In [1]:

```
import pandas as pd
import numpy as np
from statsmodels.tsa.arima_model import ARIMA
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.metrics import mean_squared_error
from statsmodels.tools.eval_measures import rmse
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:

```
no_confirmed = pd.read_csv("time_series_covid19_confirmed_global.csv")
no_deaths =  pd.read_csv("time_series_covid19_deaths_global.csv")
no_recovered =  pd.read_csv("time_series_covid19_recovered_global.csv")
```

In [3]:

```
no_confirmed.rename(columns={'Country/Region':'Country'}, inplace=True)
no_recovered.rename(columns={'Country/Region':'Country'}, inplace=True)
no_deaths.rename(columns={'Country/Region':'Country'}, inplace=True)
```

In [4]:

```
no_confirmed = no_confirmed.melt(id_vars=["Province/State","Country","Lat","Long"],var_
no_deaths = no_deaths.melt(id_vars=["Province/State","Country","Lat","Long"],var_name =
no_recovered = no_recovered.melt(id_vars=["Province/State","Country","Lat","Long"],var_
```

In [5]:

```
no_confirmed["Deaths"] = no_deaths.Deaths
no_confirmed["Recovered"] = no_recovered.Recovered
```

In [6]:

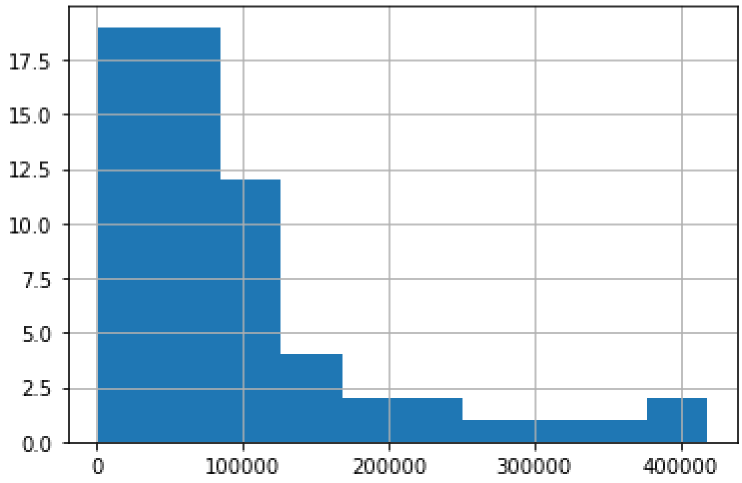
```
X = no_confirmed
X.Date = pd.to_datetime(X.Date)
```

In [7]:

```
confirmed = X.groupby('Date').sum()['Confirmed']
deaths = X.groupby('Date').sum()['Deaths']
recovered = X.groupby('Date').sum()['Recovered']
```

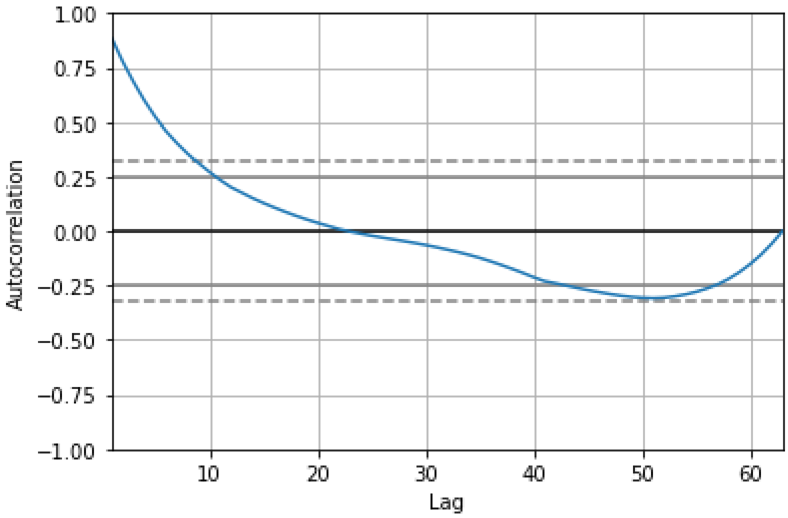
In [8]:

```
from pandas import read_csv
from matplotlib import pyplot
confirmed.hist()
pyplot.show()
```



In [9]:

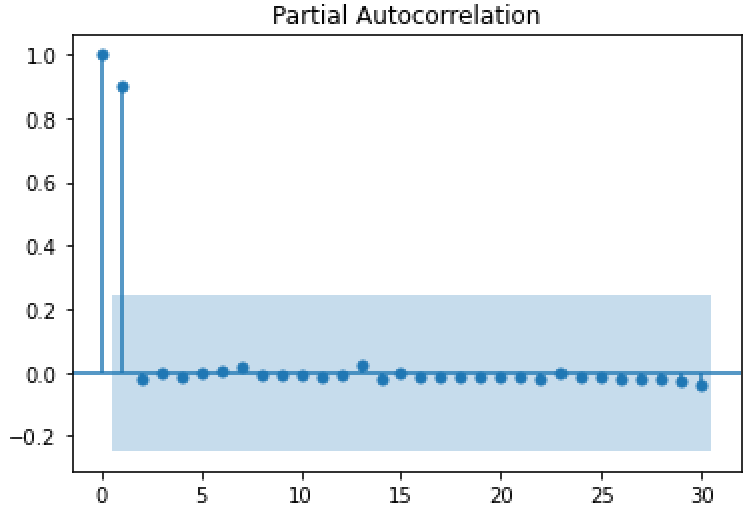
```
import pandas
from pandas.plotting import autocorrelation_plot
autocorrelation_plot(confirmed)
pyplot.show()
```

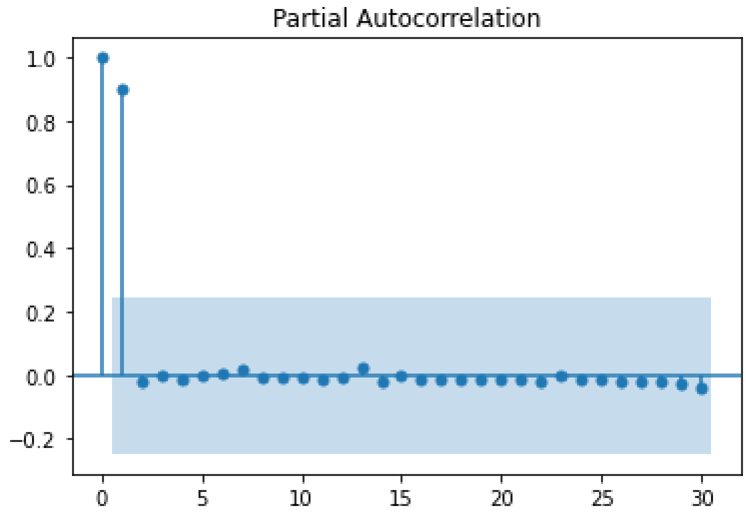


In [10]:

```
import statsmodels.api as sm
sm.graphics.tsa.plot_pacf(confirmed,lags='30')
```

Out[10]:





In [11]:

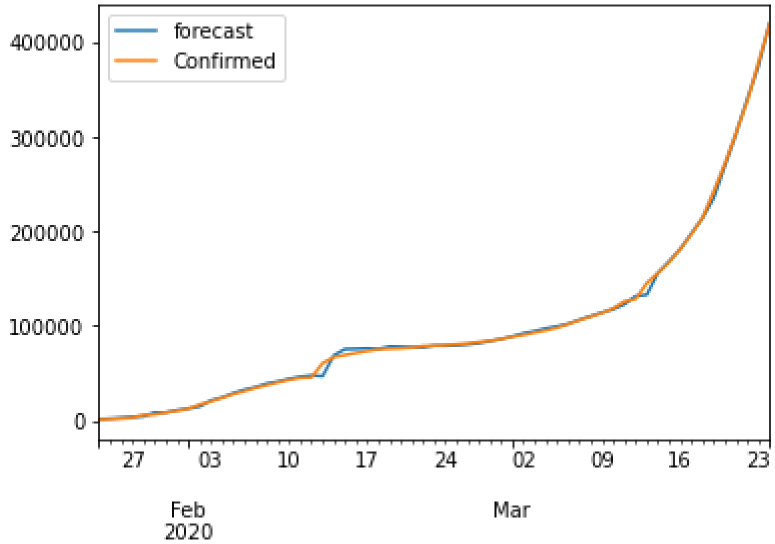
```
con = ARIMA(confirmed, order=(6,2,2))
fitting_con = con.fit(disp=0)
print(fitting_con.summary())
```

