In [1]:
```
import warnings
import itertools
import numpy as np
import matplotlib.pyplot as plt
warnings.filterwarnings("ignore")
plt.style.use('fivethirtyeight')
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.read_excel("Sample - Superstore.xls")
df
```

Out[2]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Category	
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-BO-10001798	Furniture	E
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-CH-10000454	Furniture	
2	3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036	West	OFF-LA-10000240	Office Supplies	
3	4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	FUR-TA-10000577	Furniture	
4	5	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	OFF-ST-10000760	Office Supplies	

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Category	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9989	9990	CA-2014-110422	2014-01-21	2014-01-23	Second Class	TB-21400	Tom Boeckenhauer	Consumer	United States	Miami	...	33180	South	FUR-FU-10001889	Furniture	F
9990	9991	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627	West	FUR-FU-10000747	Furniture	F
9991	9992	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627	West	TEC-PH-10003645	Technology	
9992	9993	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627	West	OFF-PA-10004041	Office Supplies	
9993	9994	CA-2017-119914	2017-05-04	2017-05-09	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westminster	...	92683	West	OFF-AP-10002684	Office Supplies	A

9994 rows × 21 columns

```
In [3]:  furniture = df.loc[df['Category'] == 'Furniture']
         furniture['Order Date'].min(), furniture['Order Date'].max()
```

```
Out[3]: (Timestamp('2014-01-06 00:00:00'), Timestamp('2017-12-30 00:00:00'))
```

```
In [4]:  cols = ['Row ID', 'Order ID', 'Ship Date', 'Ship Mode', 'Customer ID', 'Customer Name', 'Segment', 'Country', 'City',
         furniture.drop(cols, axis=1, inplace=True)
         furniture = furniture.sort_values('Order Date')
```

```
furniture.isnull().sum()
```

Out[4]:

```
Order Date    0
Sales         0
dtype: int64
```

In [5]:

```
furniture = furniture.groupby('Order Date')['Sales'].sum().reset_index()
furniture
```

Out[5]:

	Order Date	Sales
0	2014-01-06	2573.8200
1	2014-01-07	76.7280
2	2014-01-10	51.9400
3	2014-01-11	9.9400
4	2014-01-13	879.9390
...	...	...
884	2017-12-24	1393.4940
885	2017-12-25	832.4540
886	2017-12-28	551.2568
887	2017-12-29	2330.7180
888	2017-12-30	323.1360

889 rows × 2 columns

In [6]:

```
furniture = furniture.set_index('Order Date')
furniture.index
```

Out[6]:

```
DatetimeIndex(['2014-01-06', '2014-01-07', '2014-01-10', '2014-01-11',
               '2014-01-13', '2014-01-14', '2014-01-16', '2014-01-19',
               '2014-01-20', '2014-01-21',
               ...
               '2017-12-18', '2017-12-19', '2017-12-21', '2017-12-22',
               '2017-12-23', '2017-12-24', '2017-12-25', '2017-12-28',
               '2017-12-29', '2017-12-30'],
              dtype='datetime64[ns]', name='Order Date', length=889, freq=None)
```

In [8]:

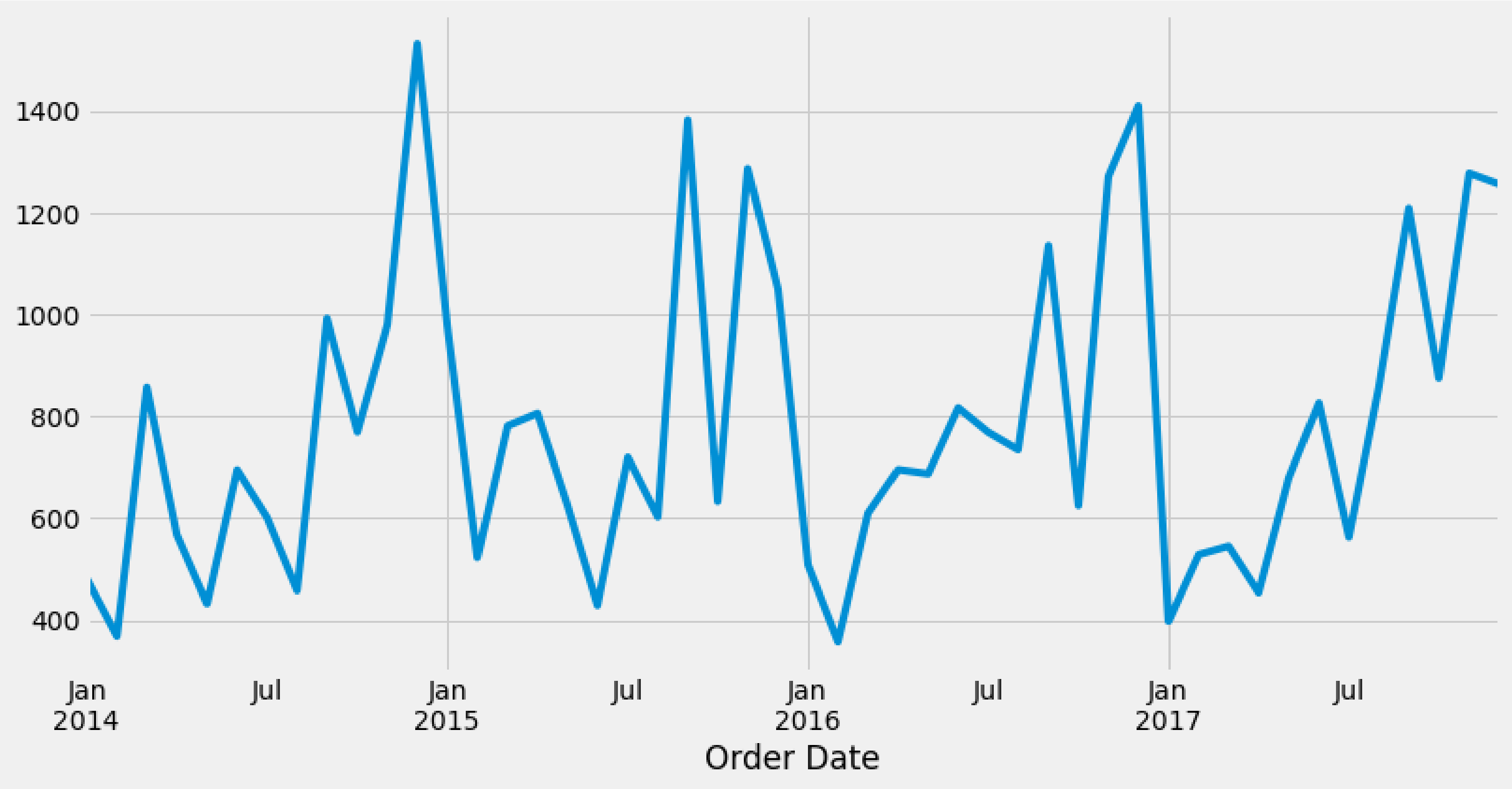
```
y_useful = furniture['Sales'].resample('MS').mean()
y_useful['2015':]
```

Out[8]:

```
Order Date
2015-01-01     978.328467
2015-02-01     522.395667
2015-03-01     781.236437
2015-04-01     805.822962
2015-05-01     624.996700
2015-06-01     428.565500
2015-07-01     719.706316
2015-08-01     602.412012
2015-09-01    1382.790684
2015-10-01     632.980184
2015-11-01    1286.701354
2015-12-01    1049.355418
2016-01-01     508.182867
2016-02-01     356.868273
2016-03-01     609.575810
2016-04-01     695.373158
2016-05-01     687.265227
2016-06-01     816.910750
2016-07-01     768.736412
2016-08-01     734.307782
2016-09-01    1135.953371
2016-10-01     624.872474
2016-11-01    1271.345152
2016-12-01    1410.719808
2017-01-01     397.602133
2017-02-01     528.179800
2017-03-01     544.672240
2017-04-01     453.297905
2017-05-01     678.302328
2017-06-01     826.460291
2017-07-01     562.524857
2017-08-01     857.881889
2017-09-01    1209.508583
2017-10-01     875.362728
2017-11-01    1277.817759
2017-12-01    1256.298672
Freq: MS, Name: Sales, dtype: float64
```

