In [8]:

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

import warnings
warnings.filterwarnings('ignore')
```

In [9]:

```
cc_data = pd.read_excel('CocaCola_Sales_Rawdata.xlsx')
cc_data
```

Out[9]:

	Quarter	Sales
0	Q1_86	1734.827000
1	Q2_86	2244.960999
2	Q3_86	2533.804993
3	Q4_86	2154.962997
4	Q1_87	1547.818996
5	Q2_87	2104.411995
6	Q3_87	2014.362999
7	Q4_87	1991.746998
8	Q1_88	1869.049999
9	Q2_88	2313.631996
10	Q3_88	2128.320000

In [10]:

```
quarter=['Q1','Q2','Q3','Q4']
n=cc_data['Quarter'][0]
n[0:2]
```

Out[10]: 'Q1'

In [11]:
```
cc_data['quarter']=0
```

In [12]:
```
for i in range(42):
    n=cc_data['Quarter'][i]
    cc_data['quarter'][i]=n[0:2]
```

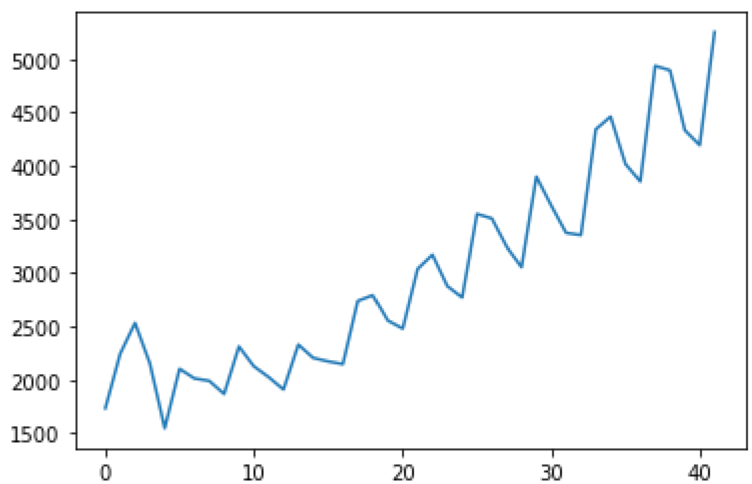
In [14]:
```
dummy=pd.DataFrame(pd.get_dummies(cc_data['quarter']))
```

In [15]:
```
cc_data_1=pd.concat((cc_data,dummy),axis=1)
t= np.arange(1,43)
cc_data_1['t']=t
cc_data_1['t_square']=cc_data_1['t']*cc_data_1['t']
```

In [16]:
```
log_Sales=np.log(cc_data_1['Sales'])
cc_data_1['log_Sales']=log_Sales
```

In [17]:
```
train= cc_data_1.head(38)
test=cc_data_1.tail(4)
cc_data_1.Sales.plot()
```

Out[17]: <AxesSubplot:>



In [18]:
```
import statsmodels.formula.api as smf
```

In [19]:
```
#linear model
linear= smf.ols('Sales~t',data=train).fit()
predlin=pd.Series(linear.predict(pd.DataFrame(test['t'])))
rmselin=np.sqrt((np.mean(np.array(test['Sales'])-np.array(predlin))**2))
rmselin
```

Out[19]: 421.17878760022813

In [20]:
```
#quadratic model
quad=smf.ols('Sales~t+t_square',data=train).fit()
predquad=pd.Series(quad.predict(pd.DataFrame(test[['t','t_square']])))
rmsequad=np.sqrt(np.mean((np.array(test['Sales'])-np.array(predquad))**2))
rmsequad
```

Out[20]: 475.56183518315095

In [21]:
```
#exponential model
expo=smf.ols('log_Sales~t',data=train).fit()
predexp=pd.Series(expo.predict(pd.DataFrame(test['t'])))
predexp
rmseexpo=np.sqrt(np.mean((np.array(test['Sales'])-np.array(np.exp(predexp)))**2))
rmseexpo
```

Out[21]: 466.24797310672346

In [22]:
```
#additive seasonality
additive= smf.ols('Sales~ Q1+Q2+Q3+Q4',data=train).fit()
predadd=pd.Series(additive.predict(pd.DataFrame(test[['Q1','Q2','Q3','Q4']])))
predadd
rmseadd=np.sqrt(np.mean((np.array(test['Sales'])-np.array(predadd))**2))
rmseadd
```

Out[22]: 1860.0238154547283

In [23]:
```
#additive seasonality with linear trend
addlinear= smf.ols('Sales~t+Q1+Q2+Q3+Q4',data=train).fit()
predaddlinear=pd.Series(addlinear.predict(pd.DataFrame(test[['t','Q1','Q2','Q3','Q4']])))
predaddlinear
```

Out[23]:
```
38    4292.265126
39    4066.761792
40    3961.769195
41    4639.214094
dtype: float64
```

In [24]:
```
rmseaddlinear=np.sqrt(np.mean((np.array(test['Sales'])-np.array(predaddlinear))**2))
rmseaddlinear
```

Out[24]:
```
464.98290239822427
```

In [25]:
```
#additive seasonality with quadratic trend
addquad=smf.ols('Sales~t+t_square+Q1+Q2+Q3+Q4',data=train).fit()
predaddquad=pd.Series(addquad.predict(pd.DataFrame(test[['t','t_square','Q1','Q2','Q3','Q4']])))
rmseaddquad=np.sqrt(np.mean((np.array(test['Sales'])-np.array(predaddquad))**2))
rmseaddquad
```

Out[25]:
```
301.73800719352977
```

In [26]:
```
#multiplicative seasonality
mulsea=smf.ols('log_Sales~Q1+Q2+Q3+Q4',data=train).fit()
predmul= pd.Series(mulsea.predict(pd.DataFrame(test[['Q1','Q2','Q3','Q4']])))
rmsemul= np.sqrt(np.mean((np.array(test['Sales'])-np.array(np.exp(predmul)))**2))
rmsemul
```

Out[26]:
```
1963.3896400779709
```

In [27]:

```
#multiplicative seasonality with linear trend
mullin= smf.ols('log_Sales~t+Q1+Q2+Q3+Q4',data=train).fit()
predmullin= pd.Series(mullin.predict(pd.DataFrame(test[['t','Q1','Q2','Q3','Q4']])))
rmsemulin=np.sqrt(np.mean((np.array(test['Sales'])-np.array(np.exp(predmullin)))**2))
rmsemulin
```

Out[27]: 225.5243904982721

In [28]:

```
#multiplicative seasonality with quadratic trend
mul_quad= smf.ols('log_Sales~t+t_square+Q1+Q2+Q3+Q4',data=train).fit()
pred_mul_quad= pd.Series(mul_quad.predict(test[['t','t_square','Q1','Q2','Q3','Q4']]))
rmse_mul_quad=np.sqrt(np.mean((np.array(test['Sales'])-np.array(np.exp(pred_mul_quad)))**2))
rmse_mul_quad
```

Out[28]: 581.8457187971785

In [29]:

```
#tabulating the rmse values

data={'Model':pd.Series(['rmse_mul_quad','rmseadd','rmseaddlinear','rmseaddquad','rmseexpo','rmselin','rmse
data
```

Out[29]:

```
{'Model': 0    rmse_mul_quad
 1          rmseadd
 2    rmseaddlinear
 3      rmseaddquad
 4         rmseexpo
 5          rmselin
 6          rmsemul
 7        rmsemulin
 8         rmsequad
 dtype: object,
 'Values': 0     581.845719
 1    1860.023815
 2     464.982902
 3     301.738007
 4     466.247973
 5     421.178788
 6    1963.389640
 7     225.524390
 8     475.561835
 dtype: float64}
```

In [30]:
```
Rmse=pd.DataFrame(data)
Rmse
```

Out[30]:

	Model	Values
0	rmse_mul_quad	581.845719
1	rmseadd	1860.023815
2	rmseaddlinear	464.982902
3	rmseaddquad	301.738007
4	rmseexpo	466.247973
5	rmselin	421.178788
6	rmsemul	1963.389640
7	rmsemulin	225.524390
8	rmsequad	475.561835

In [ ]:
```

```