EX:No.4 DATE:1/02/25	**Implement programs for estimating & eliminating trend in time series data – aggregation, smoothing.**

AIM:

To Implement programs for estimating & eliminating trend in time series data – aggregation, smoothing..

OBJECTIVE:

To estimate and remove trends in time-series air pollution data using aggregation and smoothing techniques.

BACKGROUND:

- Time series data often has trends that affect analysis.
- **Aggregation** (e.g., monthly/yearly averaging) helps identify patterns.
- **Smoothing** (e.g., moving average, exponential smoothing) removes fluctuations.
- Trend elimination improves forecasting and stationarity.

SCOPE OF THE PROGRAM:

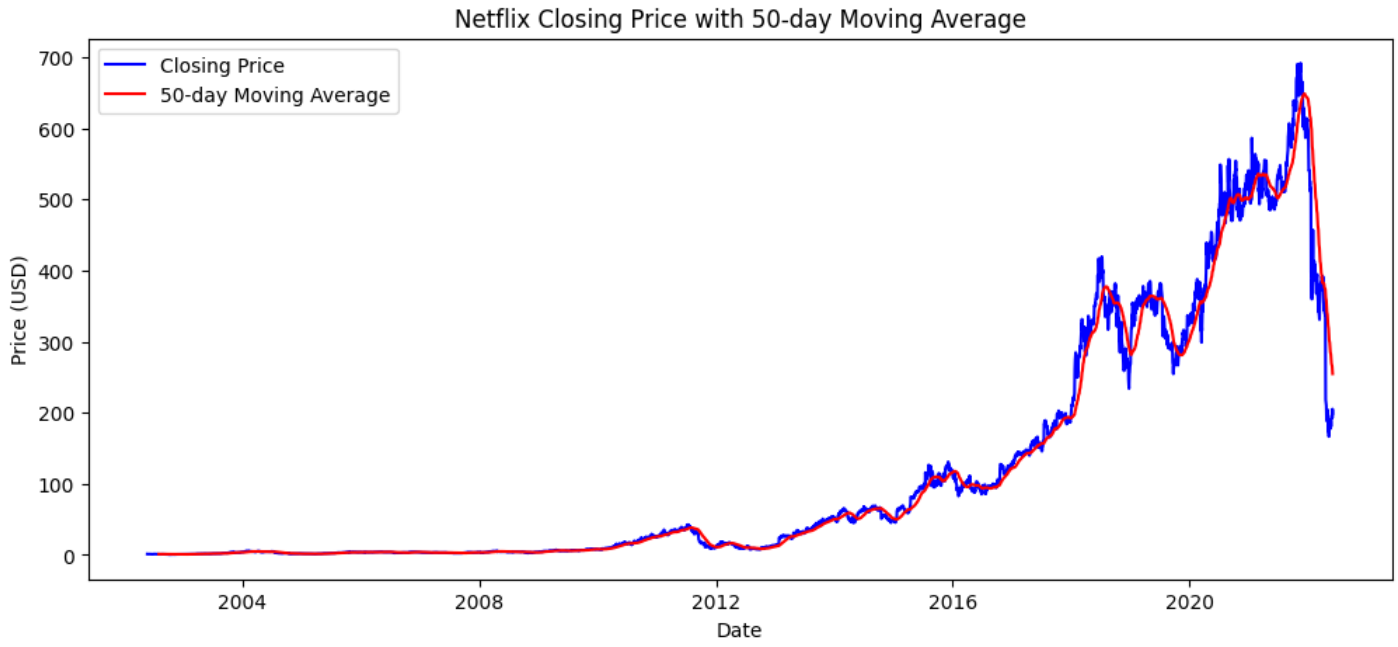
- Load and clean air pollution data (2012-2021).
- Apply **aggregation** (monthly/yearly averages) to estimate trends.
- Use **moving average smoothing** to reduce noise.
- Apply **exponential smoothing** to highlight trends

CODE:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Load the data
df_nflx = pd.read_csv('NFLX (1).csv', parse_dates=['Date'])

# Data exploration and preparation
df_nflx = df_nflx.set_index('Date')
df_nflx['Adj Close'] = df_nflx['Adj Close'].ffill()

# Trend analysis: Rolling statistics
df_nflx['Rolling_Mean'] = df_nflx['Adj Close'].rolling(window=30).mean()
df_nflx['Rolling_Std'] = df_nflx['Adj Close'].rolling(window=30).std()

# Feature engineering: Monthly averages and moving averages
monthly_avg = df_nflx['Adj Close'].resample('M').mean()
df_nflx['MA_short'] = df_nflx['Adj Close'].rolling(window=7).mean()
df_nflx['MA_long'] = df_nflx['Adj Close'].rolling(window=90).mean()

# Data visualization
```

```
# (Code for 5 graphs is already present in the notebook)

# Model evaluation
df_comparison = df_nflx[['Adj Close', 'Rolling_Mean', 'MA_short', 'MA_long']].dropna()
mae_rolling = mean_absolute_error(df_comparison['Adj Close'], df_comparison['Rolling_Mean'])
mse_rolling = mean_squared_error(df_comparison['Adj Close'], df_comparison['Rolling_Mean'])
mae_short = mean_absolute_error(df_comparison['Adj Close'], df_comparison['MA_short'])
mse_short = mean_squared_error(df_comparison['Adj Close'], df_comparison['MA_short'])
mae_long = mean_absolute_error(df_comparison['Adj Close'], df_comparison['MA_long'])
mse_long = mean_squared_error(df_comparison['Adj Close'], df_comparison['MA_long'])

results = {
    'Method': ['Rolling_Mean', 'MA_short', 'MA_long'],
    'MAE': [mae_rolling, mae_short, mae_long],
    'MSE': [mse_rolling, mse_short, mse_long]
}
results_df = pd.DataFrame(results)
```

OUTPUT:



RESULT:

Thus, the program using the time series data implementation has been done successfully.