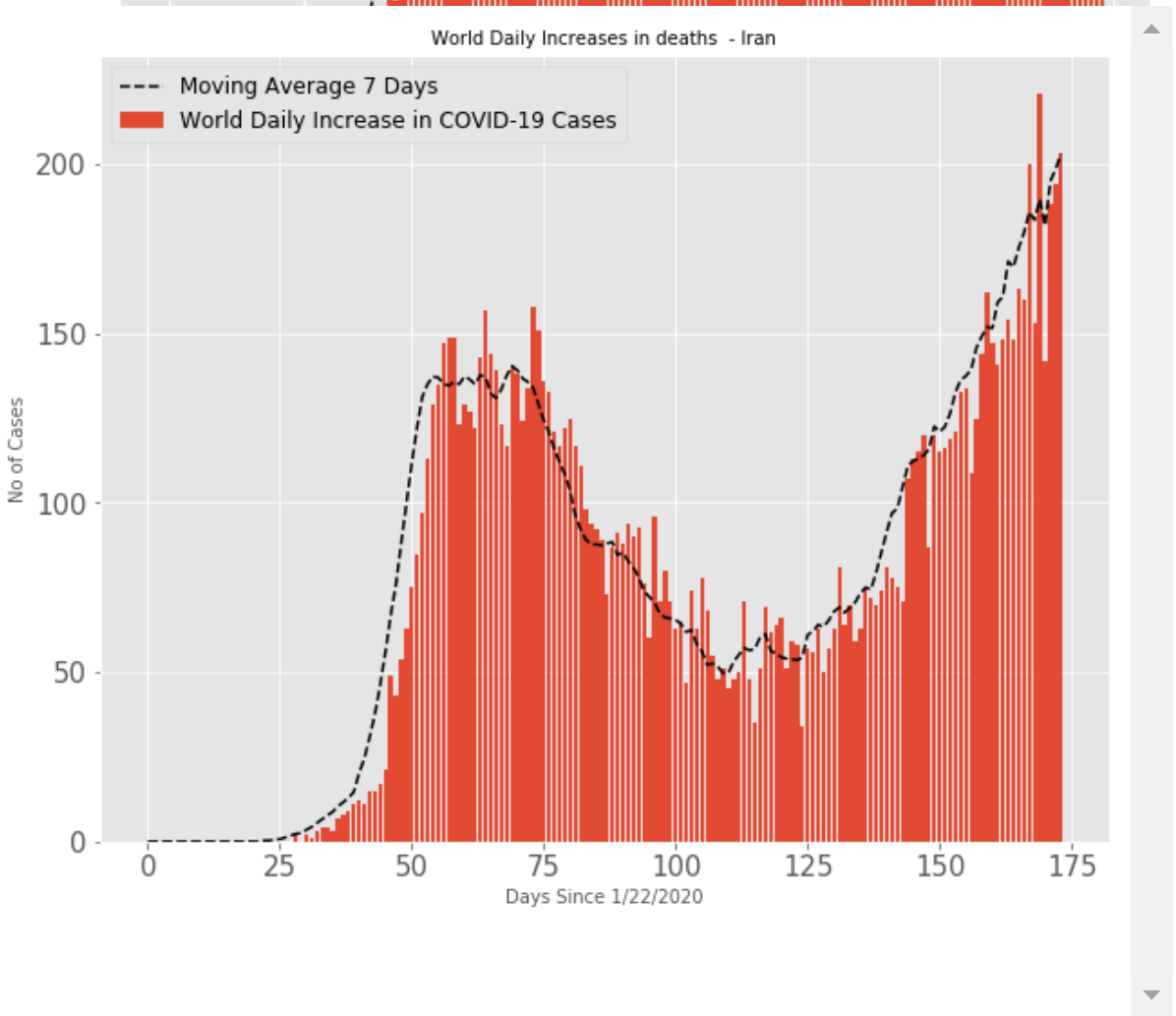

World Daily Increases in deaths - South Africa
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
200
175
150
125
100
75
50
25
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

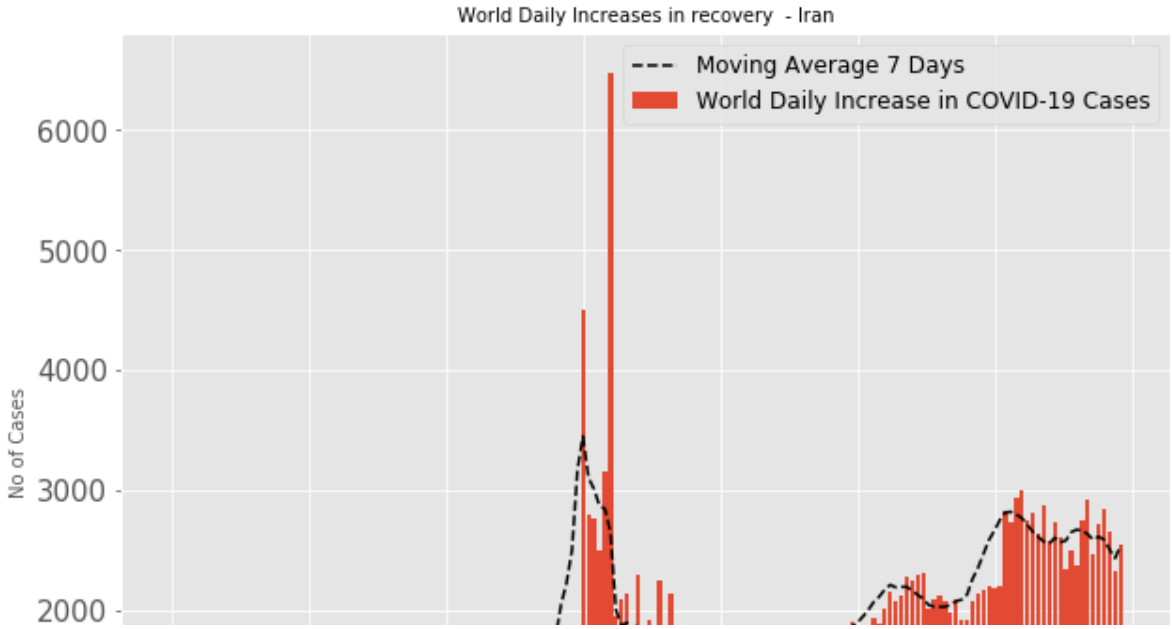

World Daily Increases in recovery - South Africa
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
8000
6000
4000
No of Cases

-*Iran-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*


Coronavirus Iran Over Time
Worldwide Coronavirus Cases
Moving Average 7 Days
250000
200000
150000
100000
50000
0
No of Cases in 1000K's
0
25
50
75
100
125
150
175
Days Since 1/22/2020


World Daily Increases in cases - Iran
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
3500
3000
2500
2000
1500
1000


World Daily Increases in deaths - Iran
Moving Average 7 Days
World Daily Increase in COVID-19 Cases
No of Cases
200
150
100
50
0
0
25
50
75
100
125
150
175
Days Since 1/22/2020

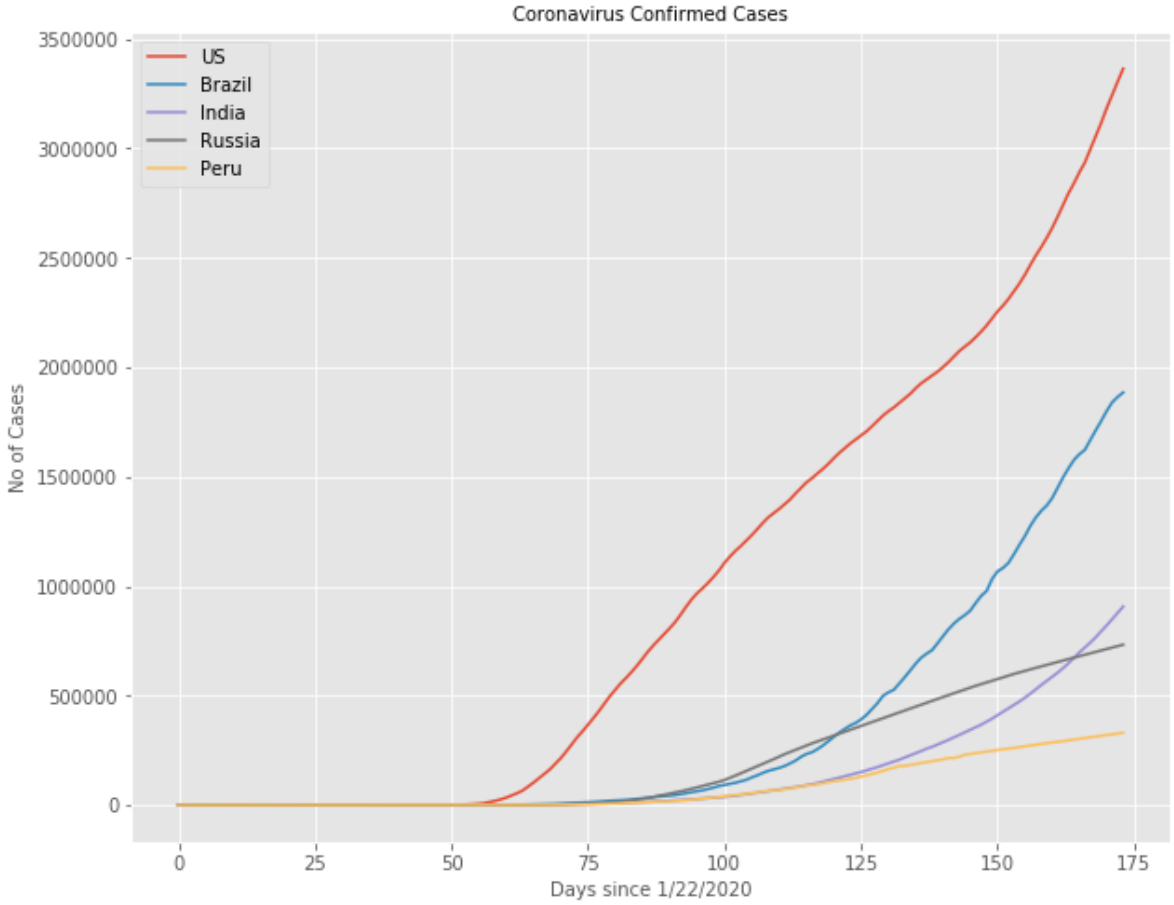


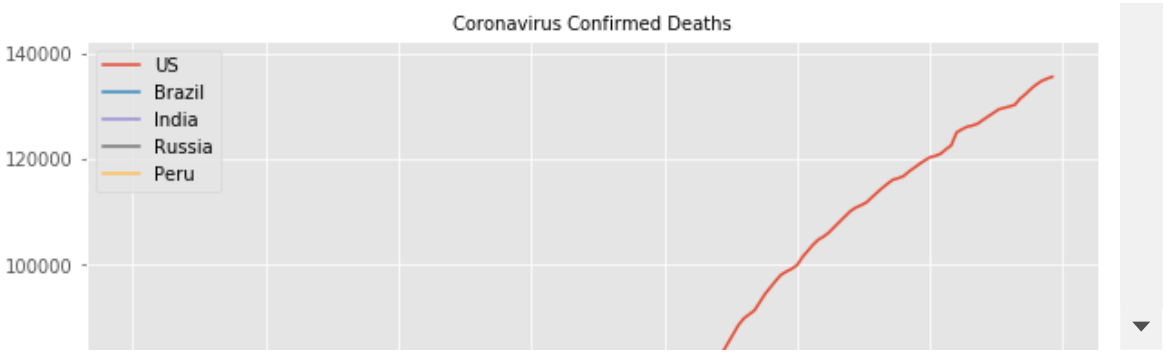
Comparison between top Countries

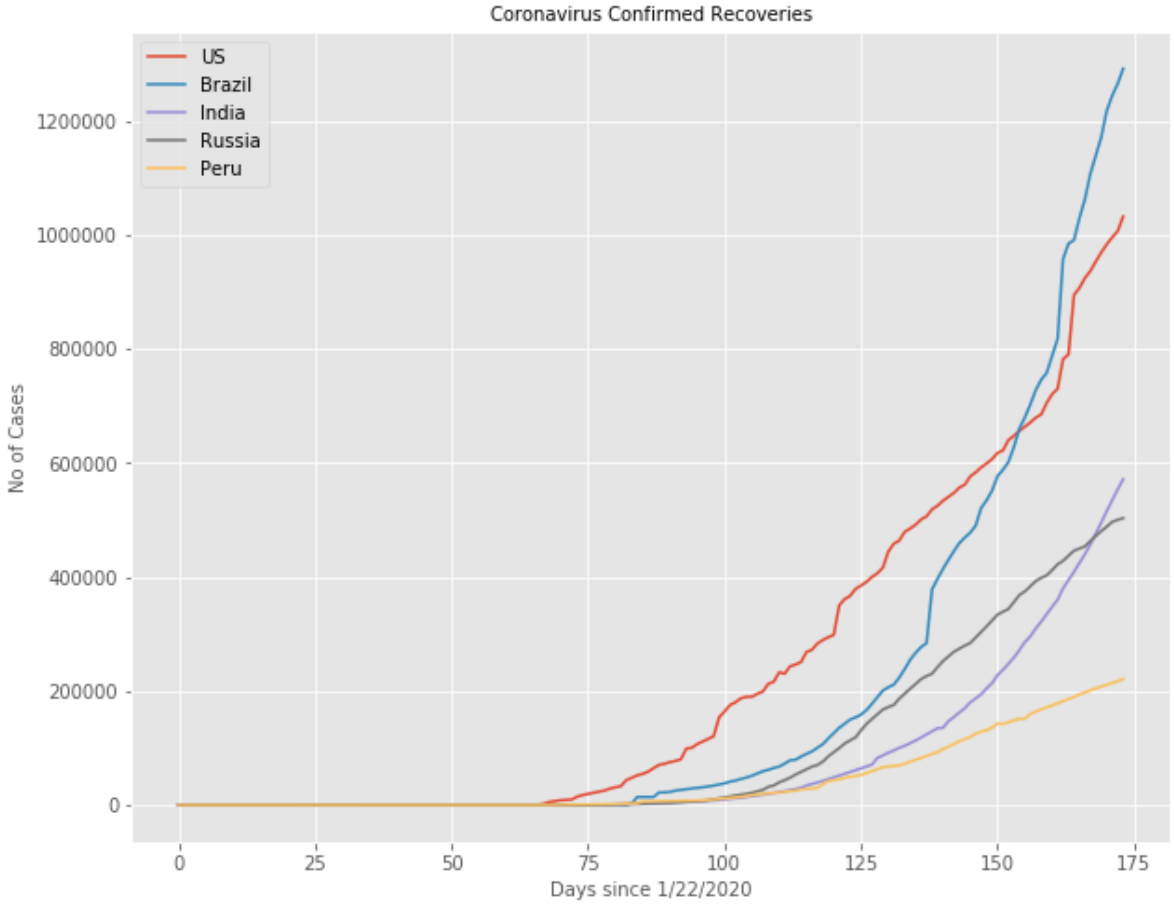
In [29]:

```
compare_countries = countries[:5]
graph_name = ['Coronavirus Confirmed Cases', 'Coronavirus Confirmed Deaths', 'Coronavirus C

for num in range(3):
    plt.figure(figsize=(10, 8))
    for country in compare_countries:
        plt.plot(get_country_info(country)[num])
    plt.legend(compare_countries, prop={'size': 10})
    plt.xlabel('Days since 1/22/2020', size=10)
    plt.ylabel('No of Cases', size=10)
    plt.title(graph_name[num], size=10)
    plt.xticks(size=10)
    plt.yticks(size=10)
    plt.show()
```







Analysis of latest data

In [30]:

```
latest_data.head()
```

Out[30]:

	FIPS	Admin2	Province_State	Country_Region	Last_Update	Lat	Long_	C
0	45001.0	Abbeville	South Carolina	US	2020-07-13 04:43:04	34.223334	-82.461707	
1	22001.0	Acadia	Louisiana	US	2020-07-13 04:43:04	30.295065	-92.414197	
2	51001.0	Accomack	Virginia	US	2020-07-13 04:43:04	37.767072	-75.632346	
3	16001.0	Ada	Idaho	US	2020-07-13 04:43:04	43.452658	-116.241552	
4	19001.0	Adair	Iowa	US	2020-07-13 04:43:04	41.330756	-94.471059	

Country data frame sorted by country with increase in cases

In [31]:

```
country_df = latest_data.groupby('Country_Region').sum().sort_values(by = 'Confirmed',ascen
```

In [32]:

```
country_df.reset_index(inplace =True)
```

In [33]:

```
country_df['Mortality_rate'] = country_df.Deaths/country_df.Confirmed
```

In [34]:

```
country_df.head(20).style.background_gradient(cmap='PuBu')
```

Out[34]:

	Country_Region	Confirmed	Deaths	Recovered	Active	Mortality_rate
0	US	3304942	135205	1006326	2208772.000000	0.040910
1	Brazil	1864681	72100	1264843	527738.000000	0.038666
2	India	878254	23174	553471	301609.000000	0.026386
3	Russia	726036	11318	500208	214510.000000	0.015589
4	Peru	326326	11870	217111	97345.000000	0.036375
5	Chile	315041	6979	283902	24160.000000	0.022153
6	Mexico	299750	35006	234905	29839.000000	0.116784
7	United Kingdom	291154	44904	1378	244872.000000	0.154228
8	South Africa	276242	4079	134874	137289.000000	0.014766
9	Iran	257303	12829	219993	24481.000000	0.049860
10	Spain	253908	28403	150376	75130.000000	0.111863
11	Pakistan	251625	5266	161917	84442.000000	0.020928
12	Italy	243061	34954	194928	13179.000000	0.143808
13	Saudi Arabia	232259	2223	167138	62898.000000	0.009571
14	Turkey	212993	5363	194515	13115.000000	0.025179
15	France	208015	30007	78513	99495.000000	0.144254
16	Germany	199919	9071	184414	6434.000000	0.045373
17	Bangladesh	183795	2352	93614	87829.000000	0.012797
18	Colombia	145362	5426	61186	78750.000000	0.037327
19	Canada	109348	8829	72954	27566.000000	0.080742

Province_State data frame sorted by Province_State with increase in cases

In [35]:

```
province_df = latest_data.groupby('Province_State').sum().sort_values(by = 'Confirmed',asce
province_df.reset_index(inplace =True)
```

In [36]:

```
province_df[province_df['Confirmed'] == 0].index
```

Out[36]:

```
Int64Index([547], dtype='int64')
```

In [37]:

```
province_df.drop(province_df[province_df['Confirmed'] == 0].index,inplace=True)
```

In [38]:

```
province_df.tail()
```

Out[38]:

	Province_State	Confirmed	Deaths	Recovered	Active
542	Northwest Territories	5	0	5	0.0
543	Anguilla	3	0	3	0.0
544	Saint Pierre and Miquelon	2	0	1	1.0
545	Tibet	1	0	1	0.0
546	Vichada	1	0	1	0.0

In [39]:

```
province_df['Mortality_rate'] = province_df.Deaths/province_df.Confirmed
```

In [40]:

```
province_df.head(20).style.background_gradient(cmap='PuBu')
```

Out[40]:

	Province_State	Confirmed	Deaths	Recovered	Active	Mortality_rate
0	New York	401706	32350	0	369356.000000	0.080532
1	Sao Paulo	371997	17848	220974	133175.000000	0.047979
2	California	324543	7051	0	317492.000000	0.021726
3	Florida	269811	4242	0	265569.000000	0.015722
4	Texas	262762	3216	0	259546.000000	0.012239
5	Maharashtra	254427	10289	140325	103813.000000	0.040440
6	England	249510	40234	0	209276.000000	0.161252
7	Metropolitana	238360	5814	219896	12650.000000	0.024392
8	Moscow	229357	4143	164560	60654.000000	0.018064
9	Lima	175631	5530	0	170101.000000	0.031486
10	New Jersey	175298	15525	0	159773.000000	0.088563
11	Illinois	155048	7388	0	147660.000000	0.047650
12	Tamil Nadu	138470	1966	89532	46972.000000	0.014198
13	Ceara	136785	6868	110224	19693.000000	0.050210
14	Rio de Janeiro	129684	11415	110061	8208.000000	0.088022
15	Para	125714	5289	112237	8188.000000	0.042072
16	Arizona	122467	2237	0	120231.000000	0.018266
17	Georgia	116935	3003	0	113932.000000	0.025681
18	Delhi	112494	3371	89968	19155.000000	0.029966
19	Massachusetts	111597	8325	0	103272.000000	0.074599

COUNTRY wise Bar Chart Visualizations for COVID-19

In [41]:

```
# As already sorted with increasing cases we can directly take the cases
country_confirmed_cases = list(country_df['Confirmed'])
unique_countries = list(country_df['Country_Region'].unique())
```

In [42]:

```
def plot_bar_graphs(x, y, title):
    plt.figure(figsize=(15, 8))
    plt.barh(x, y, color='blue')
    plt.title(title, size=10)
    plt.xticks(size=10)
    plt.yticks(size=10)
    plt.show()
```

In [43]:

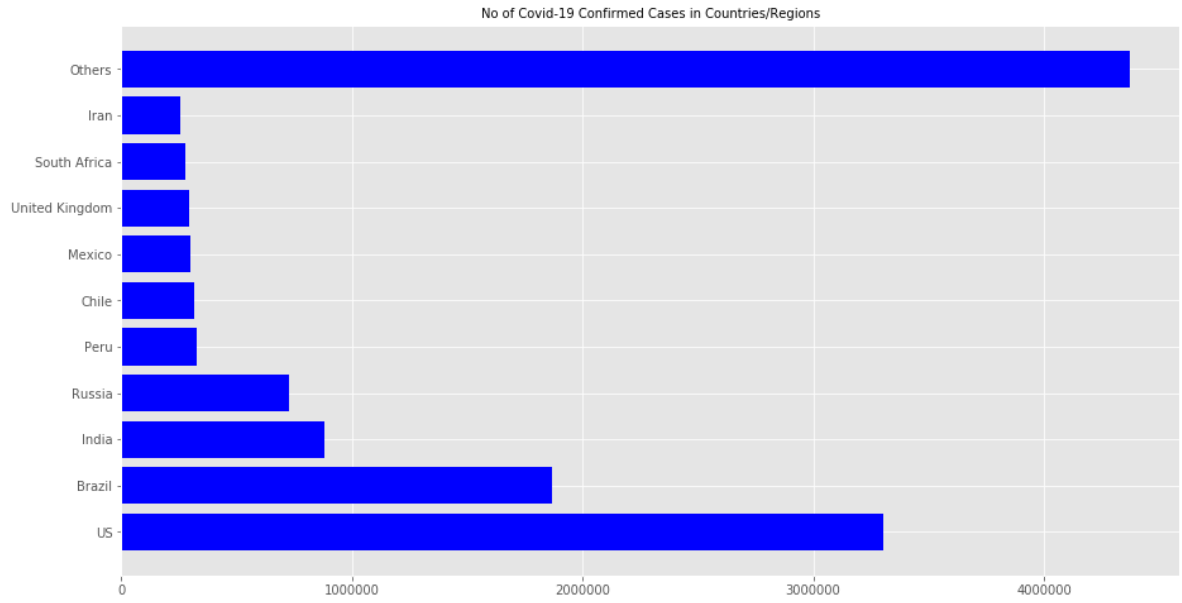
```
# Only show 15 countries with the most confirmed cases, the rest are grouped into the other
visual_unique_countries = []
visual_confirmed_cases = []
others = np.sum(country_confirmed_cases[10:])

for i in range(len(country_confirmed_cases[:10])):
    visual_unique_countries.append(unique_countries[i])
    visual_confirmed_cases.append(country_confirmed_cases[i])

visual_unique_countries.append('Others')
visual_confirmed_cases.append(others)
```

In [44]:

```
plot_bar_graphs(visual_unique_countries, visual_confirmed_cases, 'No of Covid-19 Confirmed
```



PROVINCE wise Bar Chart Visualizations for COVID-19

In [45]:

```
# As already sorted with increasing cases we can directly take the cases
province_confirmed_cases=list(province_df['Confirmed'])
unique_provinces = list(province_df.Province_State.unique())
```

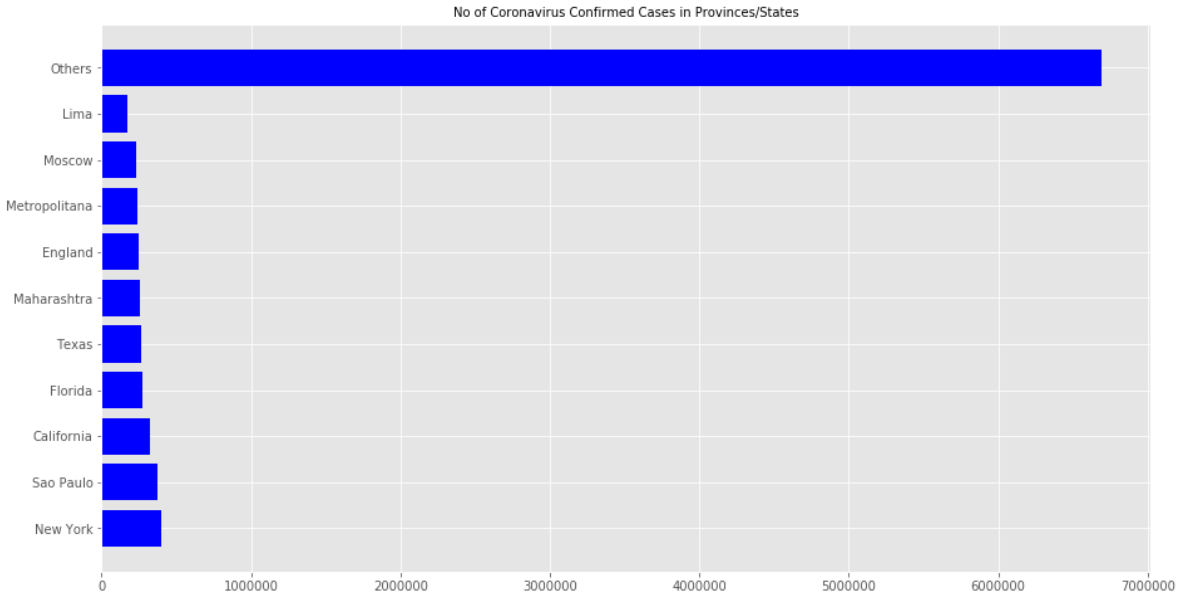
In [46]:

```
visual_unique_provinces = []
visual_confirmed_cases2 = []
others = np.sum(province_confirmed_cases[10:])
for i in range(len(province_confirmed_cases[:10])):
    visual_unique_provinces.append(unique_provinces[i])
    visual_confirmed_cases2.append(province_confirmed_cases[i])

visual_unique_provinces.append('Others')
visual_confirmed_cases2.append(others)
```

In [47]:

```
plot_bar_graphs(visual_unique_provinces, visual_confirmed_cases2, 'No of Coronavirus Confir
```



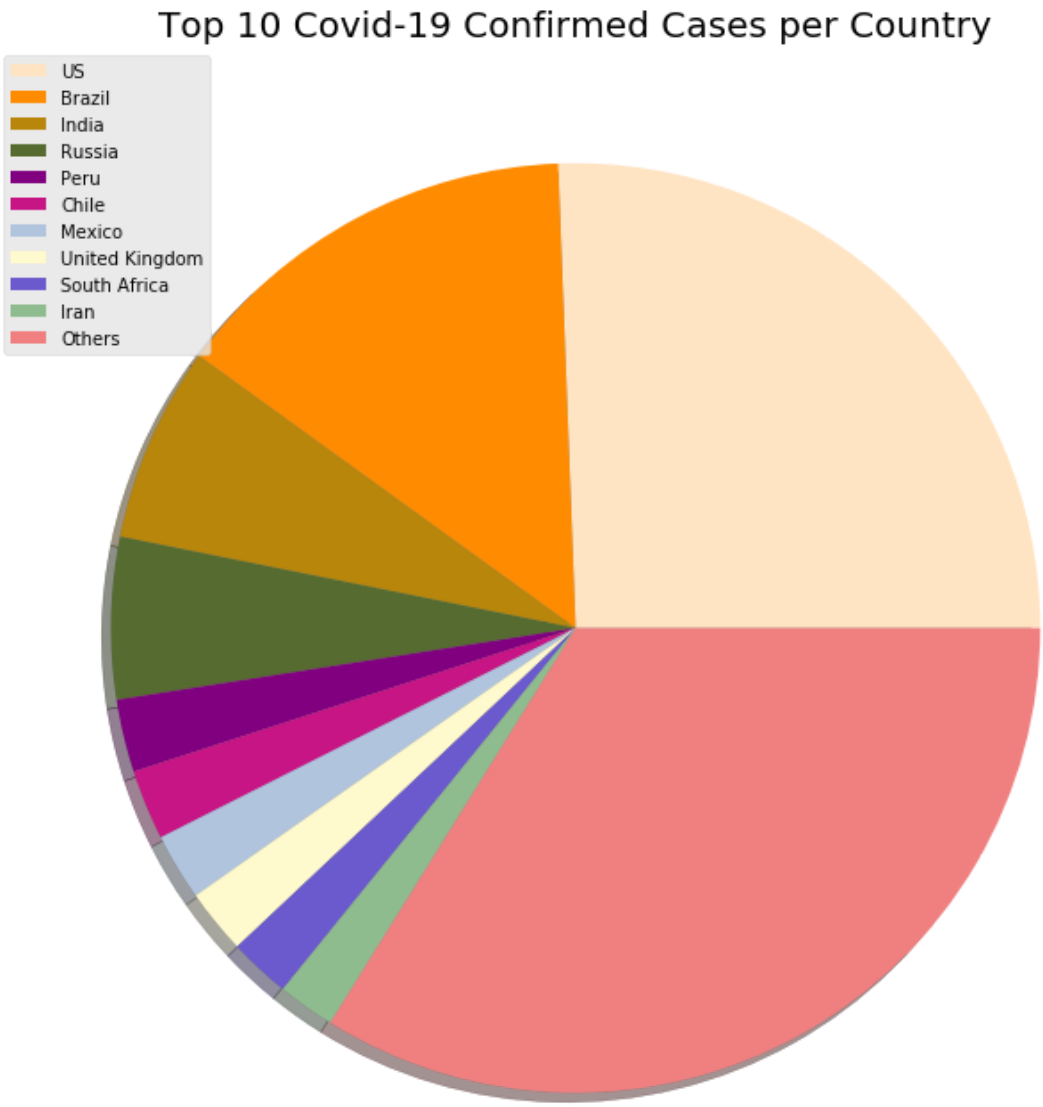
Pie Chart Visualizations for COVID-19

In [48]:

```
def plot_pie_charts(x, y, title):
    # more muted color
    c = ['bisque', 'darkorange', 'darkgoldenrod', 'darkolivegreen', 'purple',
        'mediumvioletred', 'lightsteelblue', 'lemonchiffon', 'slateblue', 'darkseagreen', '
    plt.figure(figsize=(12,12))
    plt.title(title, size=20)
    plt.pie(y,colors= c,shadow=True)
    plt.legend(x, loc='upper left', fontsize=10)
    plt.show()
```

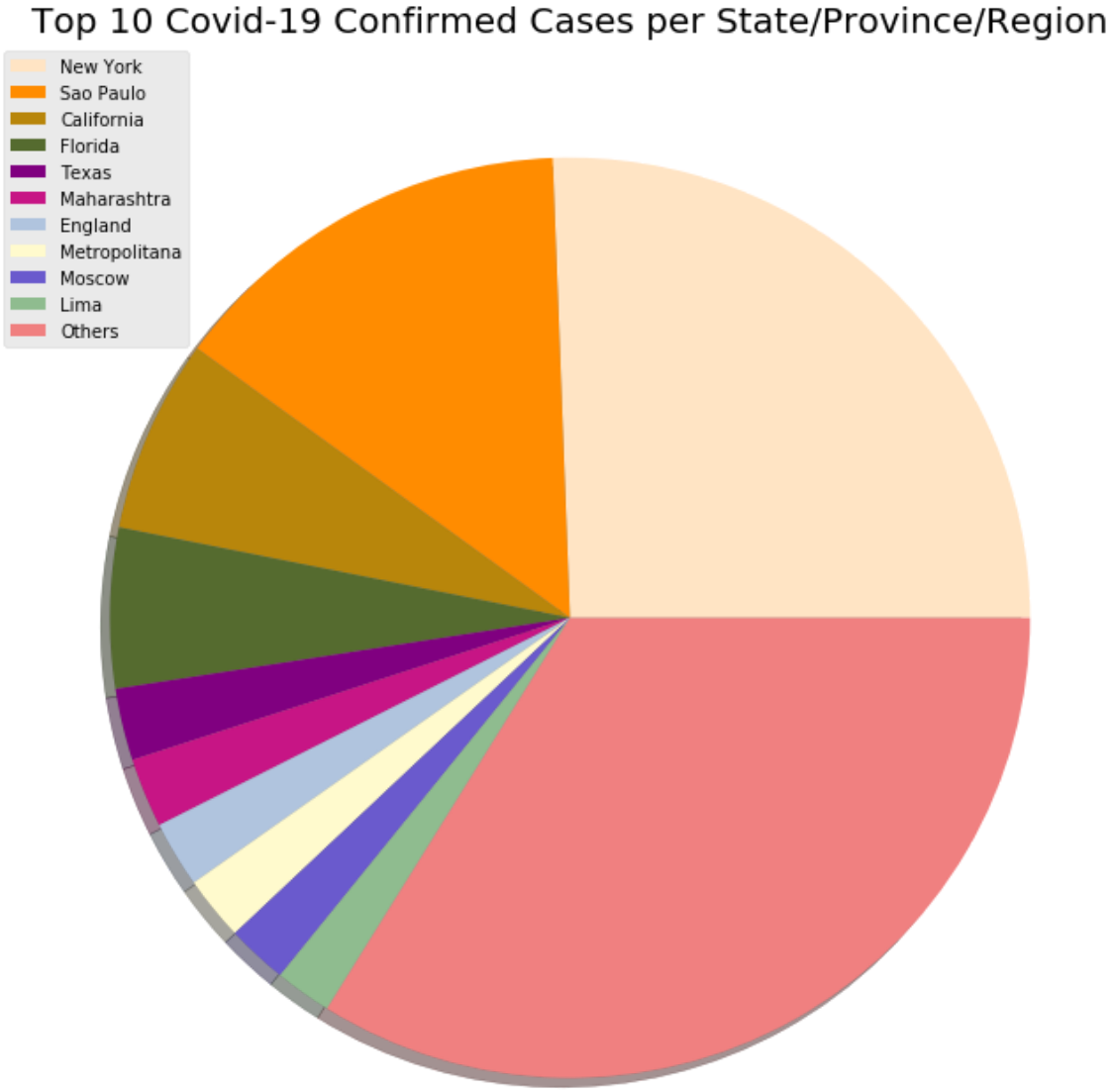
In [49]:

```
plot_pie_charts(visual_unique_countries, visual_confirmed_cases, 'Top 10 Covid-19 Confirmed
```



In [50]:

```
plot_pie_charts(visual_unique_provinces, visual_confirmed_cases, 'Top 10 Covid-19 Confirmed
```



In [51]:

```
# Plotting countries with regional data using a pie chart

def get_provinces_pie_chart(country_name):
    temp_df = latest_data[latest_data['Country_Region'] == country_name].groupby('Province_
    temp_df = temp_df.sort_values('Confirmed',ascending= False).reset_index()
    if temp_df.shape[0] > 5:
        province_name = list(temp_df.iloc[:5,0])
        province_count = list(temp_df.iloc[:5,4])
        province_name.append('others')
        province_count.append(sum(list(temp_df.iloc[5:,4])))
        title='Covid-19 Confirmed Cases in {}'.format(str(country_name))
        plot_pie_charts(province_name,province_count, title)
    else:
        province_name = list(temp_df.iloc[:,0])
        province_count = list(temp_df.iloc[:,4])
        title='Covid-19 Confirmed Cases in {}'.format(str(country_name))
        plot_pie_charts(province_name,province_count, title)
```

In [52]:

```
countries
```

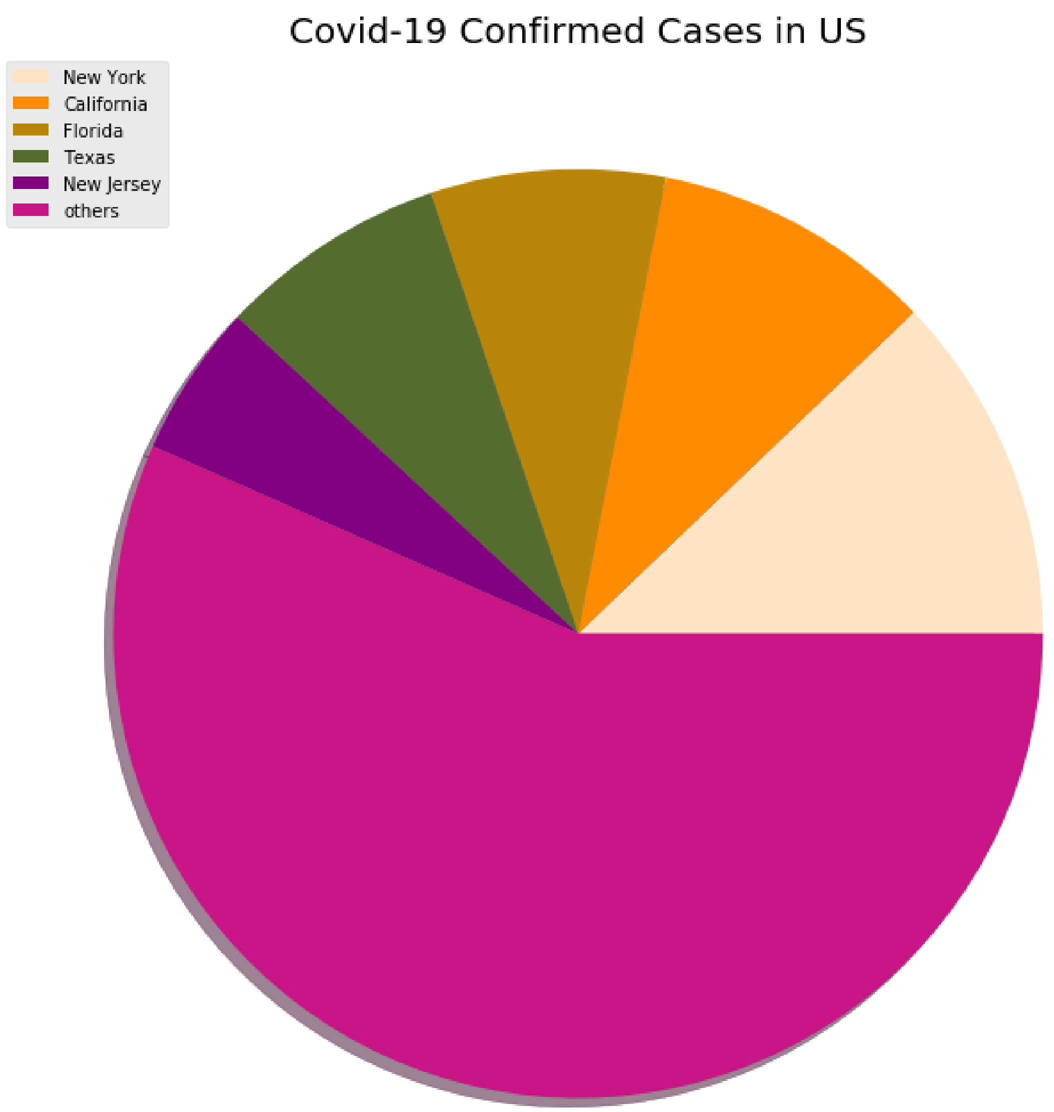
Out[52]:

```
['US',
 'Brazil',
 'India',
 'Russia',
 'Peru',
 'Chile',
 'Mexico',
 'United Kingdom',
 'South Africa',
 'Iran']
```

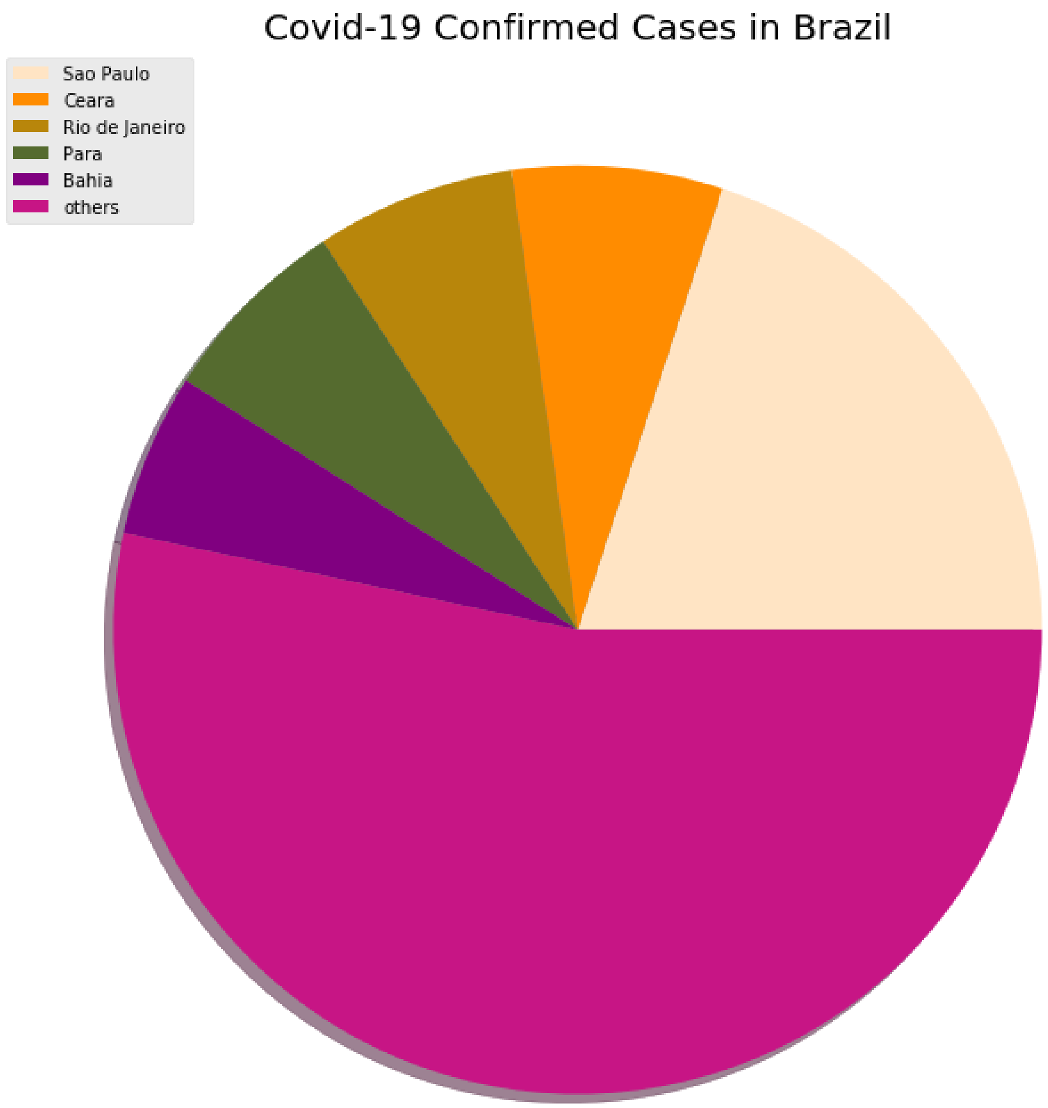
In [53]:

```
pie_chart_countries = ['US', 'Brazil', 'Russia', 'India', 'Peru', 'Mexico', 'Canada',
                       'Australia', 'China', 'Italy', 'Germany', 'France', 'United Kingdom'
#pie_chart_countries = countries[:8]
for country in pie_chart_countries:
    print('-*-*'*10,end = '')
    print(country,end ='')
    print('-*-*'*10)
    get_provinces_pie_chart(country)
```

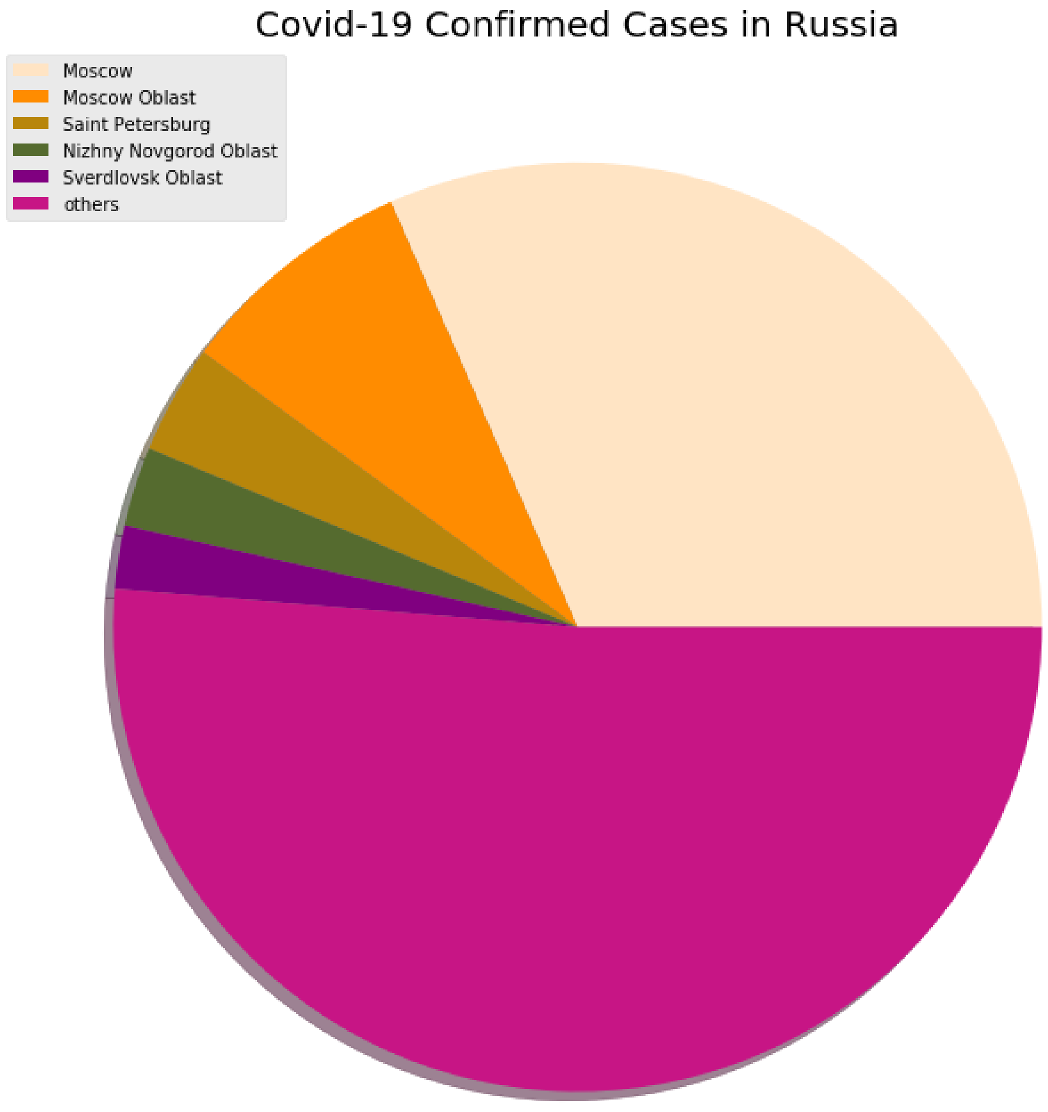
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*US-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*
```



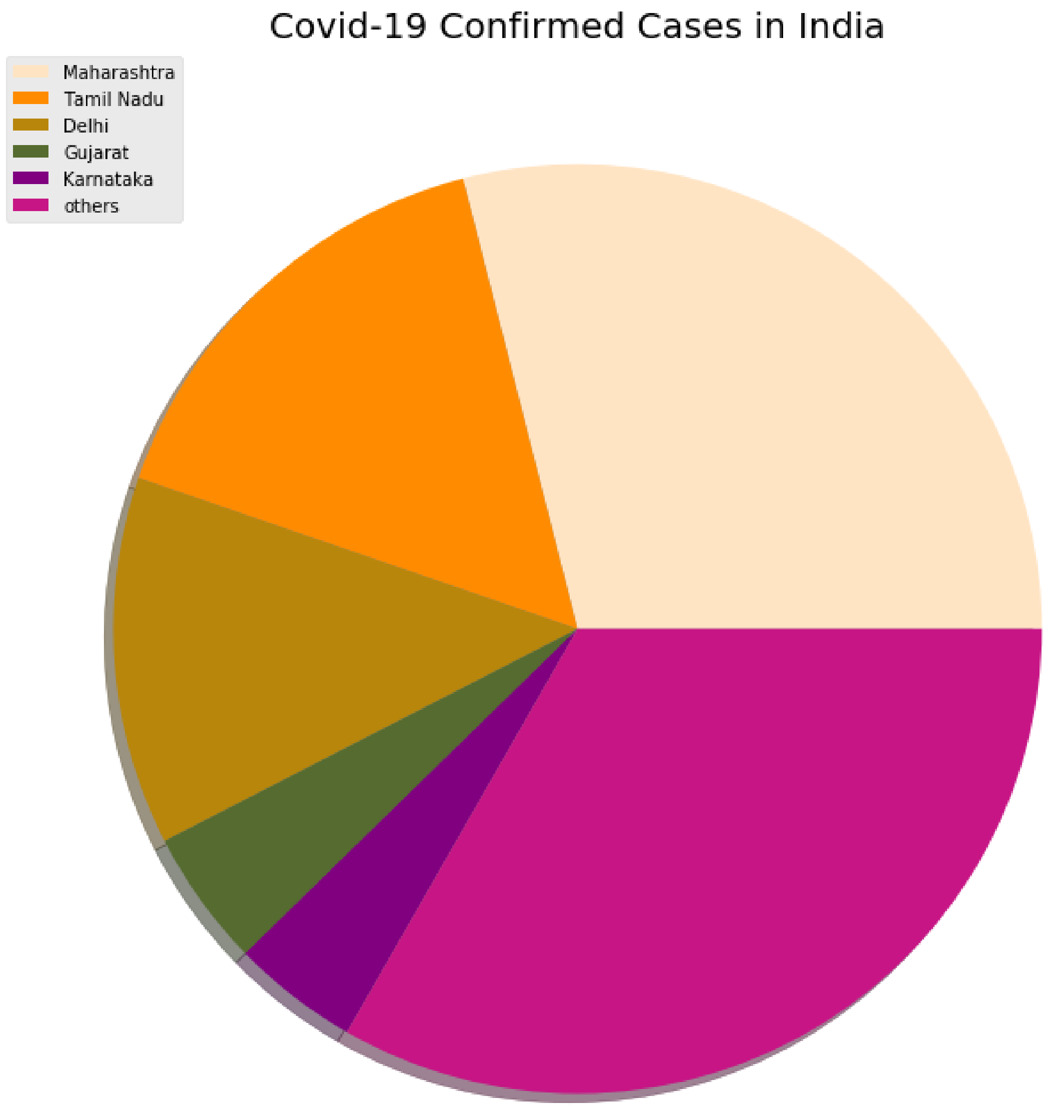
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Brazil-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*
```