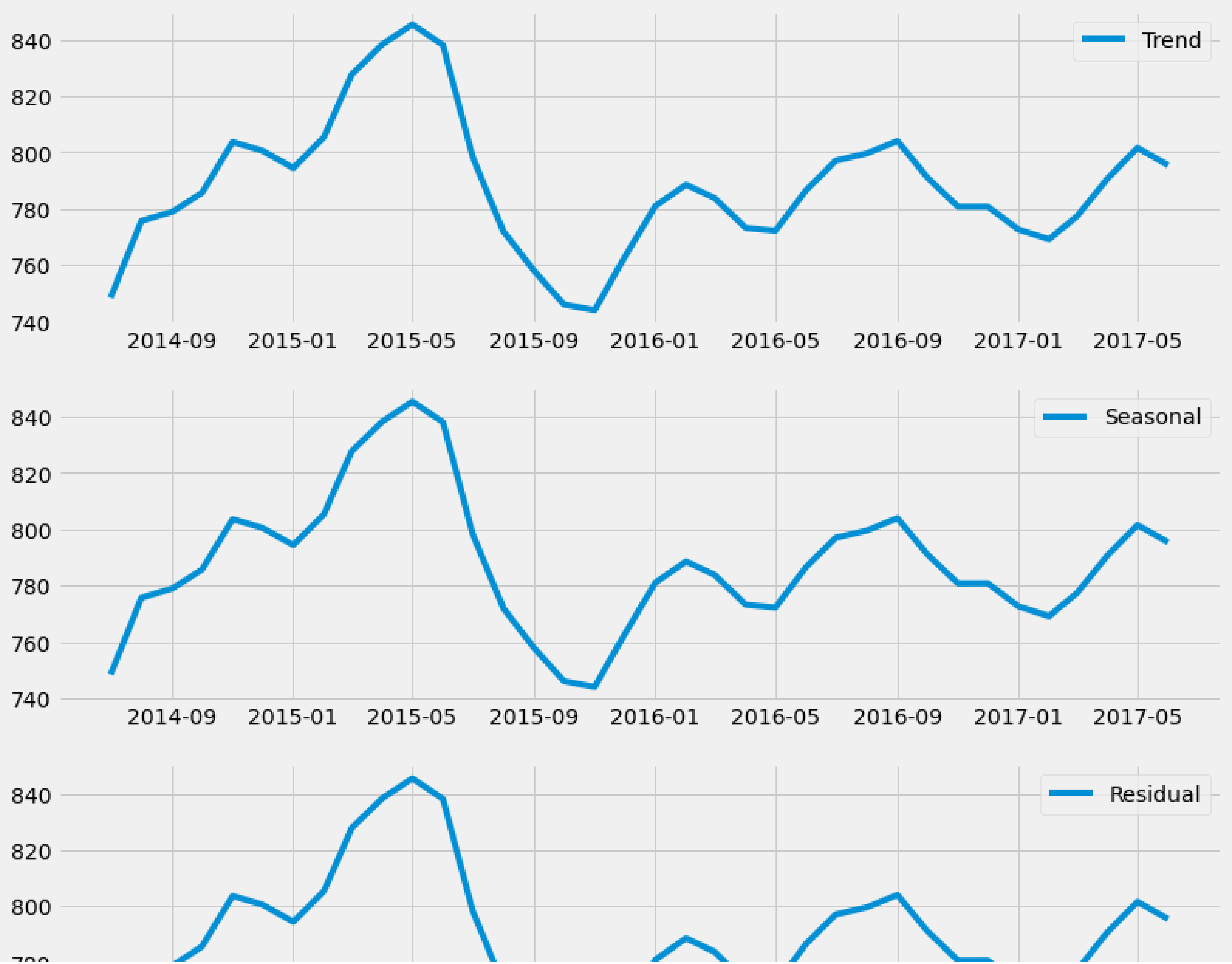
In [28]:

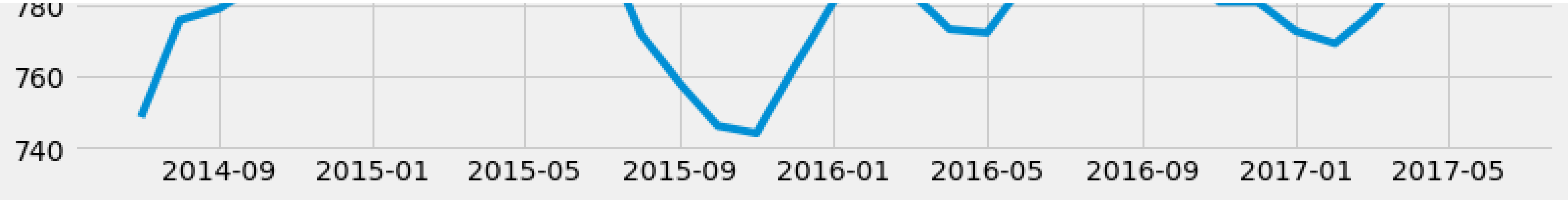
```
y_useful.plot(figsize=(12, 6))
plt.show()
```



In [10]:

```
decomposition = sm.tsa.seasonal_decompose(y_useful, model='additive')
trend = decomposition.trend
plt.figure(figsize=(12,16))
plt.subplot(411)
plt.plot(trend,label="Trend")
plt.legend()
Seasonal = decomposition.trend
plt.subplot(412)
plt.plot(trend,label="Seasonal")
plt.legend()
Residual = decomposition.trend
plt.subplot(413)
plt.plot(trend,label="Residual")
plt.legend()
plt.show()
```





In [11]:

```
p = d = q = range(0, 2)
pdq = list(itertools.product(p, d, q))
```

In [12]:

```
pdq
```

Out[12]:

```
[(0, 0, 0),
 (0, 0, 1),
 (0, 1, 0),
 (0, 1, 1),
 (1, 0, 0),
 (1, 0, 1),
 (1, 1, 0),
 (1, 1, 1)]
```

In [13]:

```
seasonal_pdq = [(x[0], x[1], x[2], 12) for x in pdq]
```

In [14]:

```
seasonal_pdq
```

Out[14]:

```
[(0, 0, 0, 12),
 (0, 0, 1, 12),
 (0, 1, 0, 12),
 (0, 1, 1, 12),
 (1, 0, 0, 12),
 (1, 0, 1, 12),
 (1, 1, 0, 12),
 (1, 1, 1, 12)]
```

In [15]:

```
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            model = sm.tsa.statespace.SARIMAX(y_useful, order=param, seasonal_order=param_seasonal, enforce_stationar
                                            enforce_invertibility=False)
            results = model.fit()
            print('ARIMA{}x{} - AIC:{}'.format(param, param_seasonal, results.aic))
        except:
            continue
```

```
ARIMA(0, 0, 0)x(0, 0, 0, 12) - AIC:769.0817523205915
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 0, 0)x(0, 0, 1, 12) - AIC:1354.0669954233208
ARIMA(0, 0, 0)x(0, 1, 0, 12) - AIC:477.7170130920899
ARIMA(0, 0, 0)x(0, 1, 1, 12) - AIC:302.27028997938197
ARIMA(0, 0, 0)x(1, 0, 0, 12) - AIC:497.2314433418337
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 0, 0)x(1, 0, 1, 12) - AIC:1144.2225605403871
ARIMA(0, 0, 0)x(1, 1, 0, 12) - AIC:318.0047199116341
ARIMA(0, 0, 0)x(1, 1, 1, 12) - AIC:304.2488280301906
ARIMA(0, 0, 1)x(0, 0, 0, 12) - AIC:720.9252270758116
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 0, 1)x(0, 0, 1, 12) - AIC:2695.913697357427
ARIMA(0, 0, 1)x(0, 1, 0, 12) - AIC:466.5607429809158
ARIMA(0, 0, 1)x(0, 1, 1, 12) - AIC:291.62613896732864
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 0, 1)x(1, 0, 0, 12) - AIC:499.5869033885854
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 0, 1)x(1, 0, 1, 12) - AIC:2347.5641565412016
ARIMA(0, 0, 1)x(1, 1, 0, 12) - AIC:319.98848769468657
ARIMA(0, 0, 1)x(1, 1, 1, 12) - AIC:291.8725576524215
ARIMA(0, 1, 0)x(0, 0, 0, 12) - AIC:677.894766843944
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 1, 0)x(0, 0, 1, 12) - AIC:1320.66170690448
ARIMA(0, 1, 0)x(0, 1, 0, 12) - AIC:486.63785672282035
ARIMA(0, 1, 0)x(0, 1, 1, 12) - AIC:304.9671228167956
ARIMA(0, 1, 0)x(1, 0, 0, 12) - AIC:497.78896630044073
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
```

```
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 1, 0)x(1, 0, 1, 12) - AIC:1366.7797226279727
ARIMA(0, 1, 0)x(1, 1, 0, 12) - AIC:319.7714068109211
ARIMA(0, 1, 0)x(1, 1, 1, 12) - AIC:306.9113200151535
ARIMA(0, 1, 1)x(0, 0, 0, 12) - AIC:649.9056176817318
ARIMA(0, 1, 1)x(0, 0, 1, 12) - AIC:2508.436923977664
ARIMA(0, 1, 1)x(0, 1, 0, 12) - AIC:458.87055484828795
ARIMA(0, 1, 1)x(0, 1, 1, 12) - AIC:279.58062316811436
ARIMA(0, 1, 1)x(1, 0, 0, 12) - AIC:486.18329774426684
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(0, 1, 1)x(1, 0, 1, 12) - AIC:803.6652444374075
ARIMA(0, 1, 1)x(1, 1, 0, 12) - AIC:310.75743684175046
ARIMA(0, 1, 1)x(1, 1, 1, 12) - AIC:281.5576621461235
ARIMA(1, 0, 0)x(0, 0, 0, 12) - AIC:692.1645522067713
ARIMA(1, 0, 0)x(0, 0, 1, 12) - AIC:1350.6453425567156
ARIMA(1, 0, 0)x(0, 1, 0, 12) - AIC:479.46321478521355
ARIMA(1, 0, 0)x(0, 1, 1, 12) - AIC:304.2077675160951
ARIMA(1, 0, 0)x(1, 0, 0, 12) - AIC:480.92593679351927
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(1, 0, 0)x(1, 0, 1, 12) - AIC:1290.889297086618
ARIMA(1, 0, 0)x(1, 1, 0, 12) - AIC:304.46646750846253
ARIMA(1, 0, 0)x(1, 1, 1, 12) - AIC:304.5842692143838
ARIMA(1, 0, 1)x(0, 0, 0, 12) - AIC:665.7794442186472
ARIMA(1, 0, 1)x(0, 0, 1, 12) - AIC:2392.9009436960123
ARIMA(1, 0, 1)x(0, 1, 0, 12) - AIC:468.3685195815001
ARIMA(1, 0, 1)x(0, 1, 1, 12) - AIC:293.3422193965914
ARIMA(1, 0, 1)x(1, 0, 0, 12) - AIC:482.5763323877232
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(1, 0, 1)x(1, 0, 1, 12) - AIC:2436.877259630648
ARIMA(1, 0, 1)x(1, 1, 0, 12) - AIC:306.0156002132424
ARIMA(1, 0, 1)x(1, 1, 1, 12) - AIC:293.7513188135041
ARIMA(1, 1, 0)x(0, 0, 0, 12) - AIC:671.2513547541902
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
ARIMA(1, 1, 0)x(0, 0, 1, 12) - AIC:1180.8208704956457
ARIMA(1, 1, 0)x(0, 1, 0, 12) - AIC:479.2003422281134
ARIMA(1, 1, 0)x(0, 1, 1, 12) - AIC:300.21306116191005
ARIMA(1, 1, 0)x(1, 0, 0, 12) - AIC:475.34036587851494
```

```
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

