In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```
!dir
```

```
 O volume na unidade C não tem nome.
 O Número de Série do Volume é 3A4A-84B2

 Pasta de C:\Users\thiag\Documents\Aulas-Python\School of AI\Ano de 2020\a
ula 4 especial\COVID-19-master\csse_covid_19_data\csse_covid_19_time_serie
s

15/04/2020  16:39    <DIR>          .
15/04/2020  16:39    <DIR>          ..
10/04/2020  20:55                 9 .gitignore
15/04/2020  16:39    <DIR>          .ipynb_checkpoints
15/04/2020  16:32            32.711 Deaths_Covid-19.jpg
10/04/2020  20:55               668 README.md
13/04/2020  15:31           385.016 Recuperados-Covid19.ipynb
11/04/2020  22:09           421.527 Serie Temporal_Covid-19.ipynb
10/04/2020  20:55            65.331 time_series_covid19_confirmed_global.c
sv
10/04/2020  20:55           859.468 time_series_covid19_confirmed_US.csv
10/04/2020  20:55            52.985 time_series_covid19_deaths_global.csv
10/04/2020  20:55           859.093 time_series_covid19_deaths_US.csv
10/04/2020  20:55            55.120 time_series_covid19_recovered_global.c
sv
15/04/2020  16:15             9.304 Untitled.ipynb
13/04/2020  14:47                 0 untitled.txt
              12 arquivo(s)      2.741.232 bytes
               3 pasta(s)   85.077.442.560 bytes disponíveis
```

In [3]:

```
DataFrame = pd.read_csv("time_series_covid19_deaths_global.csv")
DataFrame.head()
```

Out[3]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/2
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	
1	NaN	Albania	41.1533	20.1683	0	0	0	0	
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	

5 rows × 84 columns

In [4]:

```
pwd
```

Out[4]:

```
'C:\\Users\\thiag\\Documents\\Aulas-Python\\School of AI\\Ano de 2020\\aul
a 4 especial\\COVID-19-master\\csse_covid_19_data\\csse_covid_19_time_seri
es'
```

In [5]:

```
DataFrame.tail()
```

Out[5]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20
259	Saint Pierre and Miquelon	France	46.885200	-56.315900	0	0	0	0
260	NaN	South Sudan	6.877000	31.307000	0	0	0	0
261	NaN	Western Sahara	24.215500	-12.885800	0	0	0	0
262	NaN	Sao Tome and Principe	0.186360	6.613081	0	0	0	0
263	NaN	Yemen	15.552727	48.516388	0	0	0	0

5 rows × 84 columns

In [6]:

```
DataFrame.iloc[:4,:4]
```

Out[6]:

	Province/State	Country/Region	Lat	Long
0	NaN	Afghanistan	33.0000	65.0000
1	NaN	Albania	41.1533	20.1683
2	NaN	Algeria	28.0339	1.6596
3	NaN	Andorra	42.5063	1.5218

In [7]:

```
DataFrame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264 entries, 0 to 263
Data columns (total 84 columns):
Province/State    82 non-null object
Country/Region    264 non-null object
Lat               264 non-null float64
Long              264 non-null float64
1/22/20           264 non-null int64
1/23/20           264 non-null int64
1/24/20           264 non-null int64
1/25/20           264 non-null int64
1/26/20           264 non-null int64
1/27/20           264 non-null int64
1/28/20           264 non-null int64
1/29/20           264 non-null int64
1/30/20           264 non-null int64
1/31/20           264 non-null int64
2/1/20            264 non-null int64
2/2/20            264 non-null int64
2/3/20            264 non-null int64
2/4/20            264 non-null int64
2/5/20            264 non-null int64
2/6/20            264 non-null int64
2/7/20            264 non-null int64
2/8/20            264 non-null int64
2/9/20            264 non-null int64
2/10/20           264 non-null int64
2/11/20           264 non-null int64
2/12/20           264 non-null int64
2/13/20           264 non-null int64
2/14/20           264 non-null int64
2/15/20           264 non-null int64
2/16/20           264 non-null int64
2/17/20           264 non-null int64
2/18/20           264 non-null int64
2/19/20           264 non-null int64
2/20/20           264 non-null int64
2/21/20           264 non-null int64
2/22/20           264 non-null int64
2/23/20           264 non-null int64
2/24/20           264 non-null int64
2/25/20           264 non-null int64
2/26/20           264 non-null int64
2/27/20           264 non-null int64
2/28/20           264 non-null int64
2/29/20           264 non-null int64
3/1/20            264 non-null int64
3/2/20            264 non-null int64
3/3/20            264 non-null int64
3/4/20            264 non-null int64
3/5/20            264 non-null int64
3/6/20            264 non-null int64
3/7/20            264 non-null int64
3/8/20            264 non-null int64
3/9/20            264 non-null int64
3/10/20           264 non-null int64
3/11/20           264 non-null int64
3/12/20           264 non-null int64
3/13/20           264 non-null int64
3/14/20           264 non-null int64
3/15/20           264 non-null int64
```

```
3/16/20           264 non-null int64
3/17/20           264 non-null int64
3/18/20           264 non-null int64
3/19/20           264 non-null int64
3/20/20           264 non-null int64
3/21/20           264 non-null int64
3/22/20           264 non-null int64
3/23/20           264 non-null int64
3/24/20           264 non-null int64
3/25/20           264 non-null int64
3/26/20           264 non-null int64
3/27/20           264 non-null int64
3/28/20           264 non-null int64
3/29/20           264 non-null int64
3/30/20           264 non-null int64
3/31/20           264 non-null int64
4/1/20            264 non-null int64
4/2/20            264 non-null int64
4/3/20            264 non-null int64
4/4/20            264 non-null int64
4/5/20            264 non-null int64
4/6/20            264 non-null int64
4/7/20            264 non-null int64
4/8/20            264 non-null int64
4/9/20            264 non-null int64
4/10/20           264 non-null int64
dtypes: float64(2), int64(80), object(2)
memory usage: 173.3+ KB
```

In [8]:

```
DataFrame.describe()
```

Out[8]:

	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20
count	264.000000	264.000000	264.000000	264.000000	264.000000	264.000000	264.000000
mean	21.317326	22.168315	0.064394	0.068182	0.098485	0.159091	0.212121
std	24.734994	70.669996	1.046278	1.047853	1.479183	2.462894	3.201783
min	-51.796300	-135.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	6.969250	-20.026050	0.000000	0.000000	0.000000	0.000000	0.000000
50%	23.488100	20.535638	0.000000	0.000000	0.000000	0.000000	0.000000
75%	41.166075	78.750000	0.000000	0.000000	0.000000	0.000000	0.000000
max	71.706900	178.065000	17.000000	17.000000	24.000000	40.000000	52.000000

8 rows × 82 columns

In [9]:

```
DataFrame.sum()
```

Out[9]:

Lat	5627.774017
Long	5852.435261
1/22/20	17.000000
1/23/20	18.000000
1/24/20	26.000000
1/25/20	42.000000
1/26/20	56.000000
1/27/20	82.000000
1/28/20	131.000000
1/29/20	133.000000
1/30/20	171.000000
1/31/20	213.000000
2/1/20	259.000000
2/2/20	362.000000
2/3/20	426.000000
2/4/20	492.000000
2/5/20	564.000000
2/6/20	634.000000
2/7/20	719.000000
2/8/20	806.000000
2/9/20	906.000000
2/10/20	1013.000000
2/11/20	1113.000000
2/12/20	1118.000000
2/13/20	1371.000000
2/14/20	1523.000000
2/15/20	1666.000000
2/16/20	1770.000000
2/17/20	1868.000000
2/18/20	2007.000000
...	...
3/12/20	4720.000000
3/13/20	5404.000000
3/14/20	5819.000000
3/15/20	6440.000000
3/16/20	7126.000000
3/17/20	7905.000000
3/18/20	8733.000000
3/19/20	9867.000000
3/20/20	11299.000000
3/21/20	12973.000000
3/22/20	14651.000000
3/23/20	16505.000000
3/24/20	18625.000000
3/25/20	21181.000000
3/26/20	23970.000000
3/27/20	27198.000000
3/28/20	30652.000000
3/29/20	33925.000000
3/30/20	37582.000000
3/31/20	42107.000000
4/1/20	46809.000000
4/2/20	52983.000000
4/3/20	58787.000000
4/4/20	64606.000000
4/5/20	69374.000000
4/6/20	74565.000000
4/7/20	81865.000000
4/8/20	88338.000000

```
4/9/20      95455.000000
4/10/20    102525.000000
Length: 82, dtype: float64
```

In [10]:

```
Serie_Temporal = DataFrame.copy()
```

In [11]:

```
Serie_Temporal
```

Out[11]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/2
0	NaN	Afghanistan	33.000000	65.000000	0	0	0	
1	NaN	Albania	41.153300	20.168300	0	0	0	
2	NaN	Algeria	28.033900	1.659600	0	0	0	
3	NaN	Andorra	42.506300	1.521800	0	0	0	
4	NaN	Angola	-11.202700	17.873900	0	0	0	
5	NaN	Antigua and Barbuda	17.060800	-61.796400	0	0	0	
6	NaN	Argentina	-38.416100	-63.616700	0	0	0	
7	NaN	Armenia	40.069100	45.038200	0	0	0	
8	Australian Capital Territory	Australia	-35.473500	149.012400	0	0	0	
9	New South Wales	Australia	-33.868800	151.209300	0	0	0	
10	Northern Territory	Australia	-12.463400	130.845600	0	0	0	
11	Queensland	Australia	-28.016700	153.400000	0	0	0	
12	South Australia	Australia	-34.928500	138.600700	0	0	0	
13	Tasmania	Australia	-41.454500	145.970700	0	0	0	
14	Victoria	Australia	-37.813600	144.963100	0	0	0	
15	Western Australia	Australia	-31.950500	115.860500	0	0	0	
16	NaN	Austria	47.516200	14.550100	0	0	0	
17	NaN	Azerbaijan	40.143100	47.576900	0	0	0	
18	NaN	Bahamas	25.034300	-77.396300	0	0	0	
19	NaN	Bahrain	26.027500	50.550000	0	0	0	
20	NaN	Bangladesh	23.685000	90.356300	0	0	0	
21	NaN	Barbados	13.193900	-59.543200	0	0	0	
22	NaN	Belarus	53.709800	27.953400	0	0	0	
23	NaN	Belgium	50.833300	4.000000	0	0	0	
24	NaN	Benin	9.307700	2.315800	0	0	0	
25	NaN	Bhutan	27.514200	90.433600	0	0	0	
26	NaN	Bolivia	-16.290200	-63.588700	0	0	0	
27	NaN	Bosnia and Herzegovina	43.915900	17.679100	0	0	0	
28	NaN	Brazil	-14.235000	-51.925300	0	0	0	
29	NaN	Brunei	4.535300	114.727700	0	0	0	
...	...	...	...	...	...	...	...	
234	NaN	Mozambique	-18.665695	35.529562	0	0	0	
235	NaN	Syria	34.802075	38.996815	0	0	0	

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/2
236	NaN	Timor-Leste	-8.874217	125.727539	0	0	0	
237	NaN	Belize	13.193900	-59.543200	0	0	0	
238	Recovered	Canada	0.000000	0.000000	0	0	0	
239	NaN	Laos	19.856270	102.495496	0	0	0	
240	NaN	Libya	26.335100	17.228331	0	0	0	
241	NaN	West Bank and Gaza	31.952200	35.233200	0	0	0	
242	NaN	Guinea-Bissau	11.803700	-15.180400	0	0	0	
243	NaN	Mali	17.570692	-3.996166	0	0	0	
244	NaN	Saint Kitts and Nevis	17.357822	-62.782998	0	0	0	
245	Northwest Territories	Canada	64.825500	-124.845700	0	0	0	
246	Yukon	Canada	64.282300	-135.000000	0	0	0	
247	NaN	Kosovo	42.602636	20.902977	0	0	0	
248	NaN	Burma	21.916200	95.956000	0	0	0	
249	Anguilla	United Kingdom	18.220600	-63.068600	0	0	0	
250	British Virgin Islands	United Kingdom	18.420700	-64.640000	0	0	0	
251	Turks and Caicos Islands	United Kingdom	21.694000	-71.797900	0	0	0	
252	NaN	MS Zaandam	0.000000	0.000000	0	0	0	
253	NaN	Botswana	-22.328500	24.684900	0	0	0	
254	NaN	Burundi	-3.373100	29.918900	0	0	0	
255	NaN	Sierra Leone	8.460555	-11.779889	0	0	0	
256	Bonaire, Sint Eustatius and Saba	Netherlands	12.178400	-68.238500	0	0	0	
257	NaN	Malawi	-13.254308	34.301525	0	0	0	
258	Falkland Islands (Malvinas)	United Kingdom	-51.796300	-59.523600	0	0	0	
259	Saint Pierre and Miquelon	France	46.885200	-56.315900	0	0	0	
260	NaN	South Sudan	6.877000	31.307000	0	0	0	
261	NaN	Western Sahara	24.215500	-12.885800	0	0	0	
262	NaN	Sao Tome and Principe	0.186360	6.613081	0	0	0	
263	NaN	Yemen	15.552727	48.516388	0	0	0	

264 rows × 84 columns

In [13]:

```
Serie_Temporal.describe()
```

Out[13]:

	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20
count	264.000000	264.000000	264.000000	264.000000	264.000000	264.000000	264.000000
mean	21.317326	22.168315	0.064394	0.068182	0.098485	0.159091	0.212121
std	24.734994	70.669996	1.046278	1.047853	1.479183	2.462894	3.201783
min	-51.796300	-135.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	6.969250	-20.026050	0.000000	0.000000	0.000000	0.000000	0.000000
50%	23.488100	20.535638	0.000000	0.000000	0.000000	0.000000	0.000000
75%	41.166075	78.750000	0.000000	0.000000	0.000000	0.000000	0.000000
max	71.706900	178.065000	17.000000	17.000000	24.000000	40.000000	52.000000

8 rows × 82 columns

In [14]:

```
Serie_Temporal.sum()
```

Out[14]:

Lat	5627.774017
Long	5852.435261
1/22/20	17.000000
1/23/20	18.000000
1/24/20	26.000000
1/25/20	42.000000
1/26/20	56.000000
1/27/20	82.000000
1/28/20	131.000000
1/29/20	133.000000
1/30/20	171.000000
1/31/20	213.000000
2/1/20	259.000000
2/2/20	362.000000
2/3/20	426.000000
2/4/20	492.000000
2/5/20	564.000000
2/6/20	634.000000
2/7/20	719.000000
2/8/20	806.000000
2/9/20	906.000000
2/10/20	1013.000000
2/11/20	1113.000000
2/12/20	1118.000000
2/13/20	1371.000000
2/14/20	1523.000000
2/15/20	1666.000000
2/16/20	1770.000000
2/17/20	1868.000000
2/18/20	2007.000000
...	...
3/12/20	4720.000000
3/13/20	5404.000000
3/14/20	5819.000000
3/15/20	6440.000000
3/16/20	7126.000000
3/17/20	7905.000000
3/18/20	8733.000000
3/19/20	9867.000000
3/20/20	11299.000000
3/21/20	12973.000000
3/22/20	14651.000000
3/23/20	16505.000000
3/24/20	18625.000000
3/25/20	21181.000000
3/26/20	23970.000000
3/27/20	27198.000000
3/28/20	30652.000000
3/29/20	33925.000000
3/30/20	37582.000000
3/31/20	42107.000000
4/1/20	46809.000000
4/2/20	52983.000000
4/3/20	58787.000000
4/4/20	64606.000000
4/5/20	69374.000000
4/6/20	74565.000000
4/7/20	81865.000000
4/8/20	88338.000000

```
4/9/20      95455.000000
4/10/20    102525.000000
Length: 82, dtype: float64
```

In [15]:

```
Serie_Temporal.drop(columns = ["Province/State","Country/Region","Lat","Long"], inplace
= True)
```

In [16]:

```
Serie_Temporal
```

Out[16]:

	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	1/31/20	...
0	0	0	0	0	0	0	0	0	0	0	...
1	0	0	0	0	0	0	0	0	0	0	...
2	0	0	0	0	0	0	0	0	0	0	...
3	0	0	0	0	0	0	0	0	0	0	...
4	0	0	0	0	0	0	0	0	0	0	...
5	0	0	0	0	0	0	0	0	0	0	...
6	0	0	0	0	0	0	0	0	0	0	...
7	0	0	0	0	0	0	0	0	0	0	...
8	0	0	0	0	0	0	0	0	0	0	...
9	0	0	0	0	0	0	0	0	0	0	...
10	0	0	0	0	0	0	0	0	0	0	...
11	0	0	0	0	0	0	0	0	0	0	...
12	0	0	0	0	0	0	0	0	0	0	...
13	0	0	0	0	0	0	0	0	0	0	...
14	0	0	0	0	0	0	0	0	0	0	...
15	0	0	0	0	0	0	0	0	0	0	...
16	0	0	0	0	0	0	0	0	0	0	...
17	0	0	0	0	0	0	0	0	0	0	...
18	0	0	0	0	0	0	0	0	0	0	...
19	0	0	0	0	0	0	0	0	0	0	...
20	0	0	0	0	0	0	0	0	0	0	...
21	0	0	0	0	0	0	0	0	0	0	...
22	0	0	0	0	0	0	0	0	0	0	...
23	0	0	0	0	0	0	0	0	0	0	...
24	0	0	0	0	0	0	0	0	0	0	...
25	0	0	0	0	0	0	0	0	0	0	...
26	0	0	0	0	0	0	0	0	0	0	...
27	0	0	0	0	0	0	0	0	0	0	...
28	0	0	0	0	0	0	0	0	0	0	...
29	0	0	0	0	0	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...	...
234	0	0	0	0	0	0	0	0	0	0	...
235	0	0	0	0	0	0	0	0	0	0	...
236	0	0	0	0	0	0	0	0	0	0	...
237	0	0	0	0	0	0	0	0	0	0	...
238	0	0	0	0	0	0	0	0	0	0	...
239	0	0	0	0	0	0	0	0	0	0	...

	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	1/31/20	...
240	0	0	0	0	0	0	0	0	0	0	...
241	0	0	0	0	0	0	0	0	0	0	...
242	0	0	0	0	0	0	0	0	0	0	...
243	0	0	0	0	0	0	0	0	0	0	...
244	0	0	0	0	0	0	0	0	0	0	...
245	0	0	0	0	0	0	0	0	0	0	...
246	0	0	0	0	0	0	0	0	0	0	...
247	0	0	0	0	0	0	0	0	0	0	...
248	0	0	0	0	0	0	0	0	0	0	...
249	0	0	0	0	0	0	0	0	0	0	...
250	0	0	0	0	0	0	0	0	0	0	...
251	0	0	0	0	0	0	0	0	0	0	...
252	0	0	0	0	0	0	0	0	0	0	...
253	0	0	0	0	0	0	0	0	0	0	...
254	0	0	0	0	0	0	0	0	0	0	...
255	0	0	0	0	0	0	0	0	0	0	...
256	0	0	0	0	0	0	0	0	0	0	...
257	0	0	0	0	0	0	0	0	0	0	...
258	0	0	0	0	0	0	0	0	0	0	...
259	0	0	0	0	0	0	0	0	0	0	...
260	0	0	0	0	0	0	0	0	0	0	...
261	0	0	0	0	0	0	0	0	0	0	...
262	0	0	0	0	0	0	0	0	0	0	...
263	0	0	0	0	0	0	0	0	0	0	...

264 rows × 80 columns

In [17]:

```
Serie_Temporal.iloc[:4,:4]
```

Out[17]:

	1/22/20	1/23/20	1/24/20	1/25/20
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

In [23]:

```
Serie_Geral = Serie_Temporal.sum().copy()
```

In [25]:

```
Serie_Geral = pd.DataFrame(Serie_Geral)
Serie_Geral.rename(columns = {0:"y"}, inplace = True)
```

In [26]:

```
Serie_Geral
```

Out[26]:

	y
1/22/20	17
1/23/20	18
1/24/20	26
1/25/20	42
1/26/20	56
1/27/20	82
1/28/20	131
1/29/20	133
1/30/20	171
1/31/20	213
2/1/20	259
2/2/20	362
2/3/20	426
2/4/20	492
2/5/20	564
2/6/20	634
2/7/20	719
2/8/20	806
2/9/20	906
2/10/20	1013
2/11/20	1113
2/12/20	1118
2/13/20	1371
2/14/20	1523
2/15/20	1666
2/16/20	1770
2/17/20	1868
2/18/20	2007
2/19/20	2122
2/20/20	2247
...	...
3/12/20	4720
3/13/20	5404
3/14/20	5819
3/15/20	6440
3/16/20	7126
3/17/20	7905

	y
3/18/20	8733
3/19/20	9867
3/20/20	11299
3/21/20	12973
3/22/20	14651
3/23/20	16505
3/24/20	18625
3/25/20	21181
3/26/20	23970
3/27/20	27198
3/28/20	30652
3/29/20	33925
3/30/20	37582
3/31/20	42107
4/1/20	46809
4/2/20	52983
4/3/20	58787
4/4/20	64606
4/5/20	69374
4/6/20	74565
4/7/20	81865
4/8/20	88338
4/9/20	95455
4/10/20	102525

80 rows × 1 columns

In [27]:

```
Serie_Geral.head()
```

Out[27]:

	y
1/22/20	17
1/23/20	18
1/24/20	26
1/25/20	42
1/26/20	56

In [22]:

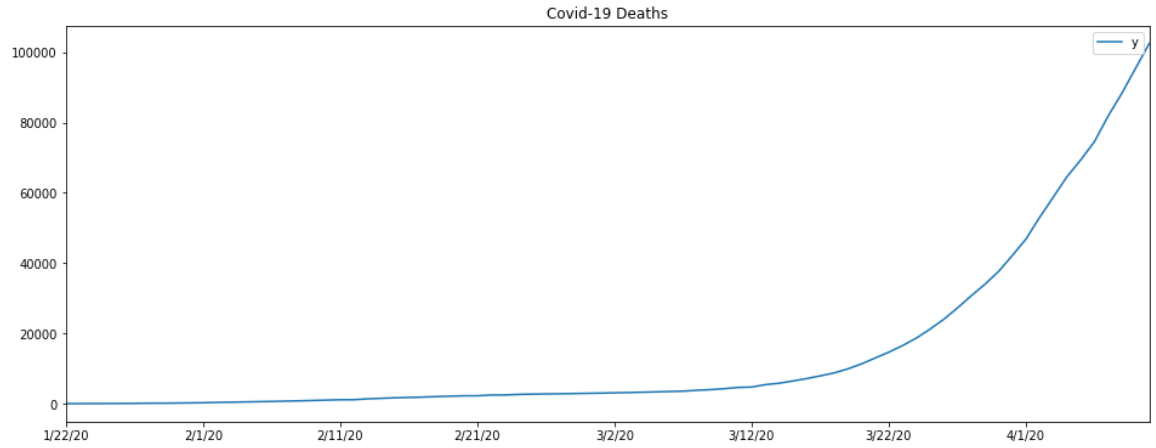
```
Serie_Geral.head()
```

Out[22]:

	y
1/22/20	17
1/23/20	18
1/24/20	26
1/25/20	42
1/26/20	56

In [28]:

```
Serie_Geral.plot(figsize=(16,6))
plt.savefig("Deaths-Covid-19.jpg")
plt.title("Covid-19 Deaths")
plt.show()
```



In [29]:

```
Serie_Geral.index = pd.to_datetime(Serie_Geral.index)
```

In [30]:

```
Serie_Geral.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 80 entries, 2020-01-22 to 2020-04-10
Data columns (total 1 columns):
y    80 non-null int64
dtypes: int64(1)
memory usage: 1.2 KB
```

In [31]:

```
tempos = list(range(Serie_Geral.shape[0]))
Serie_Geral["tempos"] = tempos
```

In [32]:

```
Serie_Geral.index.day
```

Out[32]:

```
Int64Index([22, 23, 24, 25, 26, 27, 28, 29, 30, 31,  1,  2,  3,  4,  5,
6,  7,
             8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 2
3, 24,
            25, 26, 27, 28, 29,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 1
1, 12,
            13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 2
8, 29,
            30, 31,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10],
           dtype='int64')
```

In [35]:

```
Serie_Geral["Dia"] = Serie_Geral.index.day
```

In [36]:

```
Serie_Geral
```

Out[36]:

	y	tempos	Dia
2020-01-22	17	0	22
2020-01-23	18	1	23
2020-01-24	26	2	24
2020-01-25	42	3	25
2020-01-26	56	4	26
2020-01-27	82	5	27
2020-01-28	131	6	28
2020-01-29	133	7	29
2020-01-30	171	8	30
2020-01-31	213	9	31
2020-02-01	259	10	1
2020-02-02	362	11	2
2020-02-03	426	12	3
2020-02-04	492	13	4
2020-02-05	564	14	5
2020-02-06	634	15	6
2020-02-07	719	16	7
2020-02-08	806	17	8
2020-02-09	906	18	9
2020-02-10	1013	19	10
2020-02-11	1113	20	11
2020-02-12	1118	21	12
2020-02-13	1371	22	13
2020-02-14	1523	23	14
2020-02-15	1666	24	15
2020-02-16	1770	25	16
2020-02-17	1868	26	17
2020-02-18	2007	27	18
2020-02-19	2122	28	19
2020-02-20	2247	29	20
...	...	...	...
2020-03-12	4720	50	12
2020-03-13	5404	51	13
2020-03-14	5819	52	14
2020-03-15	6440	53	15
2020-03-16	7126	54	16
2020-03-17	7905	55	17

	y	tempos	Dia
2020-03-18	8733	56	18
2020-03-19	9867	57	19
2020-03-20	11299	58	20
2020-03-21	12973	59	21
2020-03-22	14651	60	22
2020-03-23	16505	61	23
2020-03-24	18625	62	24
2020-03-25	21181	63	25
2020-03-26	23970	64	26
2020-03-27	27198	65	27
2020-03-28	30652	66	28
2020-03-29	33925	67	29
2020-03-30	37582	68	30
2020-03-31	42107	69	31
2020-04-01	46809	70	1
2020-04-02	52983	71	2
2020-04-03	58787	72	3
2020-04-04	64606	73	4
2020-04-05	69374	74	5
2020-04-06	74565	75	6
2020-04-07	81865	76	7
2020-04-08	88338	77	8
2020-04-09	95455	78	9
2020-04-10	102525	79	10

80 rows × 3 columns

In [37]:

```
Serie_Geral["Dia da Semana"] = Serie_Geral.index.dayofweek
```

In [38]:

```
Serie_Geral
```

Out[38]:

	y	tempos	Dia	Dia da Semana
2020-01-22	17	0	22	2
2020-01-23	18	1	23	3
2020-01-24	26	2	24	4
2020-01-25	42	3	25	5
2020-01-26	56	4	26	6
2020-01-27	82	5	27	0
2020-01-28	131	6	28	1
2020-01-29	133	7	29	2
2020-01-30	171	8	30	3
2020-01-31	213	9	31	4
2020-02-01	259	10	1	5
2020-02-02	362	11	2	6
2020-02-03	426	12	3	0
2020-02-04	492	13	4	1
2020-02-05	564	14	5	2
2020-02-06	634	15	6	3
2020-02-07	719	16	7	4
2020-02-08	806	17	8	5
2020-02-09	906	18	9	6
2020-02-10	1013	19	10	0
2020-02-11	1113	20	11	1
2020-02-12	1118	21	12	2
2020-02-13	1371	22	13	3
2020-02-14	1523	23	14	4
2020-02-15	1666	24	15	5
2020-02-16	1770	25	16	6
2020-02-17	1868	26	17	0
2020-02-18	2007	27	18	1
2020-02-19	2122	28	19	2
2020-02-20	2247	29	20	3
...	...	...	...	...
2020-03-12	4720	50	12	3
2020-03-13	5404	51	13	4
2020-03-14	5819	52	14	5
2020-03-15	6440	53	15	6
2020-03-16	7126	54	16	0
2020-03-17	7905	55	17	1

	y	tempos	Dia	Dia da Semana
2020-03-18	8733	56	18	2
2020-03-19	9867	57	19	3
2020-03-20	11299	58	20	4
2020-03-21	12973	59	21	5
2020-03-22	14651	60	22	6
2020-03-23	16505	61	23	0
2020-03-24	18625	62	24	1
2020-03-25	21181	63	25	2
2020-03-26	23970	64	26	3
2020-03-27	27198	65	27	4
2020-03-28	30652	66	28	5
2020-03-29	33925	67	29	6
2020-03-30	37582	68	30	0
2020-03-31	42107	69	31	1
2020-04-01	46809	70	1	2
2020-04-02	52983	71	2	3
2020-04-03	58787	72	3	4
2020-04-04	64606	73	4	5
2020-04-05	69374	74	5	6
2020-04-06	74565	75	6	0
2020-04-07	81865	76	7	1
2020-04-08	88338	77	8	2
2020-04-09	95455	78	9	3
2020-04-10	102525	79	10	4

80 rows × 4 columns

In [39]:

```
Serie_Geral.head()
```

Out[39]:

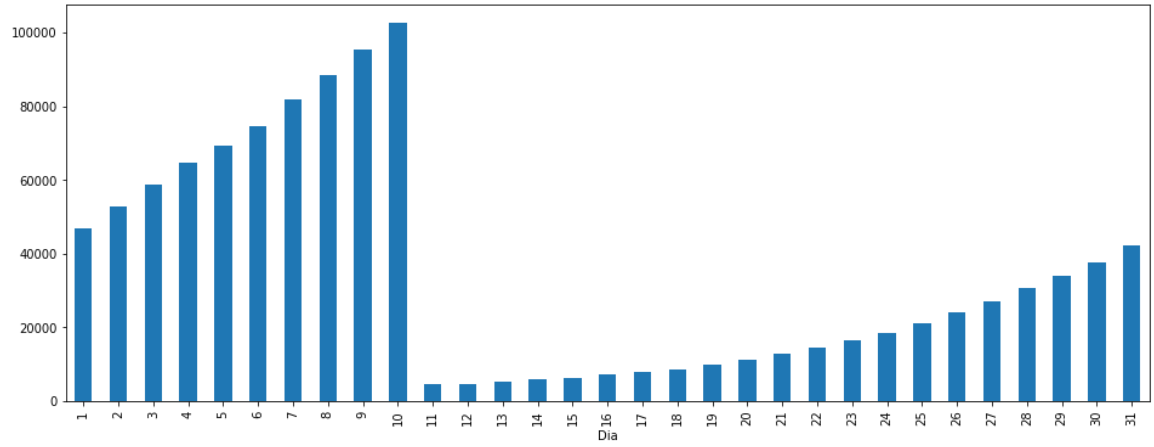
	y	tempos	Dia	Dia da Semana
2020-01-22	17	0	22	2
2020-01-23	18	1	23	3
2020-01-24	26	2	24	4
2020-01-25	42	3	25	5
2020-01-26	56	4	26	6

In [40]:

```
Serie_Geral.groupby("Dia")["y"].max().plot.bar(figsize=(16,6))
```

Out[40]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d24071ec18>
```

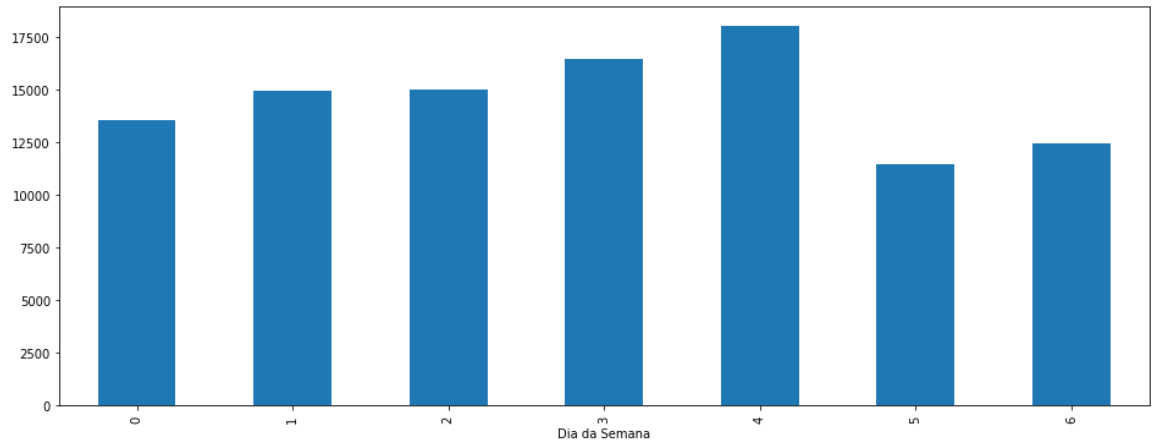


In [41]:

```
Serie_Geral.groupby("Dia da Semana")["y"].mean().plot.bar(figsize=(16,6))
```

Out[41]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d240e80f98>
```



In [42]:

```
Serie_Geral.shape[0]
```

Out[42]:

```
80
```

In [43]:

```
Taxa_Treino = 0.9
X_Treino = Serie_Geral.iloc[:round(Serie_Geral.shape[0] * Taxa_Treino), 1:]
X_Treino
```

Out[43]:

	tempos	Dia	Dia da Semana
2020-01-22	0	22	2
2020-01-23	1	23	3
2020-01-24	2	24	4
2020-01-25	3	25	5
2020-01-26	4	26	6
2020-01-27	5	27	0
2020-01-28	6	28	1
2020-01-29	7	29	2
2020-01-30	8	30	3
2020-01-31	9	31	4
2020-02-01	10	1	5
2020-02-02	11	2	6
2020-02-03	12	3	0
2020-02-04	13	4	1
2020-02-05	14	5	2
2020-02-06	15	6	3
2020-02-07	16	7	4
2020-02-08	17	8	5
2020-02-09	18	9	6
2020-02-10	19	10	0
2020-02-11	20	11	1
2020-02-12	21	12	2
2020-02-13	22	13	3
2020-02-14	23	14	4
2020-02-15	24	15	5
2020-02-16	25	16	6
2020-02-17	26	17	0
2020-02-18	27	18	1
2020-02-19	28	19	2
2020-02-20	29	20	3
...	...	...	...
2020-03-04	42	4	2
2020-03-05	43	5	3
2020-03-06	44	6	4
2020-03-07	45	7	5
2020-03-08	46	8	6
2020-03-09	47	9	0

	tempos	Dia	Dia da Semana
2020-03-10	48	10	1
2020-03-11	49	11	2
2020-03-12	50	12	3
2020-03-13	51	13	4
2020-03-14	52	14	5
2020-03-15	53	15	6
2020-03-16	54	16	0
2020-03-17	55	17	1
2020-03-18	56	18	2
2020-03-19	57	19	3
2020-03-20	58	20	4
2020-03-21	59	21	5
2020-03-22	60	22	6
2020-03-23	61	23	0
2020-03-24	62	24	1
2020-03-25	63	25	2
2020-03-26	64	26	3
2020-03-27	65	27	4
2020-03-28	66	28	5
2020-03-29	67	29	6
2020-03-30	68	30	0
2020-03-31	69	31	1
2020-04-01	70	1	2
2020-04-02	71	2	3

72 rows × 3 columns

In [44]:

```
Serie_Geral["Anterior"] = Serie_Geral["y"].shift(5)
Serie_Geral = Serie_Geral.iloc[5:]
Serie_Geral
```

Out[44]:

	y	tempos	Dia	Dia da Semana	Anterior
2020-01-27	82	5	27	0	17.0
2020-01-28	131	6	28	1	18.0
2020-01-29	133	7	29	2	26.0
2020-01-30	171	8	30	3	42.0
2020-01-31	213	9	31	4	56.0
2020-02-01	259	10	1	5	82.0
2020-02-02	362	11	2	6	131.0
2020-02-03	426	12	3	0	133.0
2020-02-04	492	13	4	1	171.0
2020-02-05	564	14	5	2	213.0
2020-02-06	634	15	6	3	259.0
2020-02-07	719	16	7	4	362.0
2020-02-08	806	17	8	5	426.0
2020-02-09	906	18	9	6	492.0
2020-02-10	1013	19	10	0	564.0
2020-02-11	1113	20	11	1	634.0
2020-02-12	1118	21	12	2	719.0
2020-02-13	1371	22	13	3	806.0
2020-02-14	1523	23	14	4	906.0
2020-02-15	1666	24	15	5	1013.0
2020-02-16	1770	25	16	6	1113.0
2020-02-17	1868	26	17	0	1118.0
2020-02-18	2007	27	18	1	1371.0
2020-02-19	2122	28	19	2	1523.0
2020-02-20	2247	29	20	3	1666.0
2020-02-21	2251	30	21	4	1770.0
2020-02-22	2458	31	22	5	1868.0
2020-02-23	2469	32	23	6	2007.0
2020-02-24	2629	33	24	0	2122.0
2020-02-25	2708	34	25	1	2247.0
...	...	...	...	...	...
2020-03-12	4720	50	12	3	3558.0
2020-03-13	5404	51	13	4	3802.0
2020-03-14	5819	52	14	5	3988.0
2020-03-15	6440	53	15	6	4262.0
2020-03-16	7126	54	16	0	4615.0
2020-03-17	7905	55	17	1	4720.0

	y	tempos	Dia	Dia da Semana	Anterior
2020-03-18	8733	56	18	2	5404.0
2020-03-19	9867	57	19	3	5819.0
2020-03-20	11299	58	20	4	6440.0
2020-03-21	12973	59	21	5	7126.0
2020-03-22	14651	60	22	6	7905.0
2020-03-23	16505	61	23	0	8733.0
2020-03-24	18625	62	24	1	9867.0
2020-03-25	21181	63	25	2	11299.0
2020-03-26	23970	64	26	3	12973.0
2020-03-27	27198	65	27	4	14651.0
2020-03-28	30652	66	28	5	16505.0
2020-03-29	33925	67	29	6	18625.0
2020-03-30	37582	68	30	0	21181.0
2020-03-31	42107	69	31	1	23970.0
2020-04-01	46809	70	1	2	27198.0
2020-04-02	52983	71	2	3	30652.0
2020-04-03	58787	72	3	4	33925.0
2020-04-04	64606	73	4	5	37582.0
2020-04-05	69374	74	5	6	42107.0
2020-04-06	74565	75	6	0	46809.0
2020-04-07	81865	76	7	1	52983.0
2020-04-08	88338	77	8	2	58787.0
2020-04-09	95455	78	9	3	64606.0
2020-04-10	102525	79	10	4	69374.0

75 rows × 5 columns

In [45]:

```
Taxa_Treino = 0.95
X_Treino = Serie_Geral.iloc[:round(Serie_Geral.shape[0] * Taxa_Treino), 1:]
X_Teste = Serie_Geral.iloc[round(Serie_Geral.shape[0] * Taxa_Treino):, 1:]
Y_Treino = Serie_Geral.iloc[:round(Serie_Geral.shape[0] * Taxa_Treino), 0]
Y_Teste = Serie_Geral.iloc[round(Serie_Geral.shape[0] * Taxa_Treino):, 0]
```

In [46]:

```
Y_Teste
```

Out[46]:

```
2020-04-07     81865
2020-04-08     88338
2020-04-09     95455
2020-04-10    102525
Name: y, dtype: int64
```

In [47]:

```
Y_Treino
```

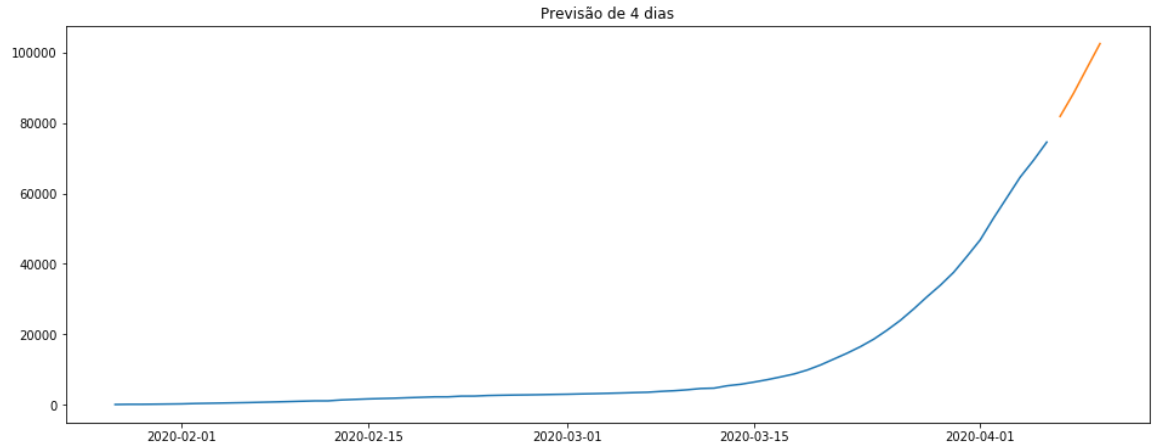
Out[47]:

```
2020-01-27       82
2020-01-28      131
2020-01-29      133
2020-01-30      171
2020-01-31      213
2020-02-01      259
2020-02-02      362
2020-02-03      426
2020-02-04      492
2020-02-05      564
2020-02-06      634
2020-02-07      719
2020-02-08      806
2020-02-09      906
2020-02-10     1013
2020-02-11     1113
2020-02-12     1118
2020-02-13     1371
2020-02-14     1523
2020-02-15     1666
2020-02-16     1770
2020-02-17     1868
2020-02-18     2007
2020-02-19     2122
2020-02-20     2247
2020-02-21     2251
2020-02-22     2458
2020-02-23     2469
2020-02-24     2629
2020-02-25     2708
               ...
2020-03-08     3802
2020-03-09     3988
2020-03-10     4262
2020-03-11     4615
2020-03-12     4720
2020-03-13     5404
2020-03-14     5819
2020-03-15     6440
2020-03-16     7126
2020-03-17     7905
2020-03-18     8733
2020-03-19     9867
2020-03-20    11299
2020-03-21    12973
2020-03-22    14651
2020-03-23    16505
2020-03-24    18625
2020-03-25    21181
2020-03-26    23970
2020-03-27    27198
2020-03-28    30652
2020-03-29    33925
2020-03-30    37582
2020-03-31    42107
2020-04-01    46809
2020-04-02    52983
2020-04-03    58787
2020-04-04    64606
```

```
2020-04-05    69374
2020-04-06    74565
Name: y, Length: 71, dtype: int64
```

In [48]:

```
plt.figure(figsize = (16,6))
plt.title(f'Previsão de {len(Y_Teste)} dias')
plt.plot(Y_Treino)
plt.plot(Y_Teste)
plt.show()
```



In [49]:

```
from sklearn.linear_model import LinearRegression
```

In [50]:

```
modelo = LinearRegression().fit(X_Treino, Y_Treino)
modelo
```

Out[50]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=F
alse)
```

In [51]:

```
modelo.score(X_Treino, Y_Treino)
```

Out[51]:

```
0.9958696803561979
```

In [52]:

```
modelo.coef_, modelo.intercept_
```

Out[52]:

```
(array([ 2.43391779, 50.00301416, 46.30578111,  1.71092836]),
 -1312.006615532544)
```

In [53]:

```
Previsto = modelo.predict(X_Teste)
Previsto.round()
```

Out[53]:

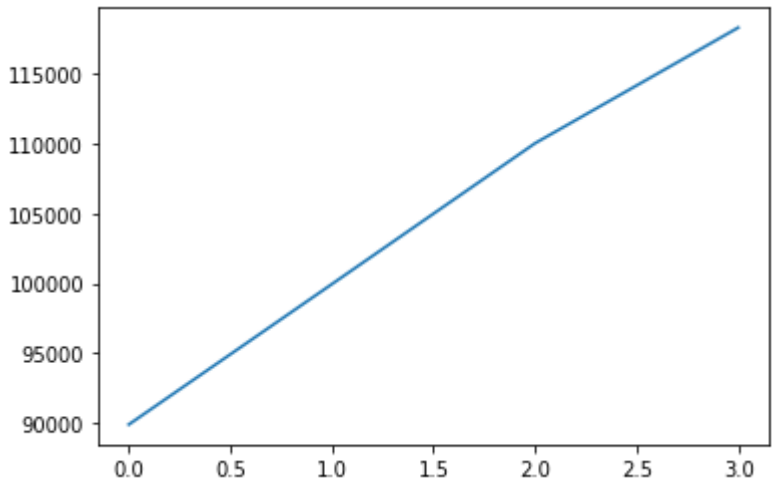
```
array([ 89919.,  99948., 110003., 118259.])
```

In [54]:

```
plt.plot(Previsto)
```

Out[54]:

```
[<matplotlib.lines.Line2D at 0x1d243586c50>]
```



In [55]:

```
df_Previsao = X_Teste.copy()
df_Previsao[0] = Previsto
df_Previsao
```

Out[55]:

	tempos	Dia	Dia da Semana	Anterior	0
2020-04-07	76	7	1	52983.0	89919.415486
2020-04-08	77	8	2	58787.0	99948.386420
2020-04-09	78	9	3	64606.0	110003.021278
2020-04-10	79	10	4	69374.0	118259.470427

In [56]:

```
plt.figure(figsize = (16,6))
plt.title(f'Previsão de {len(Y_Teste)} dias {modelo.score(X_Treino, Y_Treino)}')
plt.plot(Y_Treino)
plt.plot(Y_Teste)
plt.plot(df_Previsao)
plt.savefig("Deaths-Covid19vs1.jpg")
plt.show()
```