```
                             ARIMA Model Results
==============================================================================
====
Dep. Variable:            D2.Confirmed   No. Observations:
61
Model:                  ARIMA(6, 2, 2)   Log Likelihood                   -57
7.718
Method:                        css-mle   S.D. of innovations              302
8.549
Date:                 Fri, 14 May 2021   AIC                              117
5.436
Time:                         09:59:03   BIC                              119
6.545
Sample:                     01-24-2020   HQIC                             118
3.709
                          - 03-24-2020
==============================================================================
============
                        coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
------------
const               656.3385    604.876      1.085      0.278    -529.196
1841.873
ar.L1.D2.Confirmed    1.2910      0.130      9.900      0.000       1.035
1.547
ar.L2.D2.Confirmed   -0.2019      0.210     -0.961      0.337      -0.614
0.210
ar.L3.D2.Confirmed   -0.1114      0.212     -0.526      0.599      -0.527
0.304
ar.L4.D2.Confirmed   -0.0913      0.213     -0.429      0.668      -0.508
0.326
ar.L5.D2.Confirmed   -0.1052      0.210     -0.501      0.616      -0.516
0.306
ar.L6.D2.Confirmed    0.1200      0.143      0.838      0.402      -0.161
0.401
ma.L1.D2.Confirmed   -1.8377      0.153    -11.986      0.000      -2.138
-1.537
ma.L2.D2.Confirmed    0.9994      0.164      6.089      0.000       0.678
1.321
                                    Roots
==============================================================================
===
                  Real          Imaginary           Modulus         Freque
ncy
------------------------------------------------------------------------------
---
AR.1           -1.6644           -0.0000j            1.6644           -0.5
000
AR.2           -0.4981           -1.5297j            1.6088           -0.3
001
AR.3           -0.4981           +1.5297j            1.6088            0.3
001
AR.4            1.1378           -0.4886j            1.2383           -0.0
646
```

```
AR.5              1.1378          +0.4886j          1.2383            0.0
646
AR.6              1.2615          -0.0000j          1.2615           -0.0
000
MA.1              0.9194          -0.3941j          1.0003           -0.0
644
MA.2              0.9194          +0.3941j          1.0003            0.0
644
-----------------------------------------------------------------------------
---
```

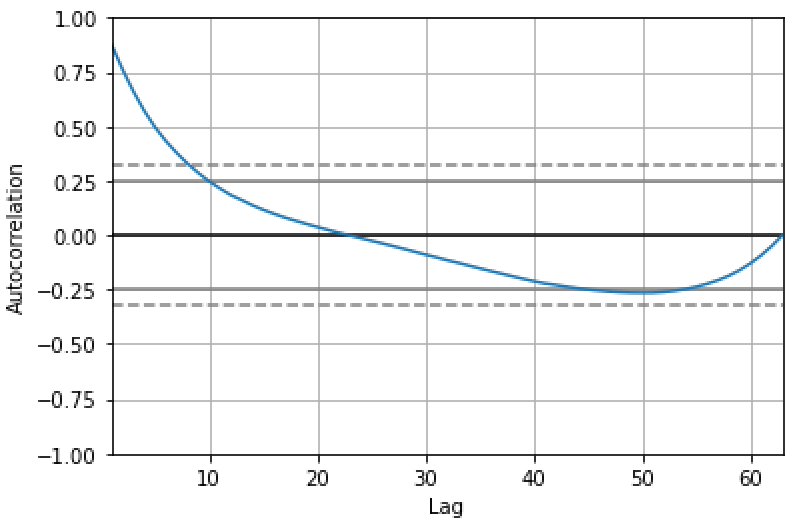
In [12]:

```
fitting_con.plot_predict(dynamic=False)
plt.show()
```



In [13]:

```
import pandas
from pandas.plotting import autocorrelation_plot
autocorrelation_plot(deaths)
pyplot.show()
```



In [14]:

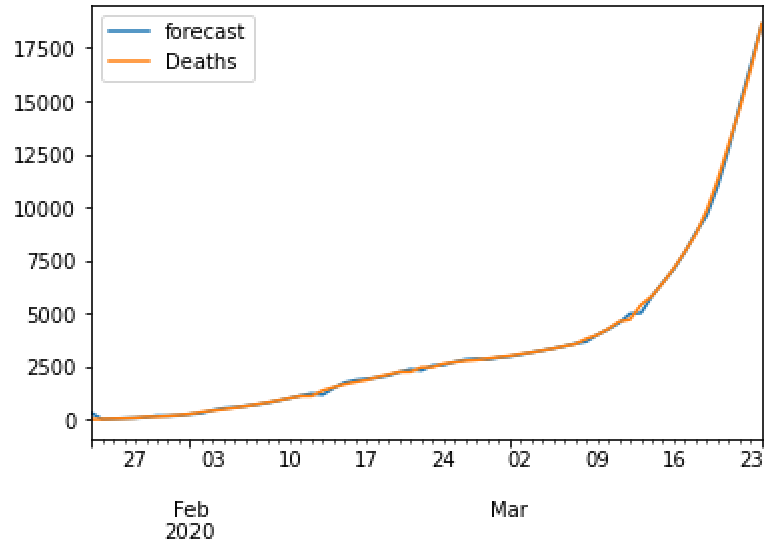
```
death = ARIMA(deaths, order=(6,1,0))
fitting_death = death.fit(disp=0)
print(fitting_death.summary())
```

```
                             ARIMA Model Results
==============================================================================
====
Dep. Variable:               D.Deaths   No. Observations:
62
Model:                 ARIMA(6, 1, 0)   Log Likelihood                    -38
4.967
Method:                       css-mle   S.D. of innovations                10
1.520
Date:                Fri, 14 May 2021   AIC                                78
5.934
Time:                        09:59:06   BIC                                80
2.951
Sample:                    01-23-2020   HQIC                               79
2.615
                         - 03-24-2020
==============================================================================
========
                    coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
--------
const           300.2159        nan        nan        nan         nan
nan
ar.L1.D.Deaths    0.5306        nan        nan        nan         nan
nan
ar.L2.D.Deaths    0.7470        nan        nan        nan         nan
nan
ar.L3.D.Deaths    0.2028        nan        nan        nan         nan
nan
ar.L4.D.Deaths   -0.0580        nan        nan        nan         nan
nan
ar.L5.D.Deaths   -0.1488        nan        nan        nan         nan
nan
ar.L6.D.Deaths   -0.2735        nan        nan        nan         nan
nan
                                    Roots
==============================================================================
===
                  Real          Imaginary           Modulus         Freque
ncy
------------------------------------------------------------------------------
---
AR.1            1.0001           -0.0000j            1.0001           -0.0
000
AR.2            1.0033           -0.0000j            1.0033           -0.0
000
AR.3           -1.1732           -0.5781j            1.3079           -0.4
271
AR.4           -1.1732           +0.5781j            1.3079            0.4
271
AR.5           -0.1005           -1.4559j            1.4594           -0.2
610
AR.6           -0.1005           +1.4559j            1.4594            0.2
610
```

```
--------------------------------------------------------------------------
---
```

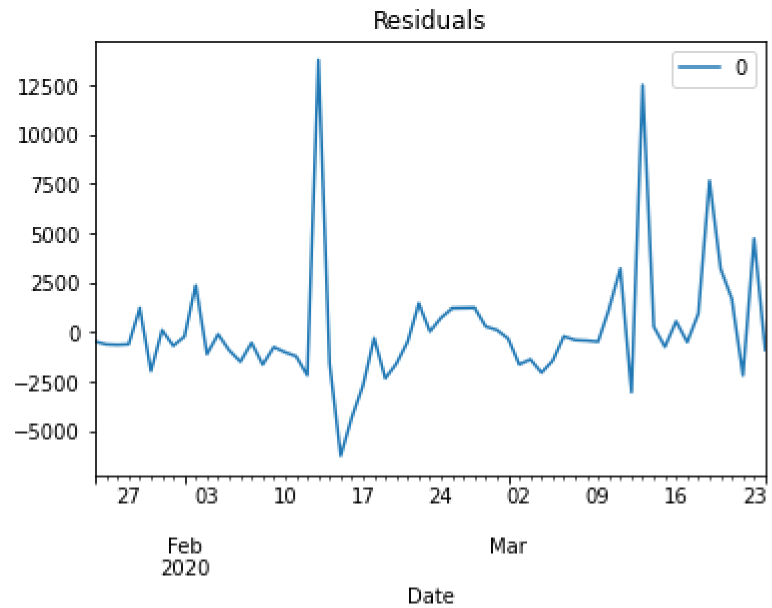
In [15]:

```python
fitting_death.plot_predict(dynamic=False)
plt.show()
```



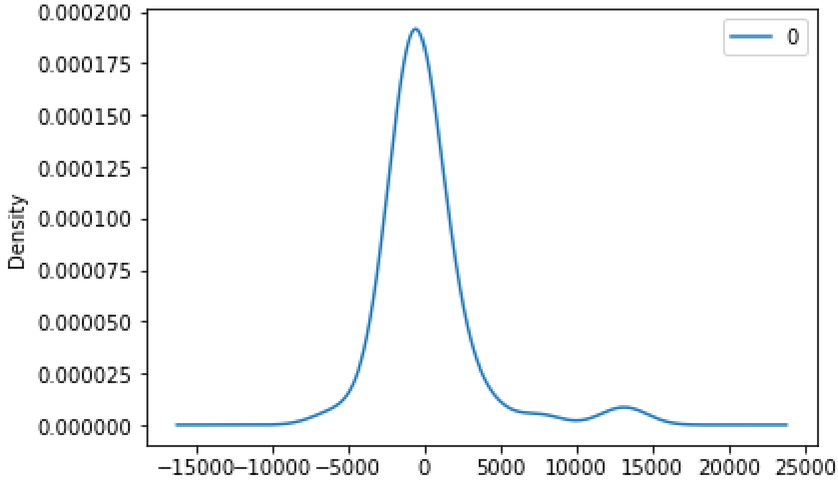
In [16]:

```python
residuals = pd.DataFrame(fitting_con.resid)
residuals.plot(title="Residuals")
pyplot.show()
```



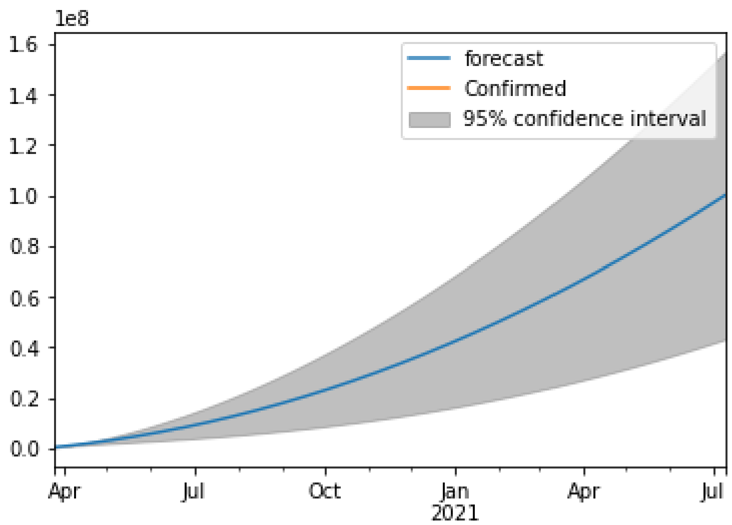
In [17]:

```
residuals.plot(kind="kde",ylabel="Density")
pyplot.show()
```



In [18]:

```
start_index = '2020-03-25'
end_index = '2021-07-10'
forecast = fitting_con.plot_predict(start=start_index, end=end_index)
#plt.show()
```



In [19]:

```
arima_prediction = fitting_con.predict(dynamic=False)
arima_prediction
```

Out[19]:

```
2020-01-24     656.338482
2020-01-25     813.528750
2020-01-26     821.344159
2020-01-27     729.979546
2020-01-28     631.988677
                  ...
2020-03-20   -1524.624216
2020-03-21    1224.438624
2020-03-22    2393.517191
2020-03-23    4168.957077
2020-03-24    -434.023733
Freq: D, Length: 61, dtype: float64
```

In [20]:

```
y=confirmed[2:]
x=np.cumsum(arima_prediction.values)
print(x)
```

```
[  656.33848168  1469.86723152  2291.21139098  3021.19093703
  3653.17961367  3544.86898584  4916.64085909  5222.60353534
  5854.66441686  6145.81436884  5589.66257292  6604.75079133
  7251.65627192  8152.66729925  9119.20462516  9881.91151908
 10930.97715201 11519.15871223 12157.32897627 12705.73214098
 13675.84715734  6611.2127623   8481.33036753 12823.55105426
 15353.71158981 15492.22544277 16442.63294873 18076.46331559
 18616.25037838 18301.48876863 16889.95249754 16367.45483597
 15400.71880777 14317.75262019 13463.08730787 13173.22153306
 13587.97016204 14357.37551633 15558.22673631 17527.30585773
 19308.49005109 21224.1696324  22577.26897921 23090.29843605
 23444.78032072 23709.15989581 23768.90252289 22808.94909967
 21044.7360424  22935.8462124  16689.08814507 17878.11740764
 20088.68161255 22041.57167631 23271.4011582  25658.20015757
 24133.57594128 25358.01456562 27751.53175617 31920.48883336
 31486.46510029]
```

In [21]:

```
mse = rmse(y,x)
print('RMSE: %f' % mse)
```

```
RMSE: 118943.297054
```

In [22]:

```
MAE=metrics.mean_absolute_error(y,x)
print(f'Mean Absolute Error:{MAE}')
```

```
Mean Absolute Error:83258.48511524936
```

In [23]:

```
EPSILON = 1e-10
def _error(actual: np.ndarray, predicted: np.ndarray):
    return actual - predicted
def _percentage_error(actual: np.ndarray, predicted: np.ndarray):
    return _error(actual, predicted) / (actual + EPSILON)

```

In [24]:

```
def rrse(actual: np.ndarray, predicted: np.ndarray):
    return np.sqrt(np.sum(np.square(actual - predicted)) / np.sum(np.square(actual - np

RRSE=rrse(y,x)
print(f'Root Relative Squared Error:{RRSE}')
```

Root Relative Squared Error:1.3013154221448553

In [25]:

```
def mape(actual: np.ndarray, predicted: np.ndarray):
    return np.mean(np.abs(_percentage_error(actual, predicted)))

MAPE=mape(y,x)
print(f'Mean Absolute Percentage Error:{MAPE}')
```

Mean Absolute Percentage Error:0.7330245186537612

In [ ]:

```

```