```
ARIMA(1, 1, 0)x(1, 0, 1, 12) - AIC:1073.2557519174643
ARIMA(1, 1, 0)x(1, 1, 0, 12) - AIC:300.6270901345431
ARIMA(1, 1, 0)x(1, 1, 1, 12) - AIC:302.32649925046746
ARIMA(1, 1, 1)x(0, 0, 0, 12) - AIC:649.0318019835194
```

```
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

```
ARIMA(1, 1, 1)x(0, 0, 1, 12) - AIC:1370.2232859458645
ARIMA(1, 1, 1)x(0, 1, 0, 12) - AIC:460.4762687610177
ARIMA(1, 1, 1)x(0, 1, 1, 12) - AIC:281.3873006939415
ARIMA(1, 1, 1)x(1, 0, 0, 12) - AIC:469.5250354660838
```

```
C:\Users\U.R Computer\anaconda\lib\site-packages\statsmodels\base\model.py:566: ConvergenceWarning: Maximum Likelihoo
d optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

```
ARIMA(1, 1, 1)x(1, 0, 1, 12) - AIC:1344.609794136266
ARIMA(1, 1, 1)x(1, 1, 0, 12) - AIC:297.7875439530794
ARIMA(1, 1, 1)x(1, 1, 1, 12) - AIC:283.36610143638575
```

In [ ]:

In [ ]:

In [26]:

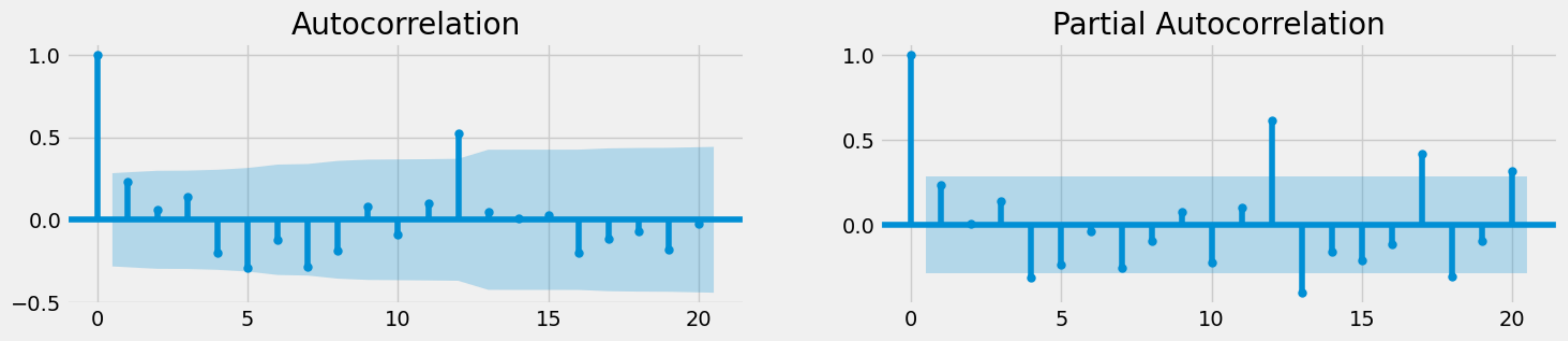
```
df_output = pd.Series(y_test[0:4],index=['Test Statistic','p-value','Lags Used','No. of Obs'])
for i,j in y_test[4].items():
    df_output["Criticality is (%s)"%i] = j
print(df_output)
```

