Lab 3 Maryna Borovyk Task 2. Grid search for simple exponential smoothing.

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

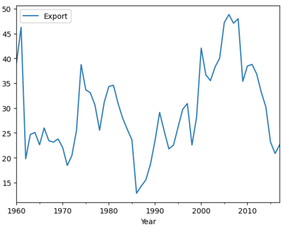
In [2]:
```
df = pd.read_csv('AlgeriaExport.txt', header = None)
df['Year']=pd.RangeIndex(start=1960, stop=1960+len(df))
df['Year'] = pd.to_datetime(df['Year'], format='%Y')
df.set_index('Year', inplace=True)
df.columns = ['Export']
df.index.freq='YS'
df.head()
```

Out[2]:

Year	Export
1960-01-01	39.043173
1961-01-01	46.244557
1962-01-01	19.793873
1963-01-01	24.684682
1964-01-01	25.084059

In [3]:
```
df.plot()
```

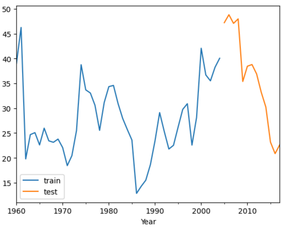
Out[3]:
```
<Axes: xlabel='Year'>
```



In [4]:
```
train_data = df.iloc[0:45]
test_data = df.iloc[45:]
```

In [5]:
```
train_data['Export'].plot(legend=True, label='train')
test_data['Export'].plot(legend=True, label='test')
```

Out[5]:
```
<Axes: xlabel='Year'>
```

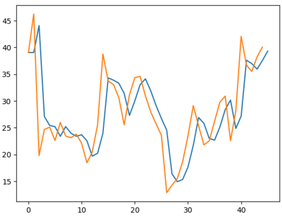


In [6]:
```
train_values=train_data['Export'].values
alpha=0.7
l0=train_values[0]
predicted_train_values=[l0]
```

In [7]:
```
for i in range(0, len(train_values-1)):
    predicted_train_values.append(alpha*train_values[i]+(1-alpha)*predicted_train_values[i])
```

In [8]:
```
plt.plot(predicted_train_values)
plt.plot(train_values)
```

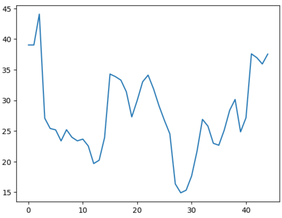
Out[8]:
```
[<matplotlib.lines.Line2D at 0x12beeaea0>]
```



In [9]:
```
def ses(train_values, alpha, l0):
    predicted_train_values = [l0]
    for i in range(0, len(train_values)-1):
        predicted_train_values.append(alpha * train_values[i] + (1 - alpha) * predicted_train_values[i])
    return predicted_train_values
```

In [10]:
```
pdt=ses(train_values, 0.7, train_values[0])
plt.plot(pdt)
```

Out[10]:
```
[<matplotlib.lines.Line2D at 0x12bf4ef90>]
```



In [11]:
```
def sse(values, predictions):
    sum=0
    for i in range(len(values)):
        sum+=(values[i]-predictions[i])**2
    return sum
```

In [12]:
```
sse(train_values, predicted_train_values)
```

Out[12]:
```
1663.92327922879
```

In [13]:
```
def ses_fit(train_values, dalpha):
    l0=train_values[0]
    alpha_grid=np.arange(0, 1, dalpha)
    sse_list=[]
    for alpha in alpha_grid:
        predicted_train_values=ses(train_values, alpha, l0)
        sse_list.append(sse(train_values, predicted_train_values))
    min_index=min(enumerate(sse_list), key=lambda x: x[1]) [0]
    best_alpha=alpha_grid[min_index]

    return best_alpha
```

In [14]:
```
alpha=ses_fit(train_values,0.01)
print(alpha)
```

```
0.76
```

In [15]:
```
def ses_predict(train_values, alpha, l0, h):
    predicted_train_values=ses(train_values, alpha, l0)
    one_step=alpha*train_values[-1]+(1-alpha)*predicted_train_values[-1]
    h_steps=[one_step]*h
    return one_step, h_steps, predicted_train_values
```

In [16]:
```
one_step, h_steps, predicted_train_values=ses_predict(train_values, alpha, train_values[0], len(test_data))
```

In [17]:
```
train_data['Fit']=predicted_train_values
test_data['Predicted']=h_steps
test_data
```

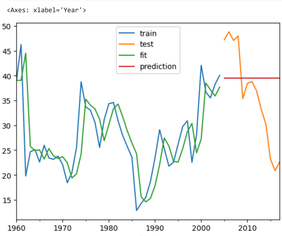
```
/var/folders/z6/r8xtfhp56gd9rzqzptq9vv880000gn/T/ipykernel_79542/1599865325.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  train_data['Fit']=predicted_train_values
/var/folders/z6/r8xtfhp56gd9rzqzptq9vv880000gn/T/ipykernel_79542/1599865325.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  test_data['Predicted']=h_steps
```

Out[17]:

Year	Export	Predicted
2005-01-01	47.205193	39.484538
2006-01-01	48.810688	39.484538
2007-01-01	47.068164	39.484538
2008-01-01	47.973345	39.484538
2009-01-01	35.371651	39.484538
2010-01-01	38.444548	39.484538
2011-01-01	38.786954	39.484538
2012-01-01	36.890548	39.484538
2013-01-01	33.209898	39.484538
2014-01-01	30.219117	39.484538
2015-01-01	23.171778	39.484538
2016-01-01	20.860011	39.484538
2017-01-01	22.638887	39.484538

In [18]:
```
train_data['Export'].plot(legend=True,label='train')
test_data['Export'].plot(legend=True,label='test')
train_data['Fit'].plot(legend=True,label='fit')
test_data['Predicted'].plot(legend=True,label='prediction')
```

Out[18]:
```
<Axes: xlabel='Year'>
```



In [18]: