**EX:No.7 221501034**

**15/04/25**

**IMPLEMENT PROGRAM FOR DECOMPOSING TIME SERIES DATA INTO TREND AND SEASONALITY.**

**AIM:**

To Implement programs for Implement program for decomposing time series data into trend and seasonality.

**OBJECTIVE:**

Smooth the electric production data to reduce noise, highlight trends, and prepare for forecasting.

**BACKGROUND:**

1.Time series data has short-term fluctuations.

2.Moving average reduces noise and clarifies trends.

3.Smoothed data improves forecast accuracy and interpretability.

**SCOPE OF THE PROGRAM:**

1.Load and clean dataset

2.Convert date column to datetime

3.Aggregate data monthly and yearly

4.Apply 3-month and 12-month moving averages

5.Plot original vs smoothed data

**ALGORITHM:**

1.Import libraries

2.Load dataset

3.Preprocess and set datetime index

4.Resample data (monthly, yearly)

5.Apply 3-month & 12-month smoothing

6.Visualize results

**.**

**PROCESS:**

# Install necessary libraries (uncomment if not already installed)

# !pip install pandas matplotlib statsmodels

import pandas as pd

import matplotlib.pyplot as plt

from statsmodels.tsa.seasonal import seasonal\_decompose

# Load the dataset

df = pd.read\_csv('/content/AAPL.csv', parse\_dates=['Date'], index\_col='Date')

# Display the first few rows of the dataset to check column names

print(df.head())

# Use the 'Close' price for decomposition

time\_series = df['Close']

# Optional: Resample to monthly frequency to smooth and match seasonal period

time\_series = time\_series.resample('M').mean()

# Decompose the time series into trend, seasonal, and residual components

# Multiplicative model assumes seasonality scales with the level; use 'additive' if you prefer linear variation

decomposition = seasonal\_decompose(time\_series, model='multiplicative', period=12)

# Plot the decomposition

plt.figure(figsize=(12, 8))

# Plot the observed data

plt.subplot(411)

plt.plot(time\_series, label='Observed')

plt.title('Observed Time Series (AAPL Close Price)')

plt.legend(loc='upper left')

# Plot the trend component

plt.subplot(412)

plt.plot(decomposition.trend, label='Trend', color='orange')

plt.title('Trend')

plt.legend(loc='upper left')

# Plot the seasonal component

plt.subplot(413)

plt.plot(decomposition.seasonal, label='Seasonality', color='green')

plt.title('Seasonality')

plt.legend(loc='upper left')

# Plot the residual component

plt.subplot(414)

plt.plot(decomposition.resid, label='Residual', color='red')

plt.title('Residual')

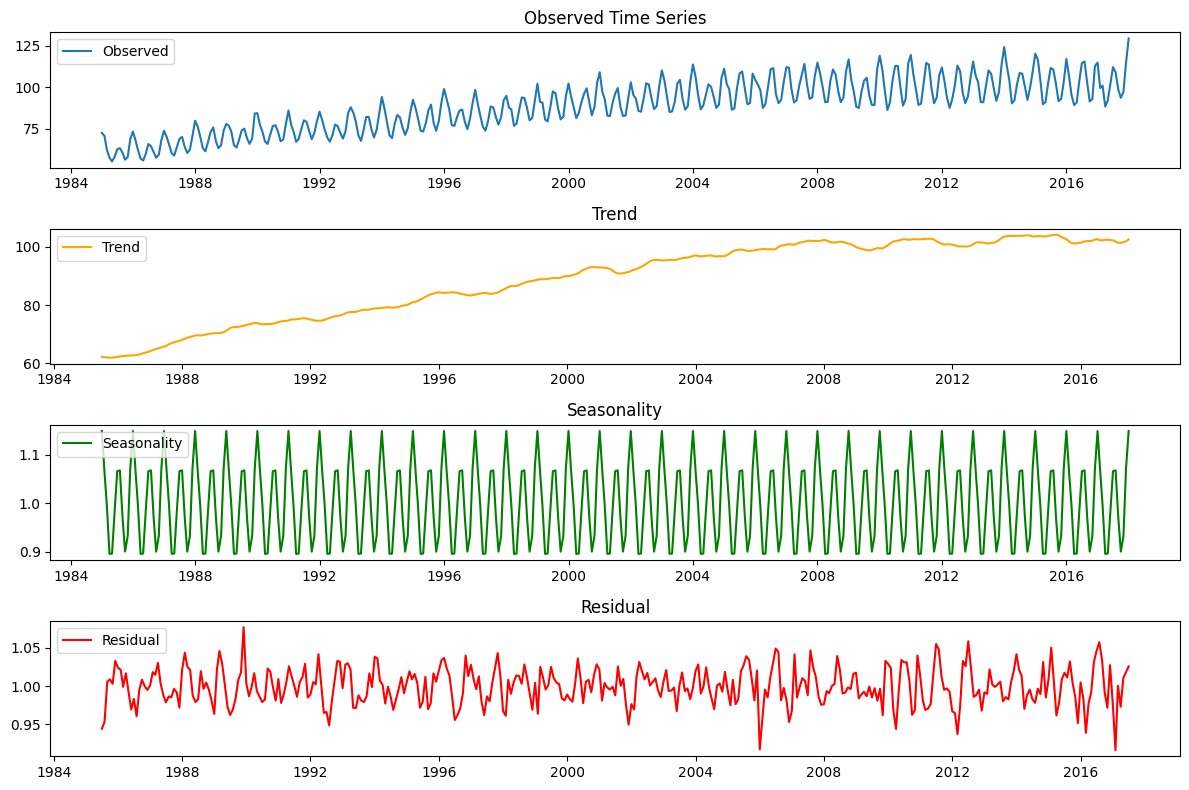
plt.legend(loc='upper left')

# Tighten the layout and show the plot

plt.tight\_layout()

plt.show()

**OUTPUT:**



**RESULT:**

The program to Implement program for decomposing time series data into trend and seasonality is implemented successfully.