Covid-19 Confirmed Cases in Brazil
Sao Paulo
Ceara
Rio de Janeiro
Para
Bahia
others

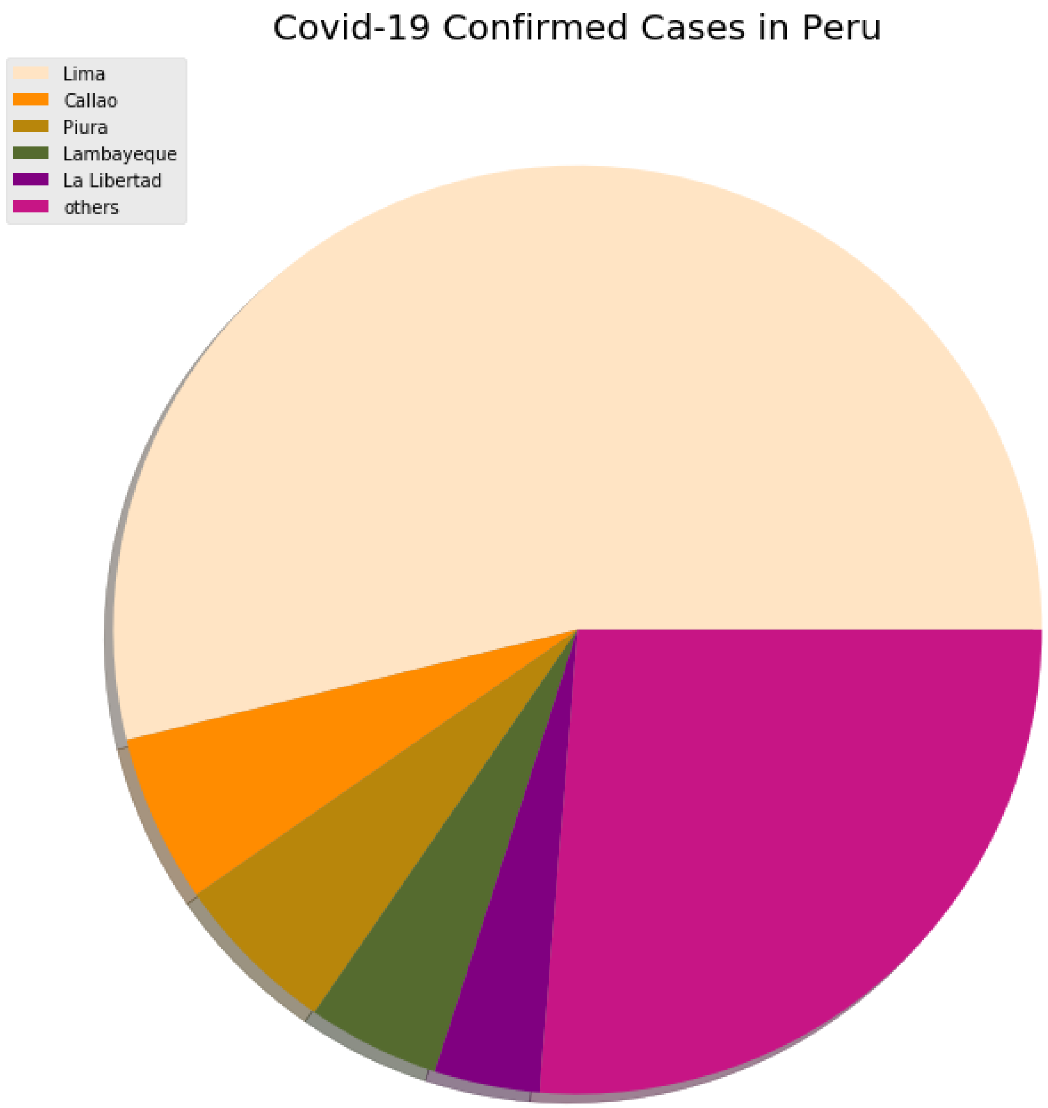
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Russia-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*
```


Covid-19 Confirmed Cases in Russia
Moscow
Moscow Oblast
Saint Petersburg
Nizhny Novgorod Oblast
Sverdlovsk Oblast
others

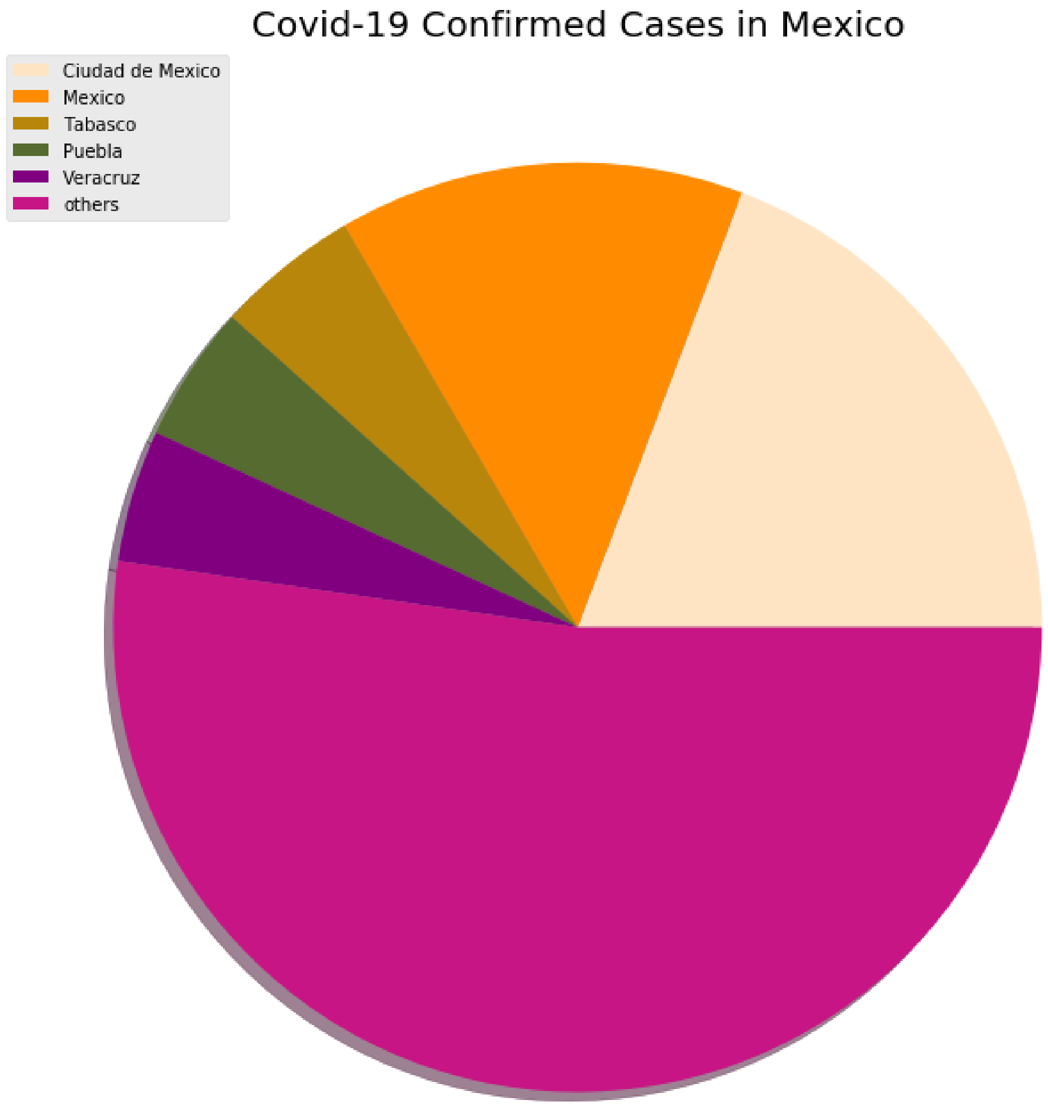
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*India-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
*-*-*-*-*
```


Covid-19 Confirmed Cases in India
Maharashtra
Tamil Nadu
Delhi
Gujarat
Karnataka
others

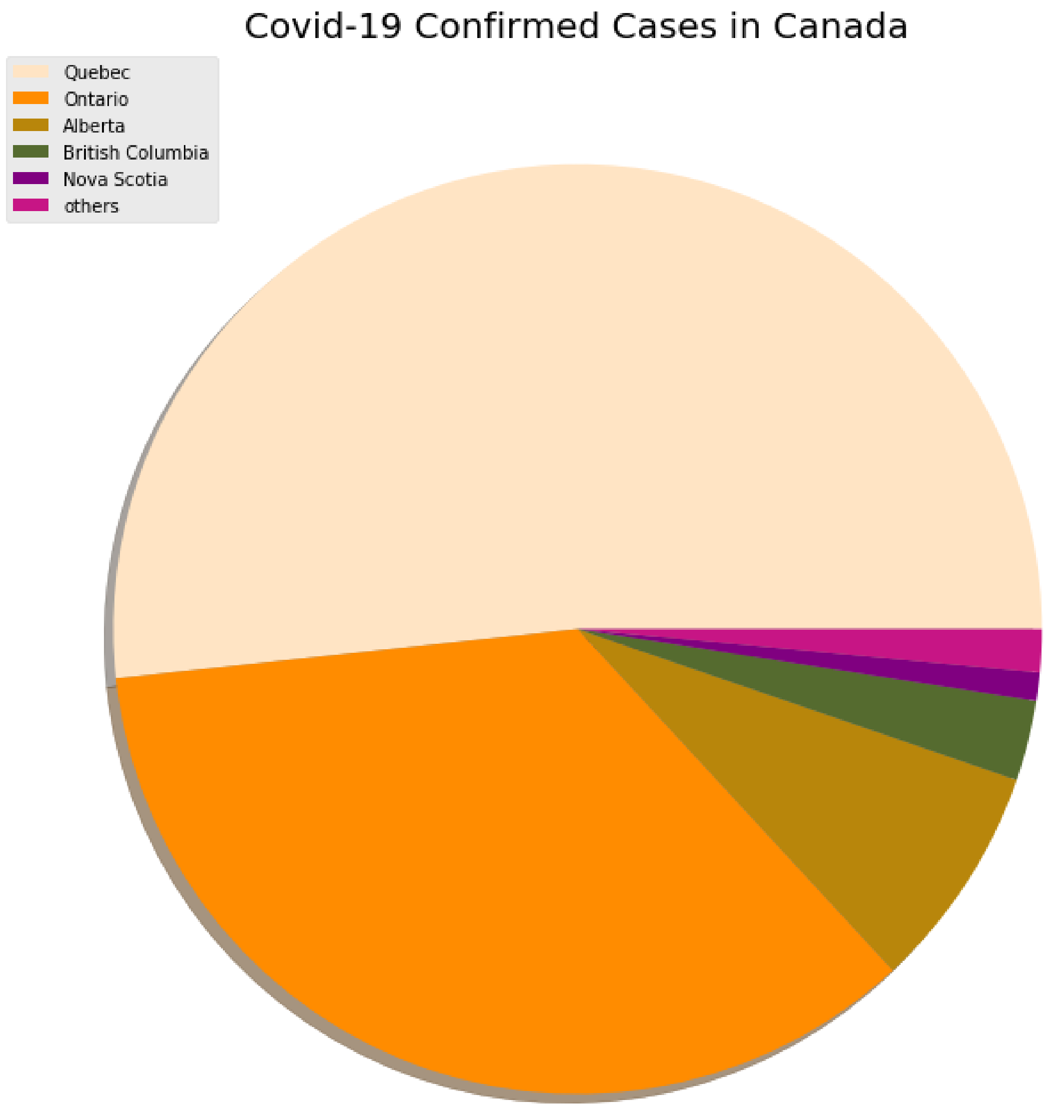
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Peru-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*
```


Covid-19 Confirmed Cases in Peru
Lima
Callao
Piura
Lambayeque
La Libertad
others

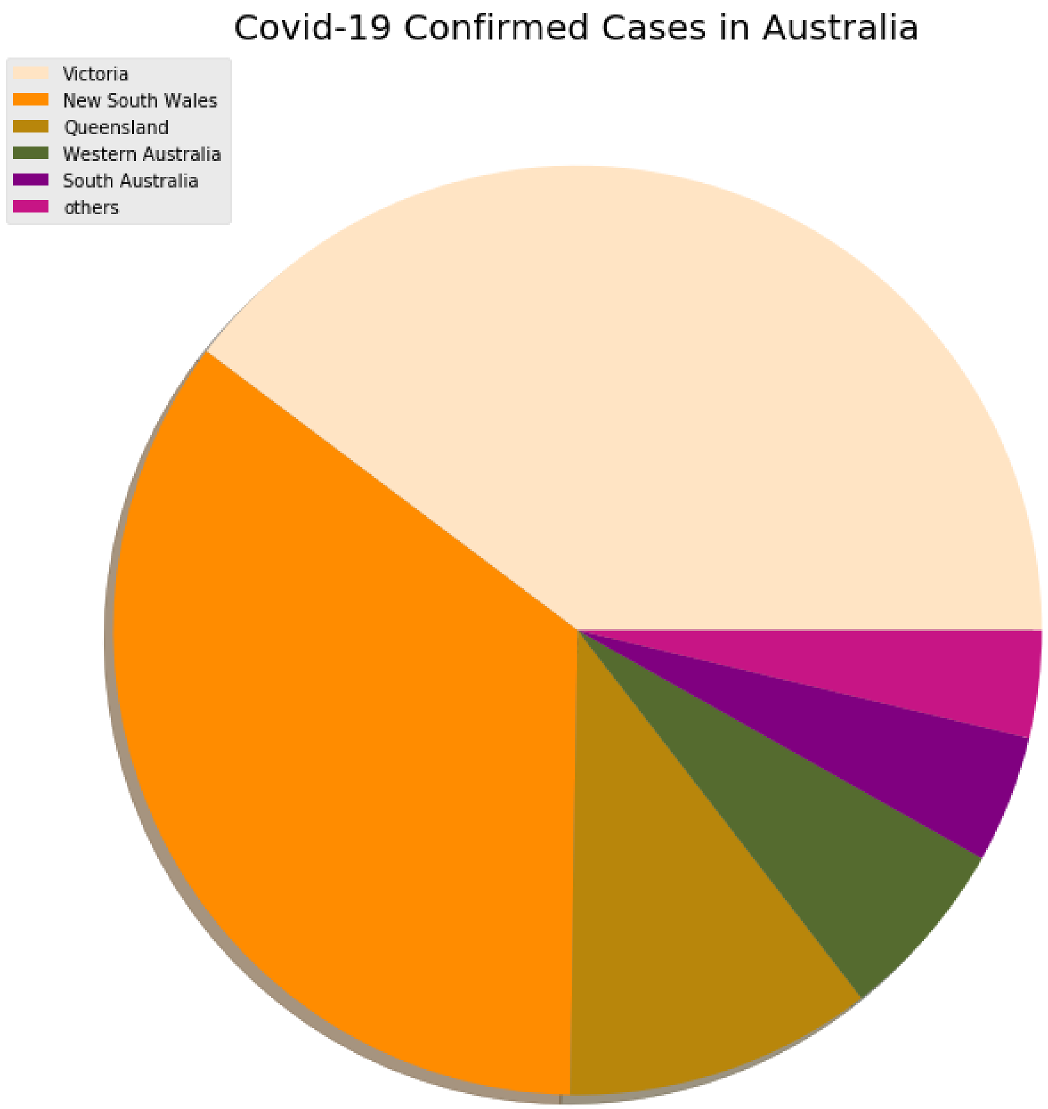
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Mexico-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*
```


Covid-19 Confirmed Cases in Mexico
Ciudad de Mexico
Mexico
Tabasco
Puebla
Veracruz
others

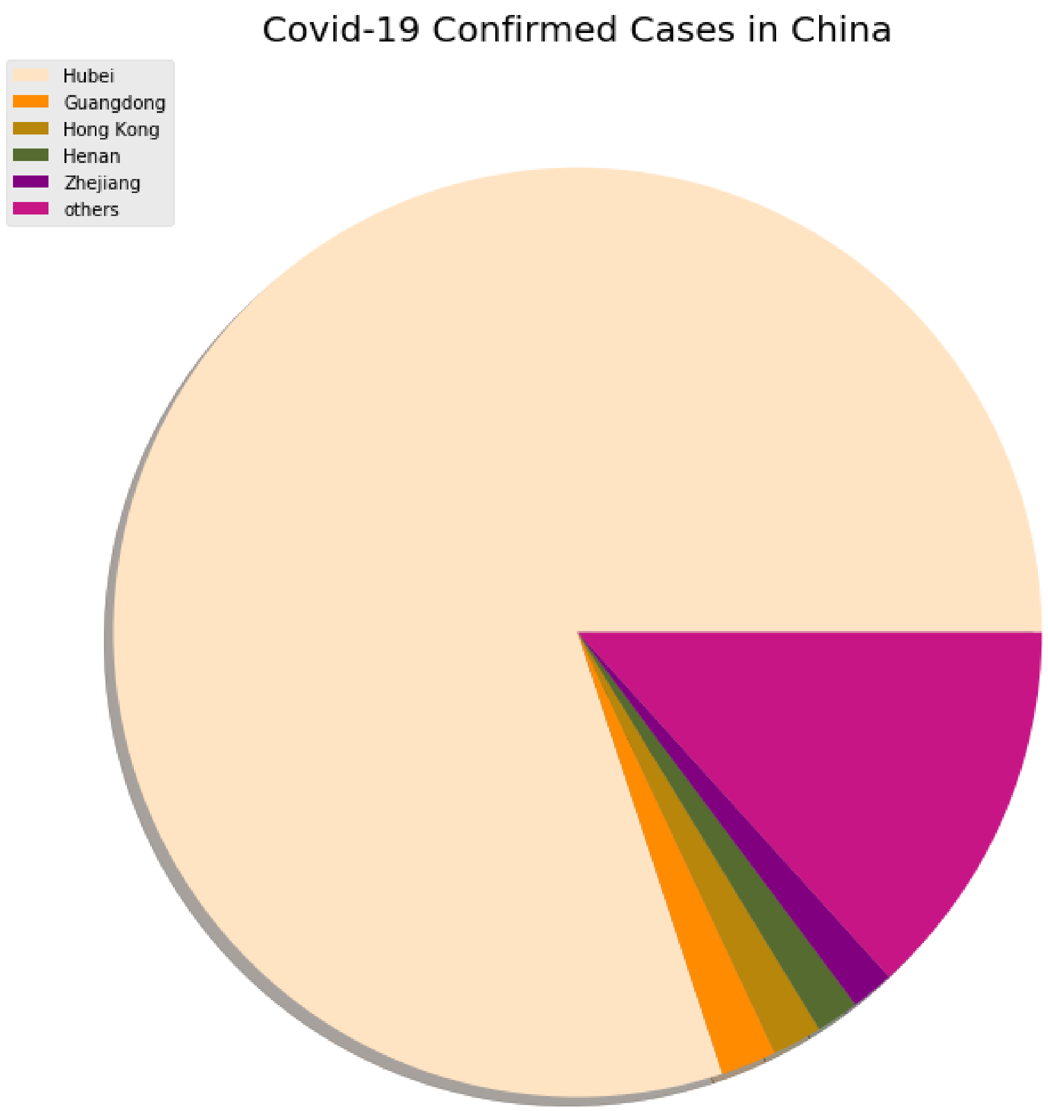
-*Canada-*


Covid-19 Confirmed Cases in Canada
Quebec
Ontario
Alberta
British Columbia
Nova Scotia
others

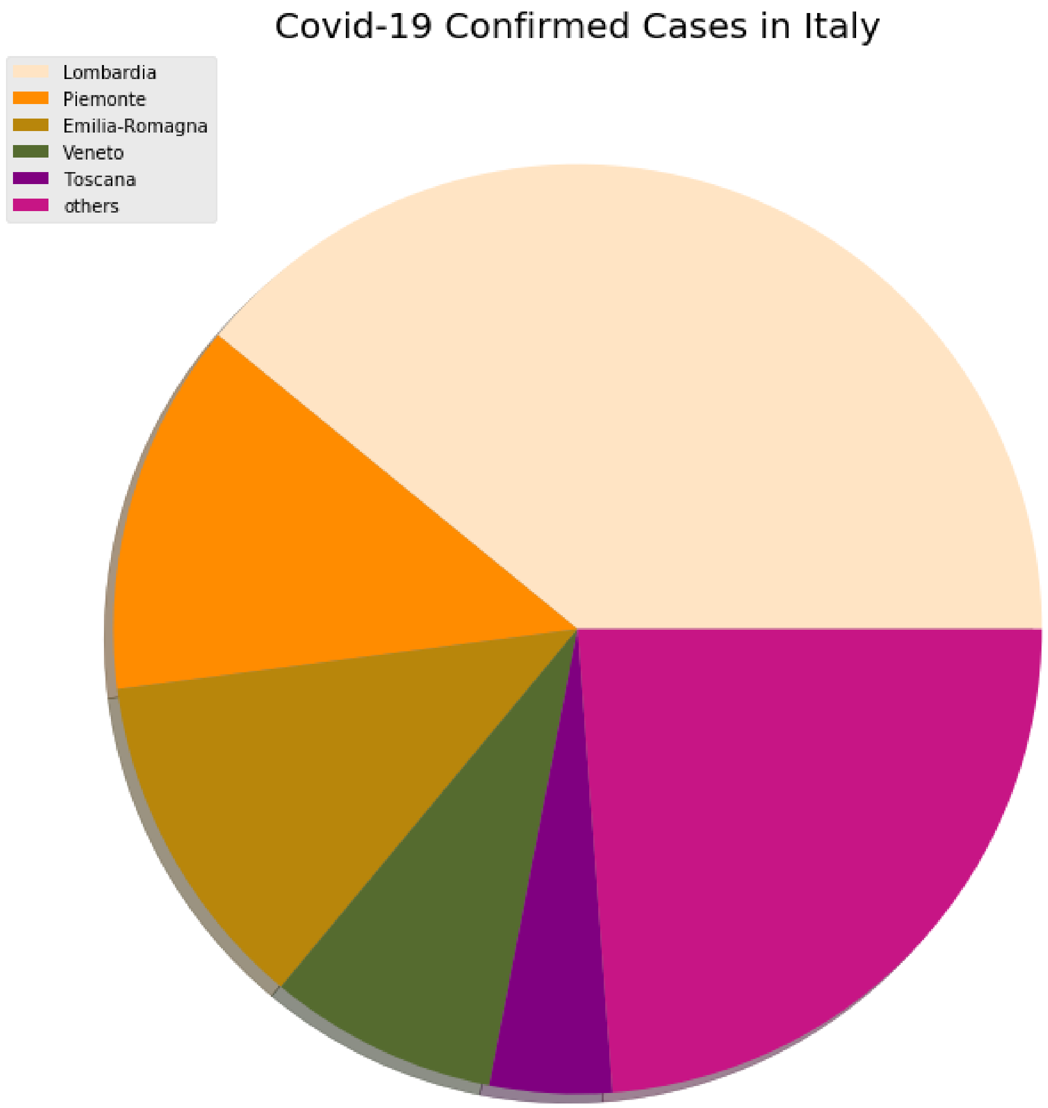
-*Australia-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
--*-*-*-*-*


Covid-19 Confirmed Cases in Australia
Victoria
New South Wales
Queensland
Western Australia
South Australia
others

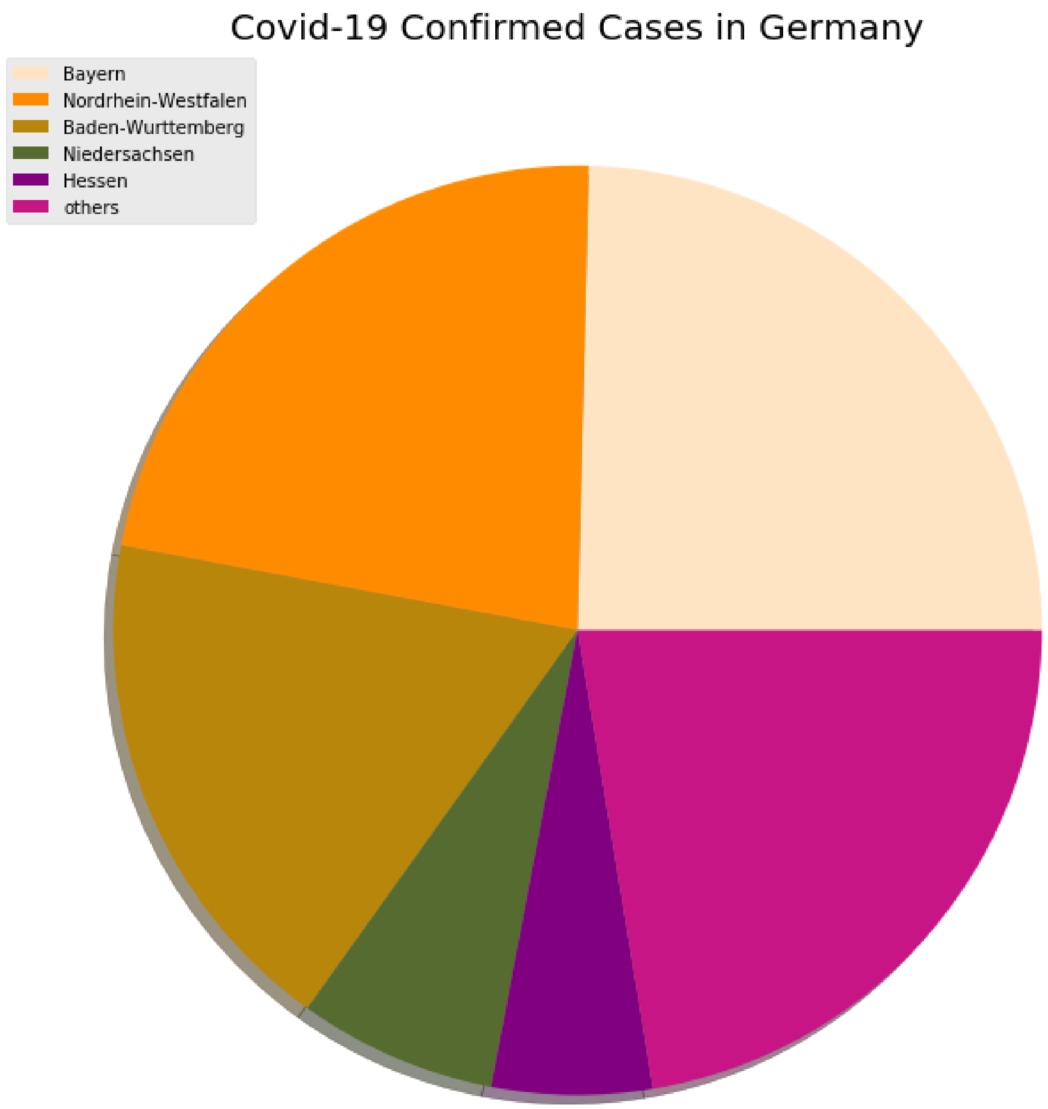
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*China-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
*-*-*-*-*
```


Covid-19 Confirmed Cases in China
Hubei
Guangdong
Hong Kong
Henan
Zhejiang
others

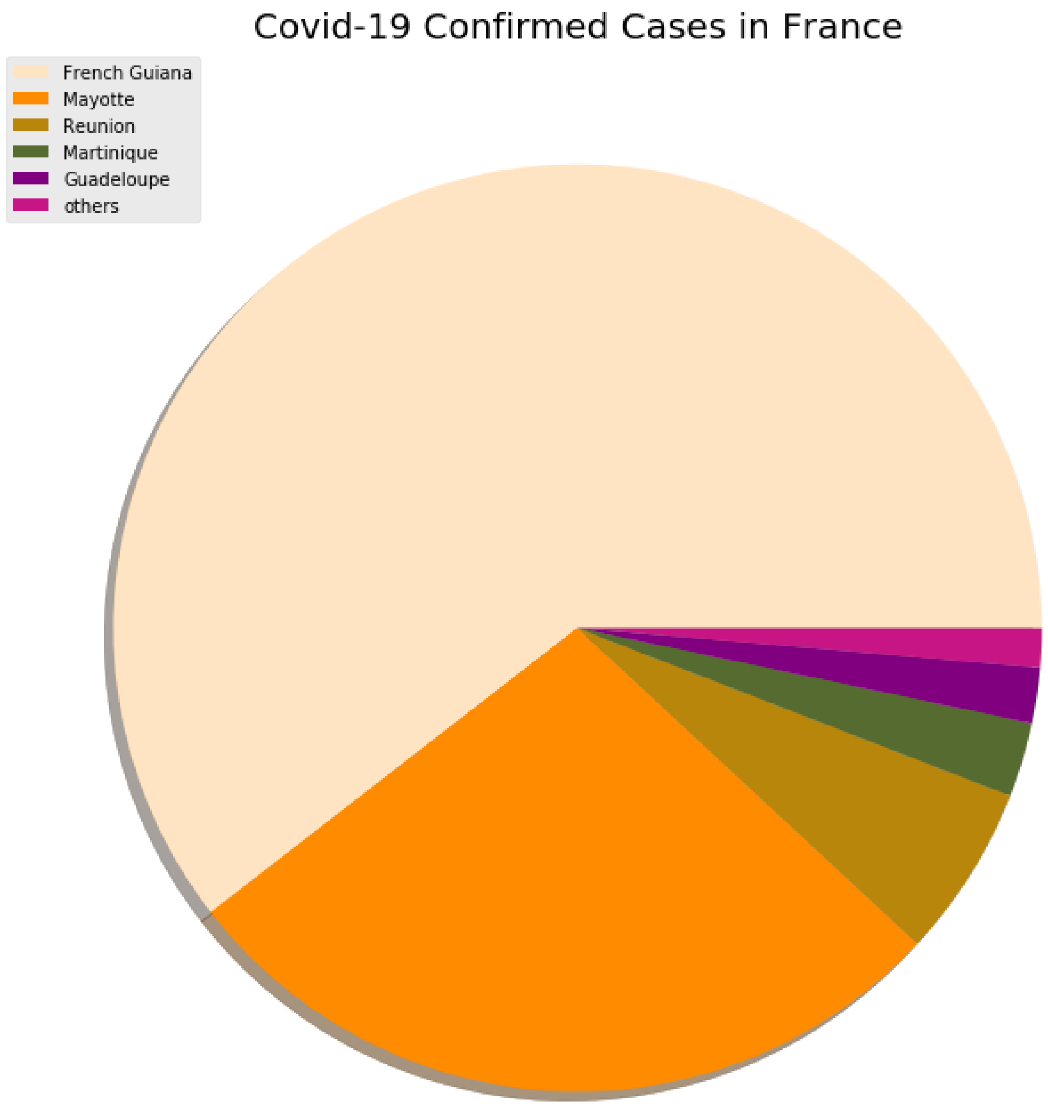
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Italy-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
*-*-*-*-*
```


Covid-19 Confirmed Cases in Italy
Lombardia
Piemonte
Emilia-Romagna
Veneto
Toscana
others

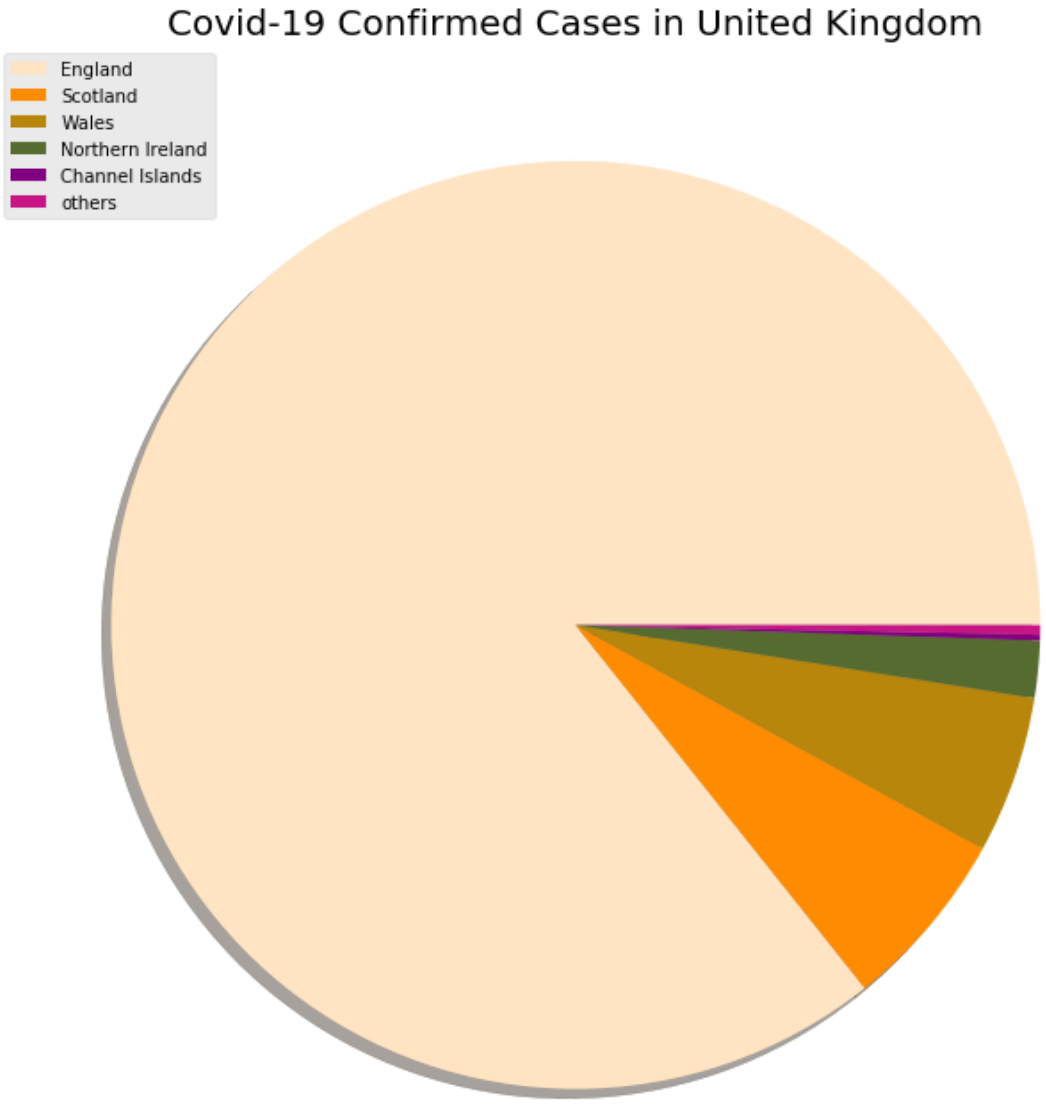
```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*Germany-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
*-*-*-*-*-*
```


Covid-19 Confirmed Cases in Germany
Bayern
Nordrhein-Westfalen
Baden-Wurttemberg
Niedersachsen
Hessen
others

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*France-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*
```


Covid-19 Confirmed Cases in France
French Guiana
Mayotte
Reunion
Martinique
Guadeloupe
others

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*United Kingdom-*-*-*-*-*-*-*-*-*-*-*
-*-*-*-*-*-*-*-*-*-*
```



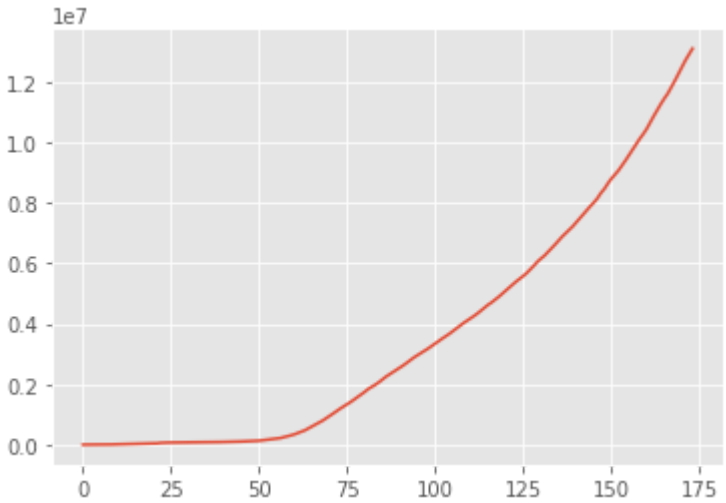
Prediction for confirmed cases over worldwide

In [54]:

```
plt.plot(days_since_1_22,world_cases)
```

Out[54]:

```
[<matplotlib.lines.Line2D at 0x16b3c136648>]
```



In [55]:

```
X_train_confirmed, X_test_confirmed, y_train_confirmed, y_test_confirmed = train_test_split
```

Finding best parameters using Randomized Search as it is faster then the Grid Search

In [56]:

```
# use this to find the optimal parameters for SVR
#c = [0.01, 0.1, 1]
#gamma = [0.01, 0.1, 1]
#epsilon = [0.01, 0.1, 1]
#shrinking = [True, False]
#degree = [3, 4, 5]

#svm_grid = {'C': c, 'gamma' : gamma, 'epsilon': epsilon, 'shrinking' : shrinking, 'degree'

#svm = SVR(kernel='poly')
#svm_search = RandomizedSearchCV(svm, svm_grid, scoring='neg_mean_squared_error', cv=3, ret
#svm_search.fit(X_train_confirmed, y_train_confirmed)'''
```

In [57]:

```
#svm_search.best_params_
```

Building SVM model

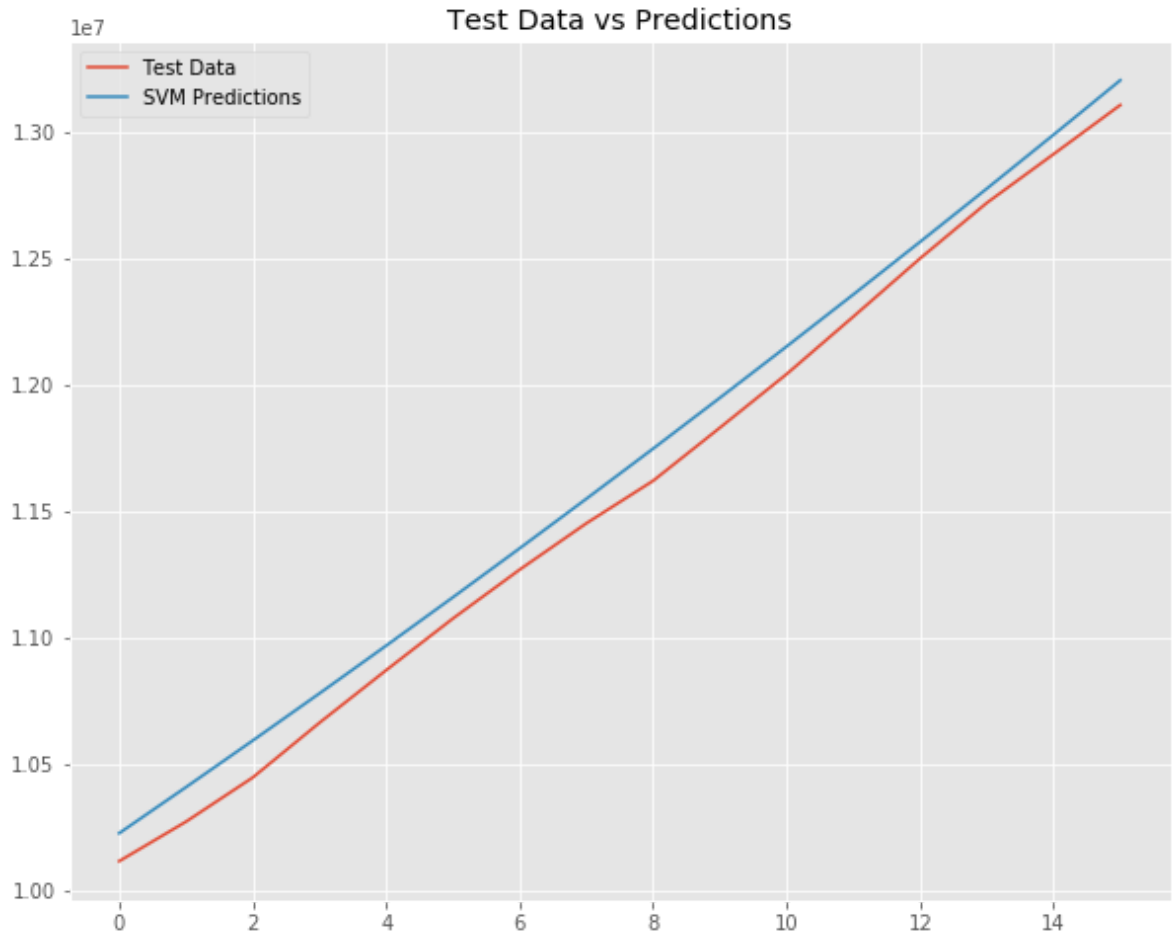
In [58]:

```
svm_confirmed = SVR(shrinking=True, kernel='poly',gamma=0.01, epsilon=1,degree=3, C=0.1)
svm_confirmed.fit(X_train_confirmed, y_train_confirmed)
svm_pred = svm_confirmed.predict(future_forcast)
```

In [59]:

```
svm_test_pred = svm_confirmed.predict(X_test_confirmed)
plt.figure(figsize=(10,8))
plt.title('Test Data vs Predictions')
plt.plot(y_test_confirmed)
plt.plot(svm_test_pred)
plt.legend(['Test Data', 'SVM Predictions'])
print('MAE:', mean_absolute_error(svm_test_pred, y_test_confirmed))
print('MSE:',mean_squared_error(svm_test_pred, y_test_confirmed)/1000000000)
print('r2_score:',r2_score(svm_test_pred, y_test_confirmed)*100)
```

```
MAE: 100525.51082913461
MSE: 10.685429589228875
r2_score: 98.72116585113653
```



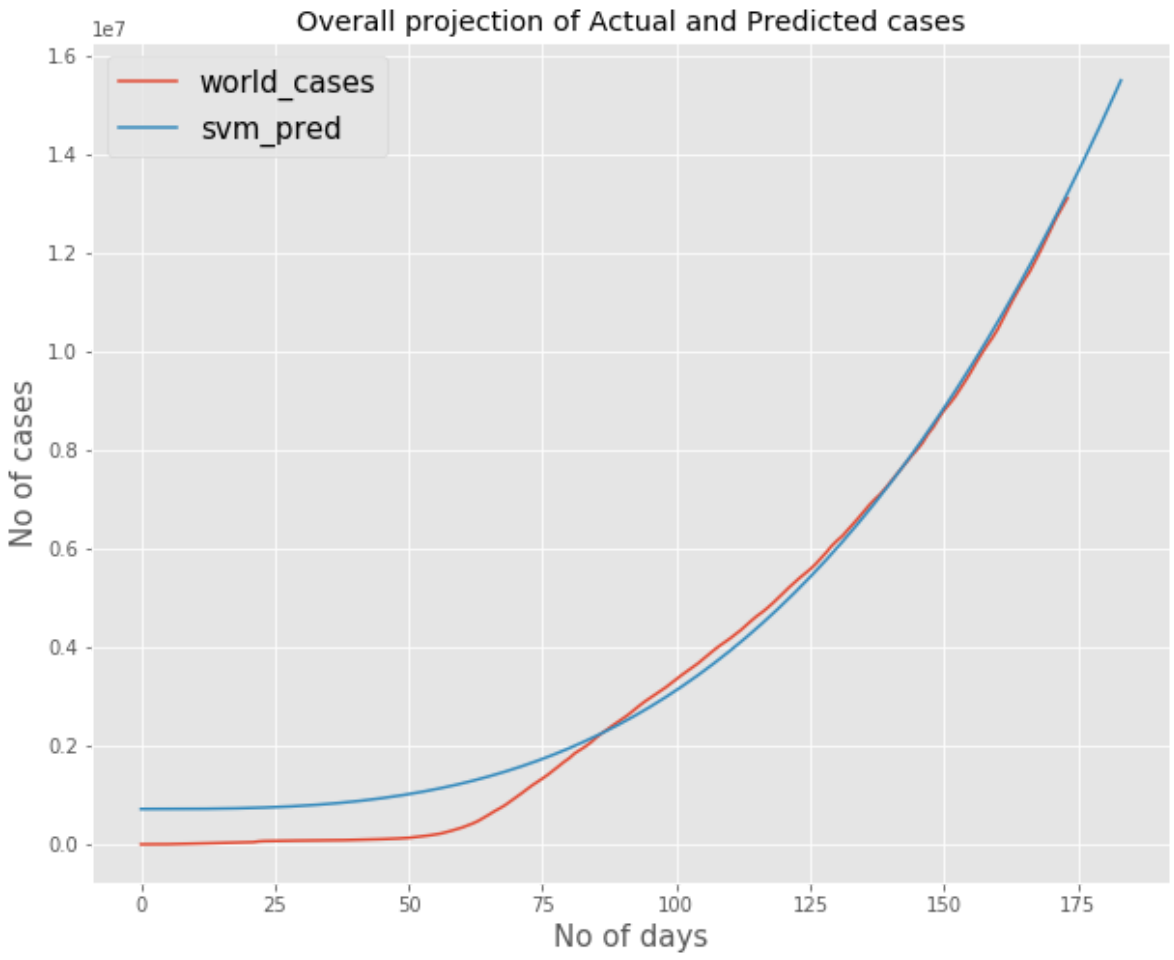
Overall projection of Actual and Predicted cases

In [60]:

```
plt.figure(figsize=(10,8))
plt.plot(days_since_1_22,world_cases)
plt.plot(future_forcast,svm_pred)
plt.title('Overall projection of Actual and Predicted cases')
plt.legend(['world_cases','svm_pred'],fontsize = 15)
plt.xlabel('No of days',size = 15)
plt.ylabel('No of cases',size=15)
```

Out[60]:

```
Text(0, 0.5, 'No of cases')
```



In [61]:

```
# transform our data for polynomial regression
poly = PolynomialFeatures(degree=5)
poly_X_train_confirmed = poly.fit_transform(X_train_confirmed)
poly_X_test_confirmed = poly.fit_transform(X_test_confirmed)
poly_future_forcast = poly.fit_transform(future_forcast)
```

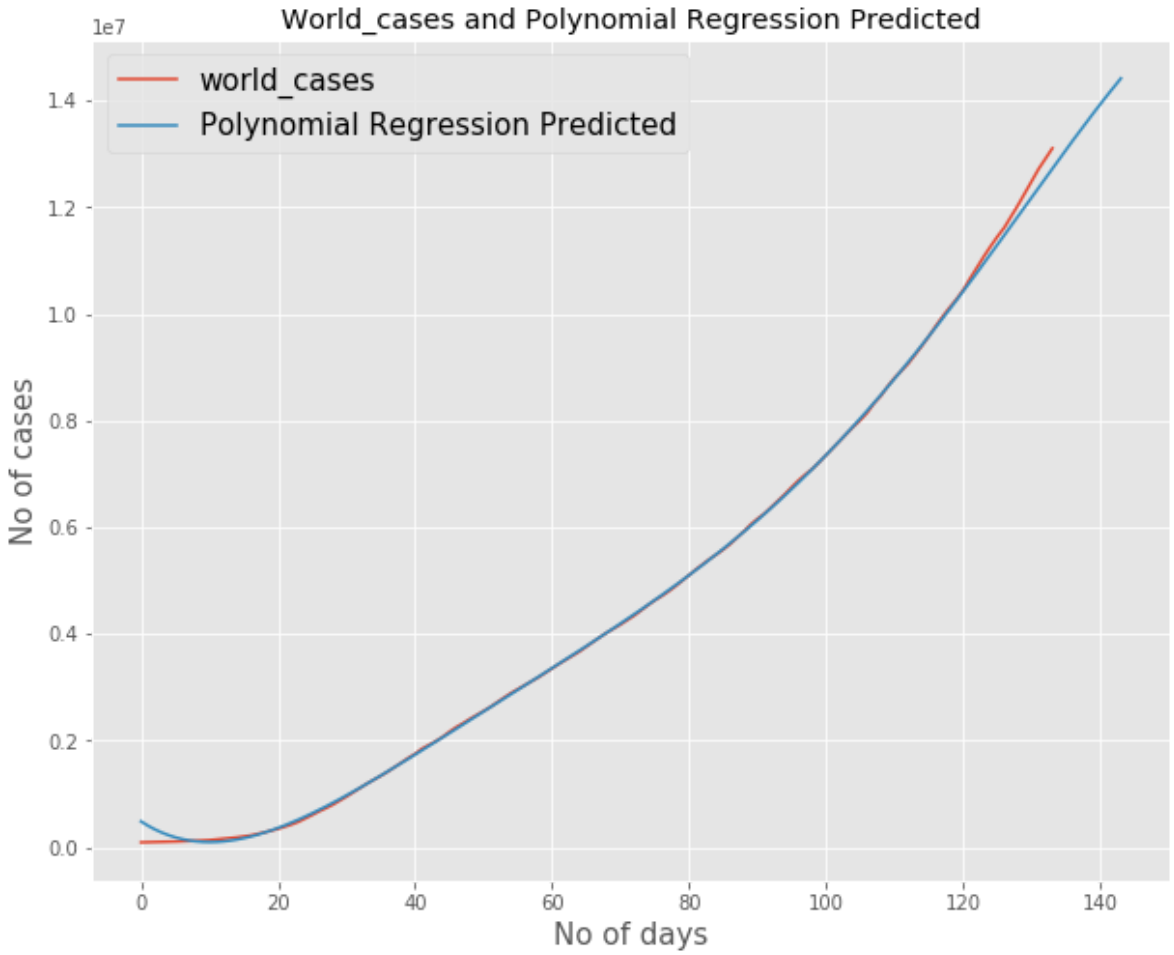
In [62]:

```
# polynomial regression
linear_model = LinearRegression(normalize=True, fit_intercept=False)
linear_model.fit(poly_X_train_confirmed, y_train_confirmed)
test_linear_pred = linear_model.predict(poly_X_test_confirmed)
linear_pred = linear_model.predict(poly_future_forcast)
print('MAE:', mean_absolute_error(test_linear_pred, y_test_confirmed))
print('MSE:',mean_squared_error(test_linear_pred, y_test_confirmed)/1000000000)
print('R2_SCORE:',r2_score(test_linear_pred, y_test_confirmed)*100)
```

```
MAE: 178039.9687996395
MSE: 45.46661105469835
R2_SCORE: 93.14031942255382
```

In [63]:

```
plt.figure(figsize=(10,8))
plt.title('World_cases and Polynomial Regression Predicted')
plt.plot(world_cases[40:])
plt.plot(linear_pred[40:])
plt.legend(['world_cases','Polynomial Regression Predicted'],fontsize = 15)
plt.xlabel('No of days',size = 15)
plt.ylabel('No of cases',size=15)
plt.show()
```



Conclusion :- SVM out performs in case of Confirmed Cases in every aspect (MSE,r2_score)

In [64]:

```
world_cases[-1][0] # current last case count
```

Out[64]:

```
13104391
```

In [65]:

```
#predicted confirmed case counts
future_case_count= [int(val) for val in svm_pred[-10:]]
future_case_count
```

Out[65]:

```
[13419976,
 13640254,
 13863064,
 14088420,
 14316337,
 14546830,
 14779912,
 15015599,
 15253904,
 15494843]
```

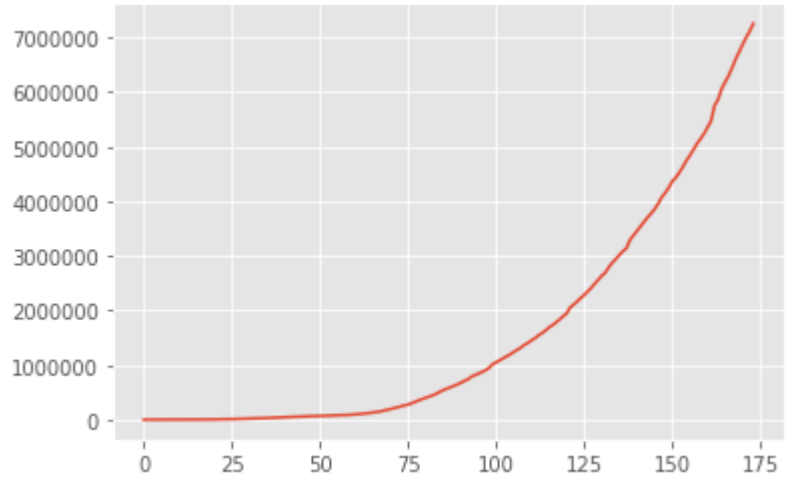
Prediction for Recovered cases over worldwide

In [66]:

```
plt.plot(days_since_1_22,total_recovered)
```

Out[66]:

```
[<matplotlib.lines.Line2D at 0x16b3c1e99c8>]
```



In [67]:

```
X_train_recovered, X_test_recovered, y_train_recovered, y_test_recovered = train_test_split
```

In [68]:

```
# use this to find the optimal parameters for SVR
#c = [0.01, 0.1, 1]
#gamma = [0.01, 0.1, 1]
#epsilon = [0.01, 0.1, 1]
#degree = [3, 4, 5]

#svm_grid = {'C': c, 'gamma' : gamma, 'epsilon': epsilon, 'shrinking' : shrinking, 'degree'

#svm = SVR(kernel='poly')
#svm_search = RandomizedSearchCV(svm, svm_grid, cv=3, n_jobs=-1, n_iter=10, verbose=1)
#svm_search.fit(X_train_confirmed, y_train_confirmed)'''
```

In [69]:

```
#svm_search.best_params_
```

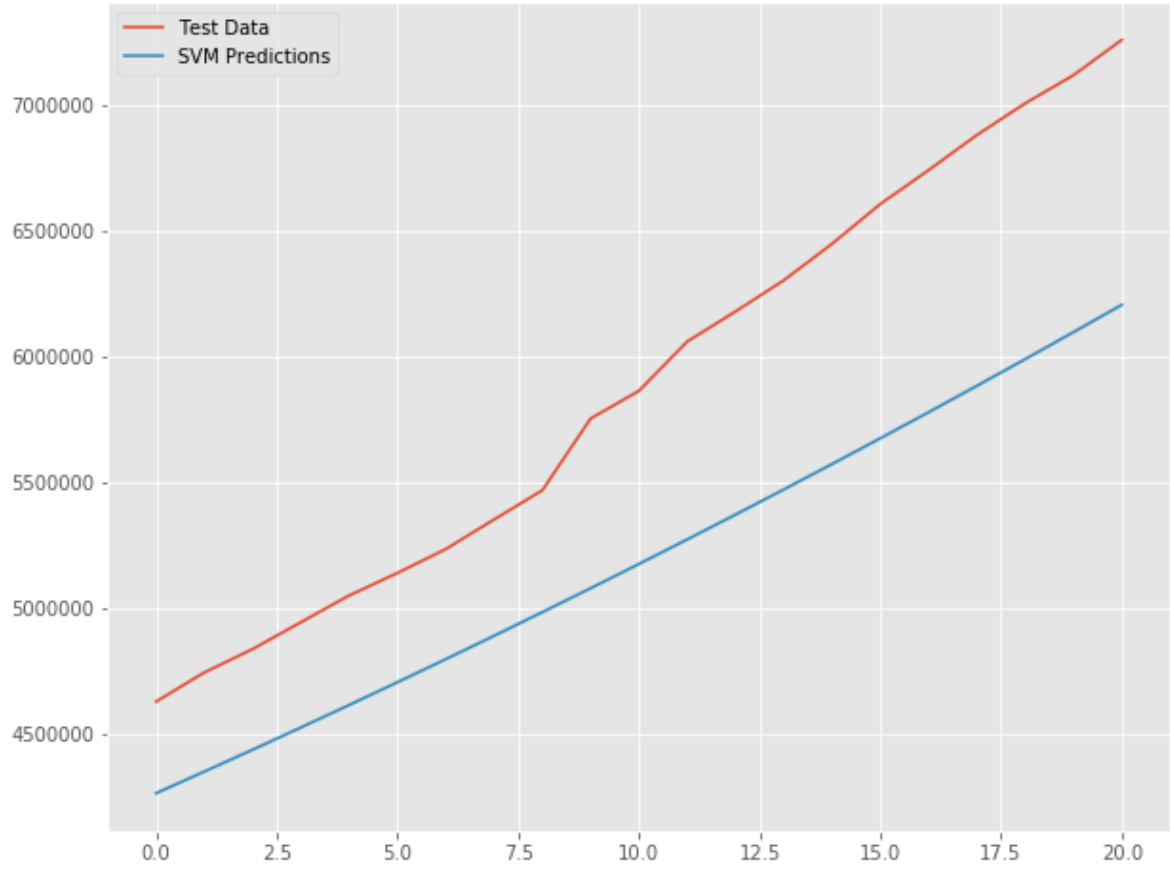
In [70]:

```
svm_recovered = SVR(shrinking=True, kernel='poly',gamma=0.01, epsilon=1,degree=3, C=0.1)
svm_recovered.fit(X_train_recovered, y_train_recovered)
svm_pred = svm_recovered.predict(future_forcast)
```

In [71]:

```
svm_test_pred = svm_recovered.predict(X_test_recovered)
plt.figure(figsize=(10,8))
plt.plot(y_test_recovered)
plt.plot(svm_test_pred)
plt.legend(['Test Data', 'SVM Predictions'])
print('MAE:', mean_absolute_error(svm_test_pred, y_test_recovered))
print('MSE:',mean_squared_error(svm_test_pred, y_test_recovered)/1000000000)
```

```
MAE: 688314.9931560386
MSE: 535.1556065192208
```



In [72]:

```
# transform our data for polynomial regression
poly = PolynomialFeatures(degree=6)
poly_X_train_recovered = poly.fit_transform(X_train_recovered)
poly_X_test_recovered = poly.fit_transform(X_test_recovered)
poly_future_forcast = poly.fit_transform(future_forcast)
```

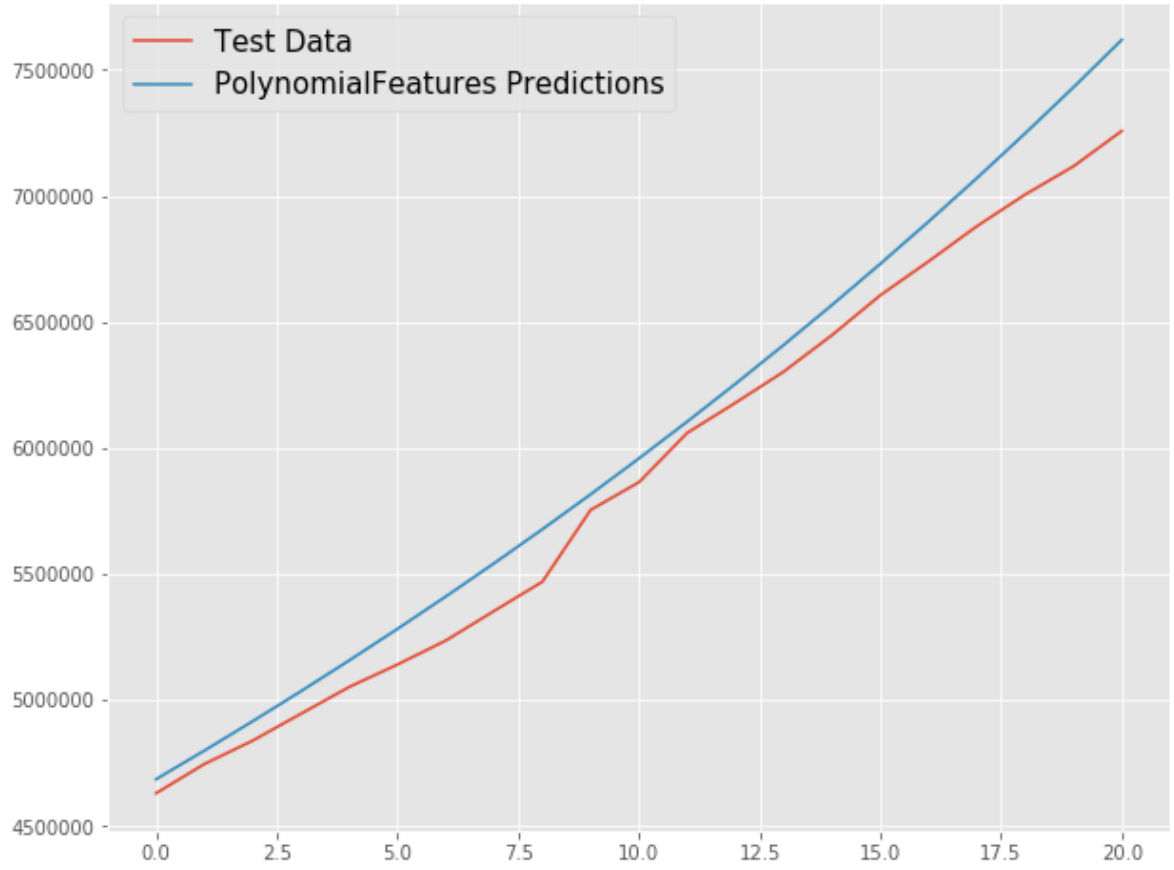
In [73]:

```
# polynomial regression
linear_model = LinearRegression(normalize=True, fit_intercept=False)
linear_model.fit(poly_X_train_recovered, y_train_recovered)
test_linear_pred = linear_model.predict(poly_X_test_recovered)
linear_pred = linear_model.predict(poly_future_forcast)
print('MAE:', mean_absolute_error(test_linear_pred, y_test_recovered))
print('MSE:',mean_squared_error(test_linear_pred, y_test_recovered)/1000000000)
print('R2_SCORE:',r2_score(test_linear_pred, y_test_recovered)*100)
plt.figure(figsize=(10,8))
plt.plot(y_test_recovered)
plt.plot(test_linear_pred)
plt.legend(['Test Data', 'PolynomialFeatures Predictions'],fontsize = 15)
```

```
MAE: 142199.317376865
MSE: 27.189698333070922
R2_SCORE: 96.52960865966634
```

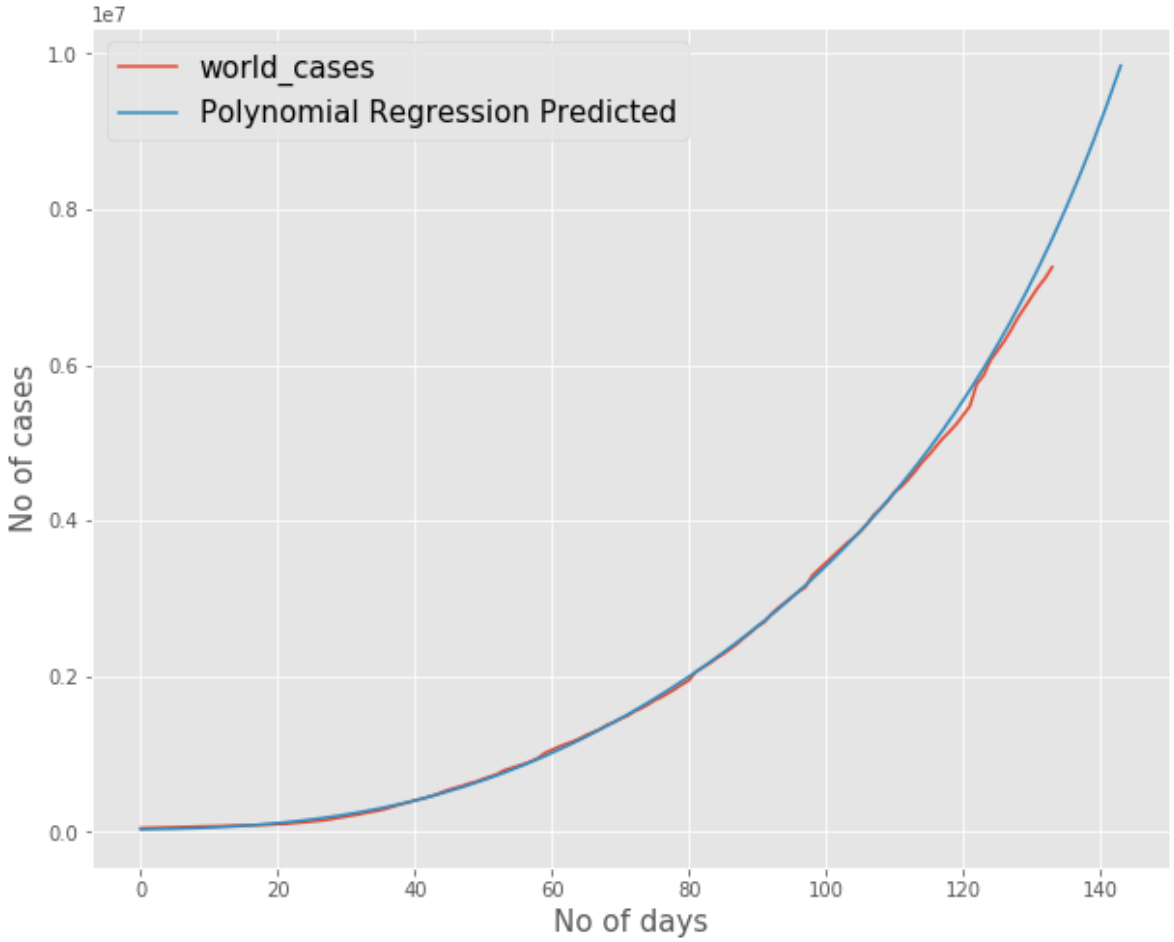
Out[73]:

```
<matplotlib.legend.Legend at 0x16b3c8aae88>
```



In [74]:

```
plt.figure(figsize=(10,8))
plt.plot(total_recovered[40:])
plt.plot(linear_pred[40:])
plt.legend(['world_cases','Polynomial Regression Predicted'],fontsize = 15)
plt.xlabel('No of days',size = 15)
plt.ylabel('No of cases',size=15)
plt.show()
```

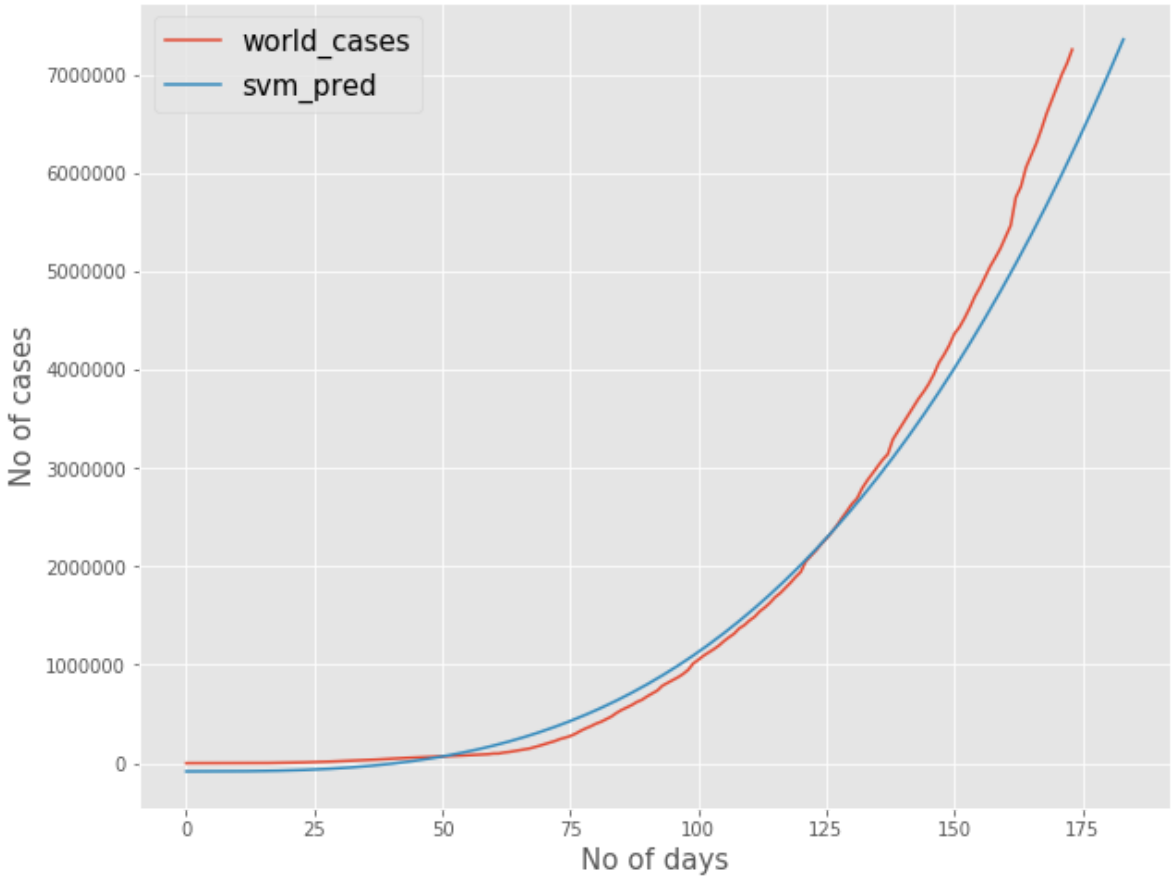


In [75]:

```
plt.figure(figsize=(10,8))
plt.plot(days_since_1_22,total_recovered)
plt.plot(future_forcast,svm_pred)
plt.legend(['world_cases','svm_pred'],fontsize = 15)
plt.xlabel('No of days',size = 15)
plt.ylabel('No of cases',size=15)
```

Out[75]:

```
Text(0, 0.5, 'No of cases')
```



Here Polynomial Regression Out Performs SVM as it can be seen from Graph and MSE values

In [76]:

```
total_deaths[-1][0]
```

Out[76]:

```
573003
```

In [77]:

```
#predicted recovery case counts
future_recover_count = linear_model.predict(poly_future_forcast[-10:])
future_recover_count = [int(val[0]) for val in future_recover_count]
future_recover_count
```

Out[77]:

```
[7812342,
 8011859,
 8217415,
 8429239,
 8647565,
 8872636,
 9104702,
 9344022,
 9590862,
 9845498]
```

Prediction for Death cases over worldwide

In [78]:

```
X_train_death, X_test_death, y_train_death, y_test_death = train_test_split(days_since_1_22
                                                                            total_deaths[50
                                                                            test_size=0.12,
                                                                            shuffle=False)
```

In [79]:

```
nd the optimal parameters for SVR
 1]
0.1, 1]


: c, 'gamma' : gamma}

l='poly')
ndomizedSearchCV(svm, svm_grid, scoring='neg_mean_squared_error', cv=3, return_train_score=1
X_train_confirmed, y_train_confirmed)'''
```

In [80]:

```
#svm_search.best_params_
```

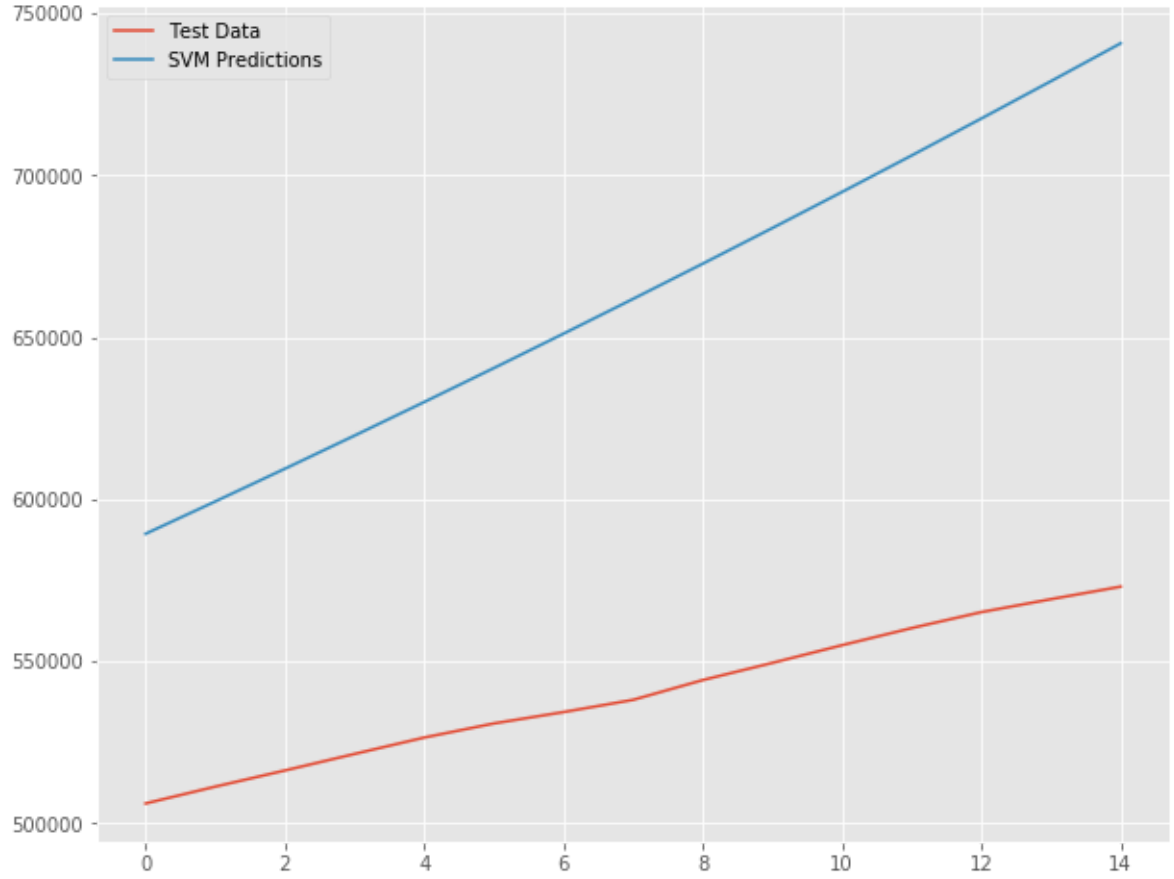
In [81]:

```
svm_death = SVR(shrinking=True, kernel='poly',gamma=0.01, epsilon=1,degree=3, C=0.1)
svm_death.fit(X_train_death, y_train_death)
svm_pred = svm_death.predict(future_forcast)
```

In [82]:

```
svm_test_pred = svm_death.predict(X_test_death)
plt.figure(figsize=(10,8))
plt.plot(y_test_death)
plt.plot(svm_test_pred)
plt.legend(['Test Data', 'SVM Predictions'])
print('MAE:', mean_absolute_error(svm_test_pred, y_test_death))
print('MSE:',mean_squared_error(svm_test_pred, y_test_death)/1000000000)
```

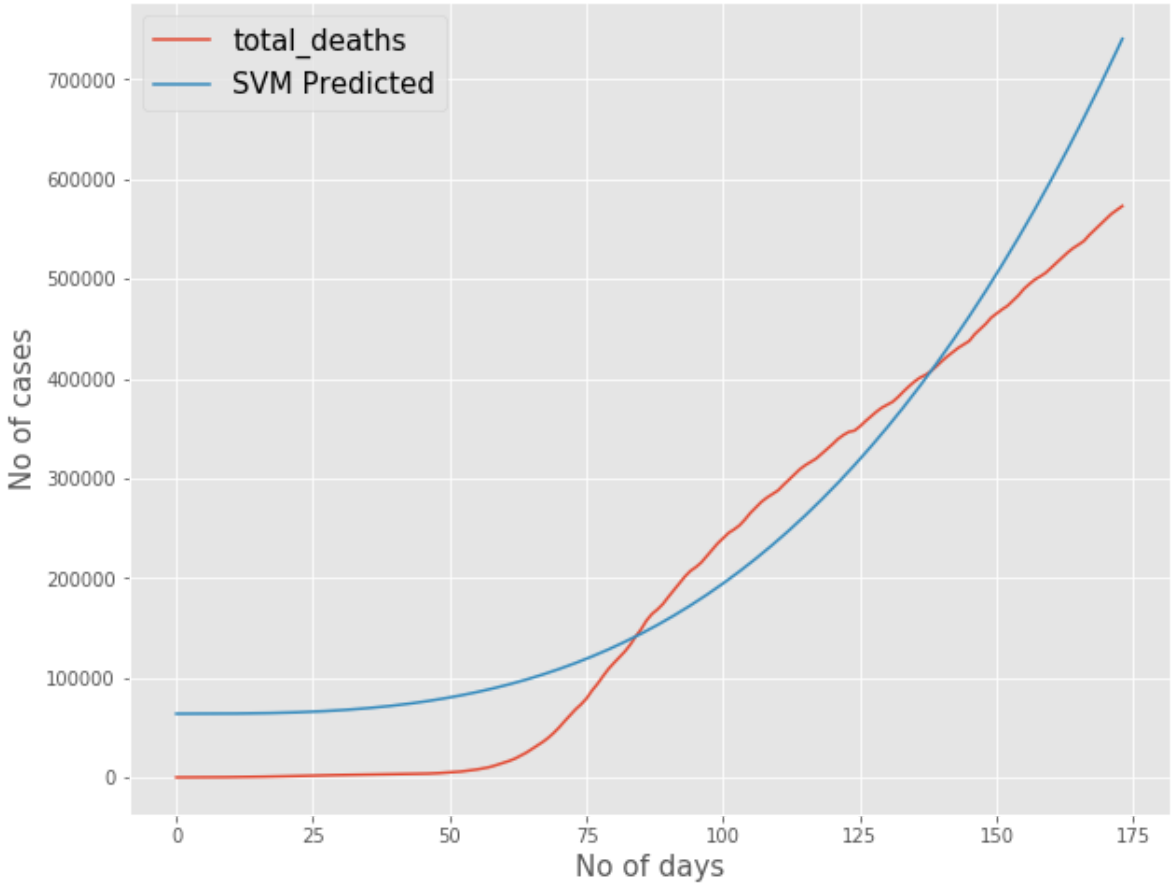
```
MAE: 122993.56283879364
MSE: 15.800004721292286
```



In [83]:

```
svm_pred = svm_death.predict(days_since_1_22)
print('MAE:', mean_absolute_error(svm_pred, total_deaths))
print('MSE:',mean_squared_error(svm_pred, total_deaths)/1000000000)
print('R2_score : ',r2_score(svm_pred, total_deaths)*100)
plt.figure(figsize=(10,8))
plt.plot(total_deaths)
plt.plot(svm_pred)
plt.legend(['total_deaths','SVM Predicted'],fontsize = 15)
plt.xlabel('No of days',size = 15)
plt.ylabel('No of cases',size=15)
plt.show()
```

```
MAE: 55486.47787262399
MSE: 4.002608608149228
R2_score :  89.30641934846717
```



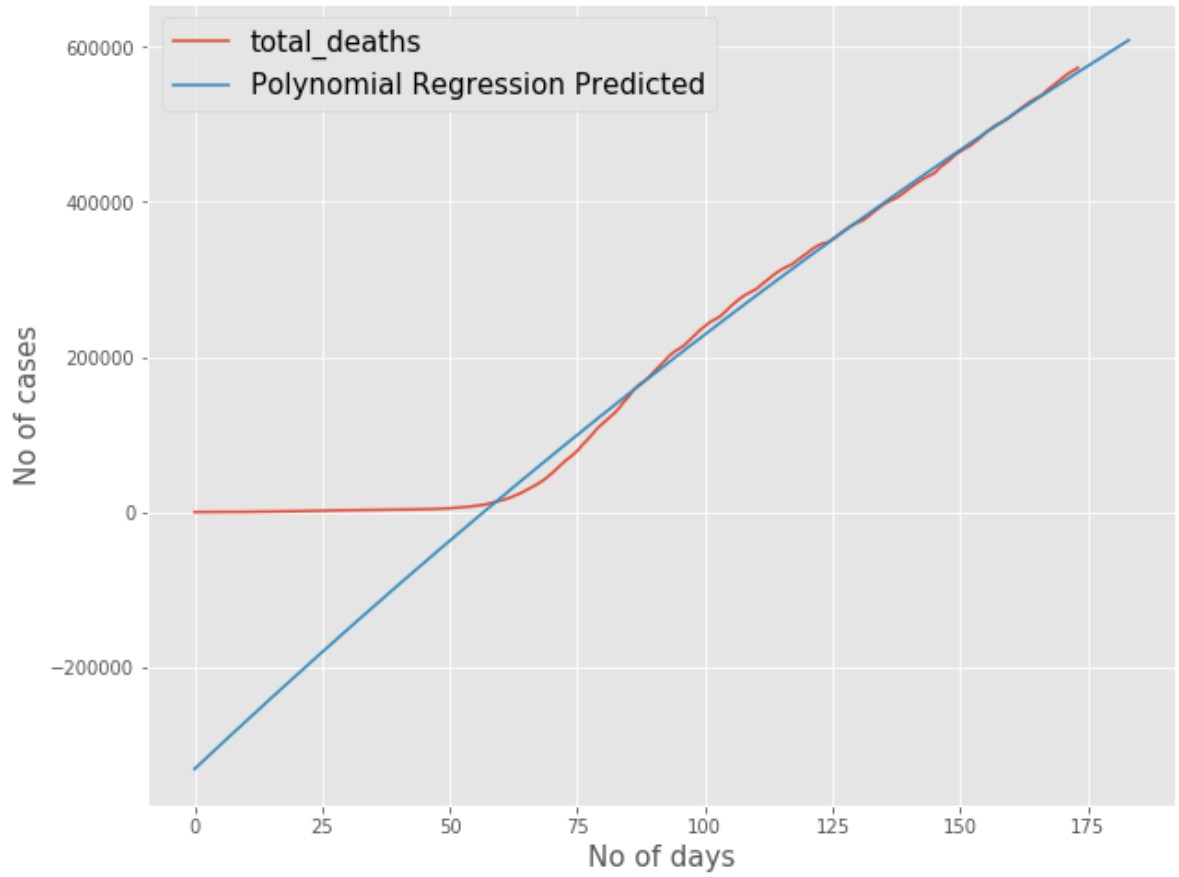
In [84]:

```
# transform our data for polynomial regression
poly = PolynomialFeatures(degree=2)
poly_X_train_death = poly.fit_transform(X_train_death)
poly_X_test_death = poly.fit_transform(X_test_death)
poly_future_forcast = poly.fit_transform(future_forcast)
```

In [85]:

```
# polynomial regression
linear_model = LinearRegression(normalize=True, fit_intercept=False)
linear_model.fit(poly_X_train_death, y_train_death)
test_linear_pred = linear_model.predict(poly_X_test_death)
linear_pred = linear_model.predict(poly_future_forcast)
print('MAE:', mean_absolute_error(test_linear_pred, y_test_death))
print('MSE:',mean_squared_error(test_linear_pred, y_test_death)/100000000)
print('R2_SCORE:',r2_score(test_linear_pred, y_test_death)*100)
plt.figure(figsize=(10,8))
plt.plot(total_deaths)
plt.plot(linear_pred)
plt.legend(['total_deaths','Polynomial Regression Predicted'],fontsize = 15)
plt.xlabel('No of days',size = 15)
plt.ylabel('No of cases',size=15)
plt.show()
```

```
MAE: 2907.487223150271
MSE: 0.13332335671676102
R2_SCORE: 96.1271025753039
```



Here Polynomial Regression Out Performs SVM as it can be seen from Graph and MSE values

In [86]:

```
total_deaths[-1][0] # recent death case count
```

Out[86]:

```
573003
```

In [87]:

```
#predicted death case counts
future_death_count = linear_model.predict(poly_future_forcast[-10:])
future_death_count = [int(val[0]) for val in future_death_count]
future_death_count
```

Out[87]:

```
[571373,
 575571,
 579758,
 583933,
 588097,
 592250,
 596392,
 600522,
 604641,
 608749]
```

Top countries effected with virus

In [88]:

```
print('Top countries effected with virus are : ',countries)
```

```
Top countries effected with virus are :  ['US', 'Brazil', 'India', 'Russia',
'Peru', 'Chile', 'Mexico', 'United Kingdom', 'South Africa', 'Iran']
```

In [89]:

```
start = '1/22/2020'
start_date = datetime.datetime.strptime(start, '%m/%d/%Y')
future_forcast_dates = []
for i in range(len(future_forcast)):
    future_forcast_dates.append((start_date + datetime.timedelta(days=i)).strftime('%m/%d/%
future_forcast_dates[-10:]
```

Out[89]:

```
['07/14/2020',
 '07/15/2020',
 '07/16/2020',
 '07/17/2020',
 '07/18/2020',
 '07/19/2020',
 '07/20/2020',
 '07/21/2020',
 '07/22/2020',
 '07/23/2020']
```

Prediction of Confirmed ,Recoveries,Deaths Case Counts

In [90]:

```
future_df = pd.DataFrame(list(zip(future_forcast_dates[-10:],future_case_count, future_reco
               columns =['Dates','Cases','Recoveries','Deaths'])

future_df['Moratality_rate'] = future_df.Deaths/future_df.Cases
future_df['Recovery_rate'] = future_df.Recoveries/future_df.Cases
future_df.set_index('Dates',inplace=True)
future_df
```

Out[90]:

Dates	Cases	Recoveries	Deaths	Moratality_rate	Recovery_rate
07/14/2020	13419976	7812342	571373	0.042576	0.582143
07/15/2020	13640254	8011859	575571	0.042197	0.587369
07/16/2020	13863064	8217415	579758	0.041820	0.592756
07/17/2020	14088420	8429239	583933	0.041448	0.598310
07/18/2020	14316337	8647565	588097	0.041079	0.604035
07/19/2020	14546830	8872636	592250	0.040713	0.609936
07/20/2020	14779912	9104702	596392	0.040352	0.616019
07/21/2020	15015599	9344022	600522	0.039993	0.622288
07/22/2020	15253904	9590862	604641	0.039638	0.628748
07/23/2020	15494843	9845498	608749	0.039287	0.635405