In [1]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
```

In [2]:
```
airline_data=pd.read_excel('Airlines+Data (1).xlsx')
airline_data
```

Out[2]:

	Month	Passengers
0	1995-01-01	112
1	1995-02-01	118
2	1995-03-01	132
3	1995-04-01	129
4	1995-05-01	121
...	...	...
91	2002-08-01	405
92	2002-09-01	355
93	2002-10-01	306
94	2002-11-01	271
95	2002-12-01	306

96 rows × 2 columns

In [3]:
```
airline_data.head()
```

Out[3]:

	Month	Passengers
0	1995-01-01	112
1	1995-02-01	118
2	1995-03-01	132
3	1995-04-01	129
4	1995-05-01	121

In [4]:
```
airline_data.shape
```

Out[4]:
```
(96, 2)
```

In [5]:
```
airline_data.isna().sum()
```

Out[5]:
```
Month         0
Passengers    0
dtype: int64
```

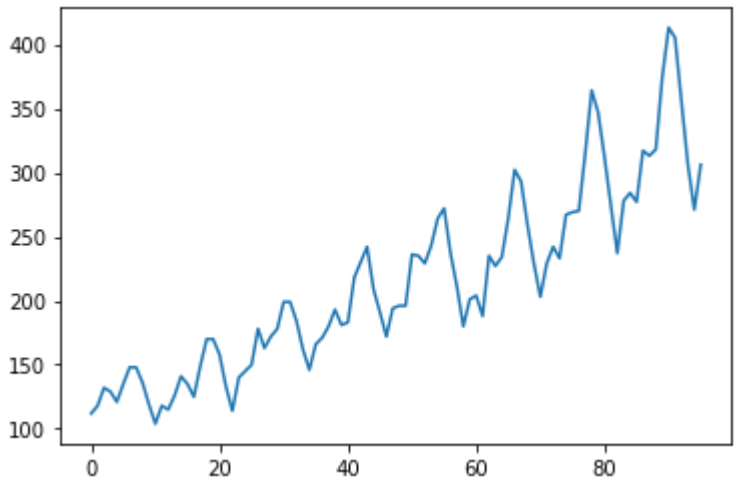
In [6]:
```
airline_data.dtypes
```

Out[6]:
```
Month         datetime64[ns]
Passengers             int64
dtype: object
```

In [7]:
```
month=['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
month=pd.DataFrame(month)
months=pd.DataFrame(np.tile(month,(8,1)))
airline_data=pd.concat([airline_data,months],axis=1)
airline_data.columns=['Month','Passengers','months']
```

In [8]:
```
month_dummies=pd.get_dummies(airline_data ['months'])
airline_data =pd.concat([airline_data ,month_dummies],axis=1)
airline_data['t']=np.arange(1,97)
airline_data['t_sq']=airline_data ['t']*airline_data ['t']
airline_data['log_passengers']=np.log(airline_data ['Passengers'])
```

In [9]:
```
Train=airline_data [0:85]
Test=airline_data [85:]
plt.plot(airline_data.iloc[:,1])
Test.set_index(np.arange(1,12),inplace=True)
```



Using Linear

In [10]:
```
import statsmodels.formula.api as smf
lin_model=smf.ols('Passengers~t',data=Train).fit()
predict_lin=lin_model.predict(Test['t'])
error_lin=Test['Passengers']-predict_lin
rmse_lin=np.sqrt(np.mean(error_lin**2))
rmse_lin
```

Out[10]:
```
55.674170015416216
```

For Exponential

In [11]:
```
exp_model=smf.ols('log_passengers~t',data=Train).fit()
predict_exp=exp_model.predict(Test['t'])
error_exp=Test['Passengers']-predict_exp
rmse_exp=np.sqrt(np.mean(error_exp**2))
rmse_exp
```

Out[11]: 329.69175113922927

For Quadratic

In [12]:
```
import statsmodels.formula.api as smf
quad_model=smf.ols('Passengers~t+t_sq',data=Train).fit()
predict_quad=quad_model.predict(Test[['t','t_sq']])
error_quad=Test['Passengers']-predict_quad
rmse_quad=np.sqrt(np.mean(error_quad**2))
rmse_quad
```

Out[12]: 50.65954577650042

Additive Seasonality

In [15]:
```
import statsmodels.formula.api as smf
add_sea_model=smf.ols('Passengers~Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nov',da
predict_add_sea=add_sea_model.predict(Test[['Jan','Feb','Mar','Apr','May','Jun','
error_add_sea=Test['Passengers']-predict_add_sea
rmse_add_sea=np.sqrt(np.mean(error_add_sea**2))
rmse_add_sea
```

Out[15]: 134.34479910432762

For Additive Seasonality Quadratic

In [20]:
```
add_sea_quad_model=smf.ols('Passengers~Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nc
pred_add_sea_quad=add_sea_quad_model.predict(Test[['Jan','Feb','Mar','Apr','May',
error_add_sea_quad=Test['Passengers']-pred_add_sea_quad
rmse_add_sea_quad=np.sqrt(np.mean(error_add_sea_quad**2))
rmse_add_sea_quad
```

Out[20]: 27.41271496120789

For Multiplicative Seasonality

In [21]:
```
import statsmodels.formula.api as smf
mul_sea_model=smf.ols('log_passengers~Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nov
predict_mul_sea=mul_sea_model.predict(Test[['Jan','Feb','Mar','Apr','May','Jun','
error_mul_sea=Test['Passengers']-predict_mul_sea
rmse_mul_sea=np.sqrt(np.mean(error_mul_sea**2))
rmse_mul_sea
```

Out[21]: 330.1926780196679

For Multiplicative Additive Seasonality

In [24]:
```
import statsmodels.formula.api as smf
mul_add_sea_model=smf.ols('log_passengers~t+Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+O
predict_mul_add_sea=mul_add_sea_model.predict(Test[['t','Jan','Feb','Mar','Apr','
error_mul_add_sea=Test['Passengers']-predict_mul_add_sea
rmse_mul_add_sea=np.sqrt(np.mean(error_mul_add_sea**2))
rmse_mul_add_sea
```

Out[24]: 329.66032649959925

In [25]:
```
# Additive Seasonality Quadratic is having least rmse So use Additive Seasonality
```

In [ ]:
```

```