**Implement programs for estimating & eliminating trend in time series data- aggregation, smoothing.**

**EX.No:4**

**DATE: 25/01/2**

**AIM:**

To implement programs for estimating & eliminating trend in time series data

**ALGORITHM:**

1. Load and preprocess the time series data (handle missing values, set datetime index).
2. Aggregate the data to a higher time frequency (e.g., daily to weekly) to reduce noise.
3. Apply moving average smoothing to highlight the underlying trend.
4. Apply exponential smoothing to weigh recent observations more heavily.
5. Decompose the series into trend, seasonality, and residual using additive/multiplicative models.
6. Eliminate trend by subtracting the trend component from the original series.

**CODE:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from statsmodels.tsa.seasonal import seasonal\_decompose

from statsmodels.tsa.holtwinters import SimpleExpSmoothing

np.random.seed(42)

date\_range = pd.date\_range(start='2020-01-01', periods=100, freq='D')

trend = np.linspace(10, 50, 100)

seasonality = 10 \* np.sin(np.linspace(0, 3 \* np.pi, 100))

noise = np.random.normal(scale=3, size=100)

data = trend + seasonality + noise

ts = pd.Series(data, index=date\_range)

ts\_weekly = ts.resample('W').mean()

rolling\_mean = ts.rolling(window=7).mean()

exp\_smoothing = SimpleExpSmoothing(ts).fit(smoothing\_level=0.2, optimized=False)

exp\_smoothed = exp\_smoothing.fittedvalues

decomposition = seasonal\_decompose(ts, model='additive', period=30)

trend\_component = decomposition.trend

detrended = ts - trend\_component

plt.figure(figsize=(14, 6))

plt.plot(ts, label='Original Data', alpha=0.6)

plt.plot(ts\_weekly, label='Weekly Aggregated', color='orange')

plt.plot(rolling\_mean, label='Moving Average (7-Day)', color='green')

plt.plot(exp\_smoothed, label='Exponential Smoothing', color='red', linestyle='--')

plt.plot(detrended, label='Detrended Series', color='purple')

plt.title('Time Series Analysis: Aggregation, Smoothing, and Detrending')

plt.xlabel('Date')

plt.ylabel('Value')

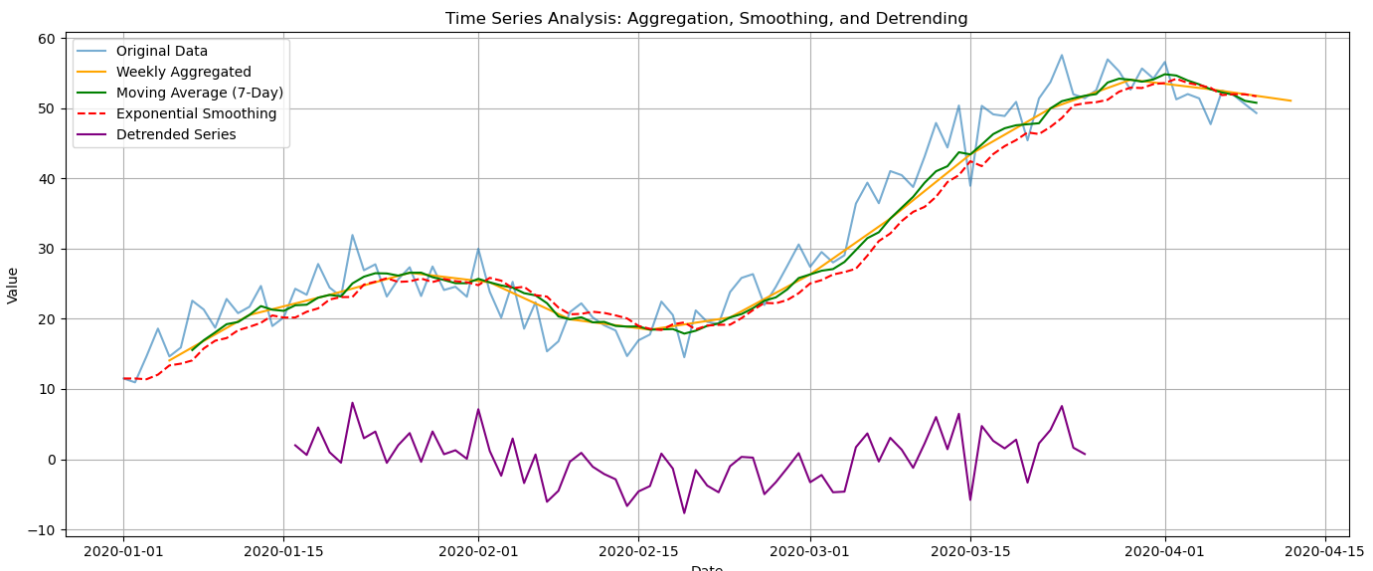
plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

**OUTPUT:**

****

**RESULT:**

Thus the program has been completed and verified successfully.