```
Test Statistic          -5.191070
p-value                  0.000009
Lags Used               10.000000
No. of Obs              37.000000
Criticality is (1%)     -3.620918
Criticality is (5%)     -2.943539
Criticality is (10%)    -2.610400
dtype: float64
```

In [17]:

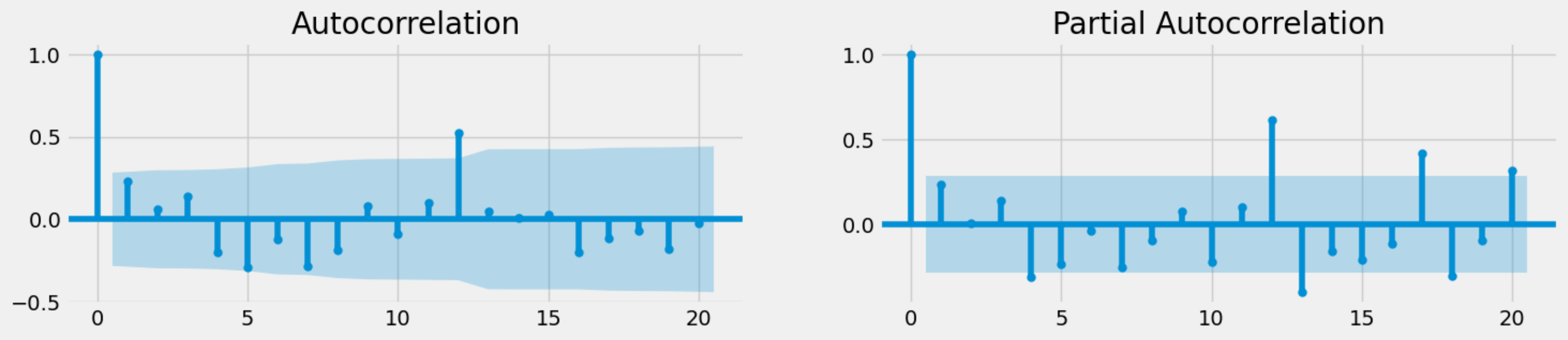
```
from statsmodels.tsa.stattools import acf, pacf
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
# fig, (ax1,ax2) = plt.subplots(1,2,figsize=(10,5))
# fig = sm.graphics.tsa.plot_acf(y_useful.values.squeeze(),lags=20,ax=ax1)
# ax1.axhline(y==-1.96/np.sqrt(len(y_useful)))
# ax1.axhline(y==1.96/np.sqrt(len(y_useful)))
# fig = sm.graphics.tsa.plot_pacf(y_useful,lags=20,ax=ax2)
# ax2.axhline(y==-1.96/np.sqrt(len(y_useful)))
# ax2.axhline(y==1.96/np.sqrt(len(y_useful)))
fig, axes = plt.subplots(1,2,figsize=(16,3), dpi= 100)
plot_acf(y_useful, lags=20, ax=axes[0])
plot_pacf(y_useful, lags=20, ax=axes[1])
```

Out[17]:



Autocorrelation

Partial Autocorrelation



Autocorrelation

Partial Autocorrelation

In [18]:

```
model = sm.tsa.ARIMA(y_useful,order=(2,0,2))
result = model.fit(disp=-1)
result
```

Out[18]:
```
<statsmodels.tsa.arima_model.ARMAResultsWrapper at 0x21efbbf31f0>
```

In [19]:
```
print(result.aic)
print(result.bic)
```

```
693.5725330243508
704.7997390897982
```

In [20]:
```
from sklearn import metrics
results_pred = result.predict()
metrics.mean_absolute_error(y_useful.values,results_pred.values)
```

Out[20]:
```
231.642903843485
```

In [52]:
```
y_useful-results_pred
```

Out[52]:
```
Order Date
2014-01-01   -305.286384
2014-02-01   -340.042123
2014-03-01    172.161118
2014-04-01   -236.917428
2014-05-01   -272.673920
2014-06-01    -18.867273
2014-07-01   -153.135170
2014-08-01   -271.649494
2014-09-01    288.759970
2014-10-01    -73.930893
2014-11-01    226.117066
2014-12-01    673.371605
2015-01-01     17.773495
2015-02-01   -301.594138
2015-03-01     35.468012
2015-04-01     11.963666
2015-05-01   -154.831394
2015-06-01   -318.918648
2015-07-01     24.100538
2015-08-01   -173.287443
2015-09-01    663.364133
2015-10-01   -323.984606
2015-11-01    587.776743
2015-12-01     73.201379
2016-01-01   -285.964330
2016-02-01   -402.065558
```

```
2016-03-01    -60.386323
2016-04-01    -59.109570
2016-05-01    -54.138546
2016-06-01     52.260155
2016-07-01    -19.138099
2016-08-01    -43.196669
2016-09-01    361.537749
2016-10-01   -253.599733
2016-11-01    547.332326
2016-12-01    463.489019
2017-01-01   -506.971833
2017-02-01   -173.358646
2017-03-01   -207.882565
2017-04-01   -251.289195
2017-05-01    -29.887843
2017-06-01     73.582003
2017-07-01   -225.027485
2017-08-01    136.902627
2017-09-01    391.926466
2017-10-01     -1.865678
2017-11-01    474.881569
2017-12-01    323.888756
Freq: MS, dtype: float64
```

In [53]:

```
metrics.r2_score(y_useful,results_pred)
```

Out[53]:

```
0.06783298355618761
```

In [21]:

```
result.summary()
```

Out[21]:

ARMA Model Results

Dep. Variable:	Sales	**No. Observations:**	48
Model:	ARMA(2, 2)	**Log Likelihood**	-340.786
Method:	css-mle	**S.D. of innovations**	292.913
Date:	Wed, 14 Jul 2021	**AIC**	693.573
Time:	13:31:40	**BIC**	704.800
Sample:	01-01-2014	**HQIC**	697.815
	- 12-01-2017		

	coef	std err	z	P>|z|	[0.025	0.975]
const	785.4806	57.423	13.679	0.000	672.933	898.029
ar.L1.Sales	-0.0880	0.633	-0.139	0.889	-1.329	1.153
ar.L2.Sales	0.2451	0.311	0.787	0.431	-0.365	0.856
ma.L1.Sales	0.3534	0.618	0.572	0.568	-0.859	1.565
ma.L2.Sales	-0.1988	0.309	-0.643	0.520	-0.805	0.408

Roots

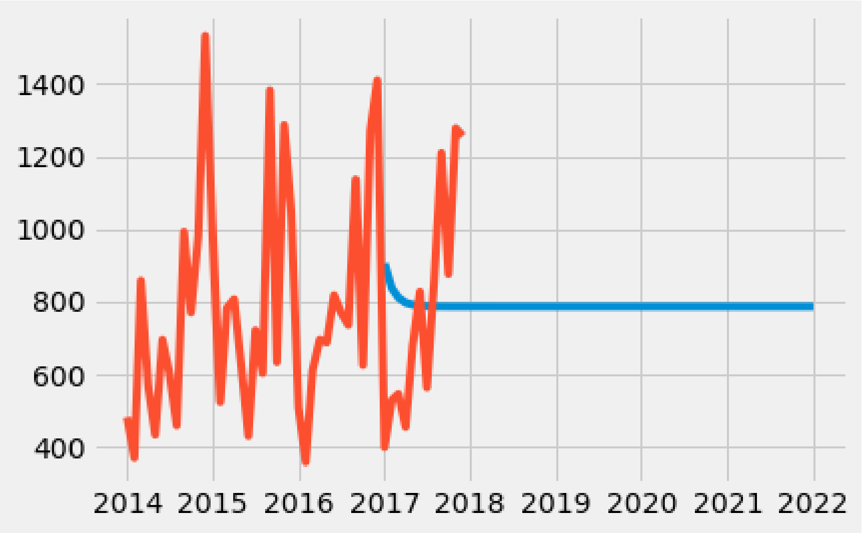
	Real	Imaginary	Modulus	Frequency
AR.1	-1.8483	+0.0000j	1.8483	0.5000
AR.2	2.2072	+0.0000j	2.2072	0.0000
MA.1	-1.5236	+0.0000j	1.5236	0.5000
MA.2	3.3015	+0.0000j	3.3015	0.0000

In [22]:

```
forc_arima = result.predict(start=36,end=96,dynamic=True)
plt.plot(forc_arima)
plt.plot(y_useful)
```

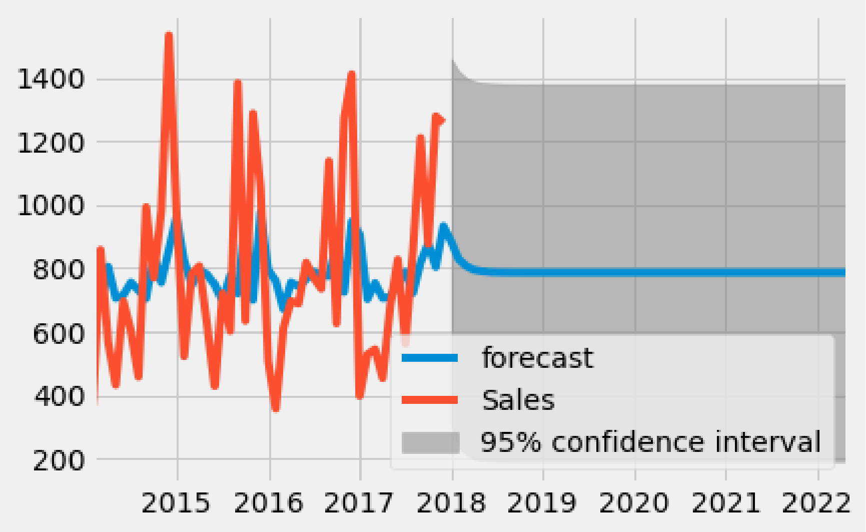
Out[22]:

```
[<matplotlib.lines.Line2D at 0x21efd277280>]
```



In [47]:

```
result.plot_predict(1,100)
x=result.forecast(steps=120)
```



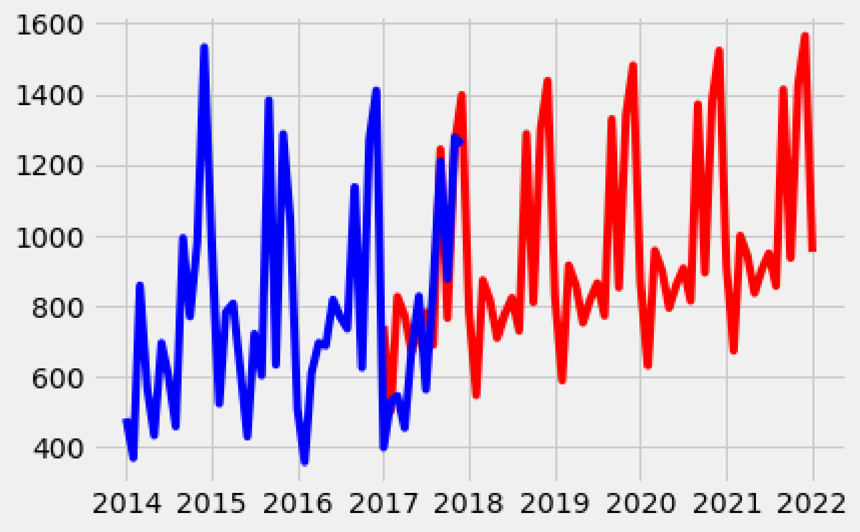
In [56]:

```
import statsmodels.api as sm
model=sm.tsa.statespace.SARIMAX(y_useful,order=(1, 1, 1),seasonal_order=(1,1,1,12))
results = model.fit()
forc_sarima = results.predict(start=36,end=96,dynamic=True)
```

```
plt.plot(forc_sarima,color="red")
plt.plot(y_useful,color="blue")
```

Out[56]:

```
[<matplotlib.lines.Line2D at 0x21e80291190>]
```



In [24]:

```
results.summary()
```

Out[24]:

SARIMAX Results

Dep. Variable:	Sales	**No. Observations:**	48
Model:	SARIMAX(1, 1, 1)x(1, 1, 1, 12)	**Log Likelihood**	-238.291
Date:	Wed, 14 Jul 2021	**AIC**	486.582
Time:	13:31:46	**BIC**	494.359
Sample:	01-01-2014	**HQIC**	489.267
	- 12-01-2017		
Covariance Type:	opg		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
ar.L1	0.0854	0.250	0.342	0.732	-0.404	0.575
ma.L1	-0.9984	13.948	-0.072	0.943	-28.337	26.340

ar.S.L12	0.0273	0.588	0.046	0.963	-1.124	1.179
ma.S.L12	-0.9921	63.664	-0.016	0.988	-125.772	123.788
sigma2	2.741e+04	1.84e+06	0.015	0.988	-3.58e+06	3.64e+06

Ljung-Box (L1) (Q):	0.05	**Jarque-Bera (JB):**	2.10
Prob(Q):	0.82	**Prob(JB):**	0.35
Heteroskedasticity (H):	0.54	**Skew:**	-0.37
Prob(H) (two-sided):	0.30	**Kurtosis:**	2.05

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

```
In [59]: metrics.mean_absolute_error(y_useful.values,results.predict().values)
```

```
Out[59]: 222.13466014555183
```

```
In [ ]:
```