3. Implement programs for check stationary time series data.

EX.N0 : 3	**Implement programs for check stationary time series data.**
DATE : 01/02/2025	

AIM:

Implement programs for check stationary time series data.

PROGRAM:

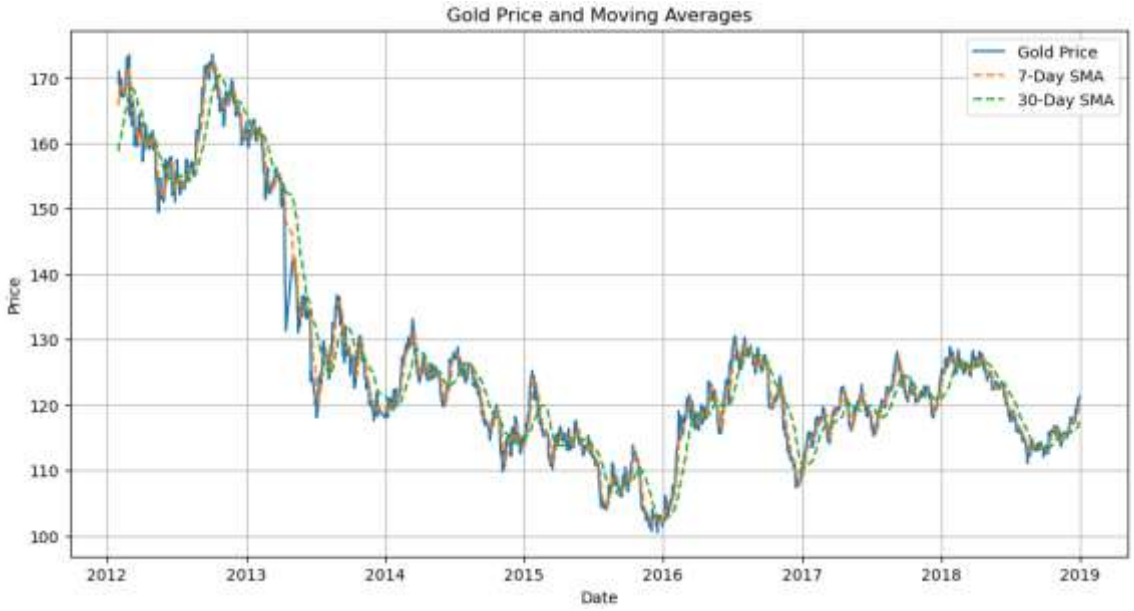
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
def load_data(filepath):
    try:
        data = pd.read_csv(filepath, parse_dates=True, index_col='Date')
        print("Dataset loaded successfully.")
        return data
    except Exception as e:
        print(f"Error loading dataset: {e}")
        return None
def clean_data(data):
    print("Initial dataset shape:", data.shape)
    data = data.drop_duplicates()
    data = data.fillna(method='ffill')  # Forward fill missing values
    data = data.fillna(method='bfill')  # Backward fill for remaining missing values
```

```
    data = data.dropna()
    print("Dataset shape after cleaning:", data.shape)
    return data
def preprocess_time_series(data):
    print("Index type:", type(data.index))
    if not isinstance(data.index, pd.DatetimeIndex):
        data.index = pd.to_datetime(data.index)
    data = data.sort_index()


    return data
def feature_engineering(data):
    data['SMA_7'] = data['Close'].rolling(window=7).mean()  # 7-day Simple Moving Average
    data['SMA_30'] = data['Close'].rolling(window=30).mean()  # 30-day Simple Moving
Average
    data['Lag_1'] = data['Close'].shift(1)  # Previous day's price
    data['Lag_7'] = data['Close'].shift(7)  # Price a week ago
    data = data.dropna()
    return data
def visualize_data(data):
    plt.figure(figsize=(12, 6))
    plt.plot(data['Close'], label='Gold Price')
    plt.plot(data['SMA_7'], label='7-Day SMA', linestyle='--')
    plt.plot(data['SMA_30'], label='30-Day SMA', linestyle='--')
    plt.title('Gold Price and Moving Averages')
    plt.xlabel('Date')
```

```
    plt.ylabel('Price')
    plt.legend()
    plt.grid()
    plt.show()
def visualize_time_series(data):
    plt.figure(figsize=(14, 7))
    plt.subplot(2, 1, 1)
    plt.plot(data['Close'], label='Gold Price', color='blue')
    plt.title('Gold Price Over Time')
    plt.xlabel('Date')
    plt.ylabel('Price')
    plt.legend()
    plt.grid()
    plt.subplot(2, 1, 2)
    plt.hist(data['Close'], bins=30, color='gold', edgecolor='black')
    plt.title('Distribution of Gold Prices')
    plt.xlabel('Price')
    plt.ylabel('Frequency')

    plt.tight_layout()
    plt.show()
def check_stationarity(data):
    print("Checking stationarity of the time series...")
    result = adfuller(data['Close'])
    print("ADF Statistic:", result[0])
```

```
    print("p-value:", result[1])
    print("Critical Values:")
    for key, value in result[4].items():
        print(f"   {key}: {value}"
    if result[1] <= 0.05:
        print("The time series is stationary (p-value <= 0.05).")
    else:
        print("The time series is not stationary (p-value > 0.05). Consider differencing or other transformations.")
def main():
    filepath = "C:\\Users\\jaya karthick\\Downloads\\archive (1) (1)\\FINAL_USO.csv"
|data = load_data(filepath)
    if data is None:
        return
   data = clean_data(data)
    data = preprocess_time_series(data)
    data = feature_engineering(data)
    visualize_data(data)
    visualize_time_series(data)
    check_stationarity(data)
    print("Processed dataset preview:\n", data.head())
if __name__ == "__main__":
    main()
```

OUTPUT:


Gold Price and Moving Averages
Gold Price
7-Day SMA
30-Day SMA
Price
Date
170
160
150
140
130
120
110
100
2012
2013
2014
2015
2016
2017
2018
2019


Gold Price Over Time
Gold Price
Price
Date
Distribution of Gold Prices
Frequency
Price

```
Checking stationarity of the time series...
ADF Statistic: -2.4912282675214468
p-value: 0.11762319598769361
Critical Values:
   1%: -3.4342322039823197
   5%: -2.863254774066211
   10%: -2.5676829016514233
The time series is not stationary (p-value > 0.05). Consider differencing or other transformations.
```

RESULT:

Thus, the program for Implement programs to check stationary of a time series data is executed successfully.