IMPLEMENTATION PROGRAM FOR VISUALIZING TIME SERIES PLOT

1. Importing Necessary Libraries

```
import pandas as pd  # For data manipulation and analysis
import numpy as np  # For numerical operations
import matplotlib.pyplot as plt  # For data visualization
import seaborn as sns  # For advanced visualizations
import warnings  # To handle warnings
warnings.filterwarnings('ignore')
```

Explanation:

1. `pandas`: Used for handling structured data.
2. `numpy`: Provides numerical computation capabilities.
3. `matplotlib.pyplot`: For creating visualizations.
4. `seaborn`: Enhances data visualization.
5. `warnings.filterwarnings('ignore')`: Suppresses warnings for cleaner output

2. Reading the Data

```
data = pd.read_csv('AirPassengers.csv', header=None)
data.columns = ['Month', 'Passengers']
data['Month'] = pd.to_datetime(data['Month'], format='%Y-%m')
data.set_index('Month', inplace=True)
data.head()
```

Explanation:

1. Reads the dataset from a CSV file.
2. Assigns column names ('Month' and 'Passengers').
3. Converts the 'Month' column to a datetime format.
4. Sets 'Month' as the index for time-series analysis.
5. Displays the first five rows of the dataset.

3. Plotting Time-Series Data

```
data.plot(figsize=(12,4))
plt.title("Time Series of Passenger Data")
plt.xlabel("Year")
plt.ylabel("Number of Passengers")
plt.grid()
plt.show()
```

Explanation:

1. Plots the passenger count over time.
2. Sets title, labels, and grid for better visualization.

4. Handling Missing Values

Mean Imputation

```
data['Passengers'] = data['Passengers'].fillna(data['Passengers'].mean())
data.plot(figsize=(12,4))
plt.title("Time Series with Mean Imputation")
plt.show()
```

Linear interpolation

```
data['Passengers'] = data['Passengers'].interpolate(method='linear')
data.plot(figsize=(12,4))
plt.title("Time Series with Linear Interpolation")
plt.show()
```

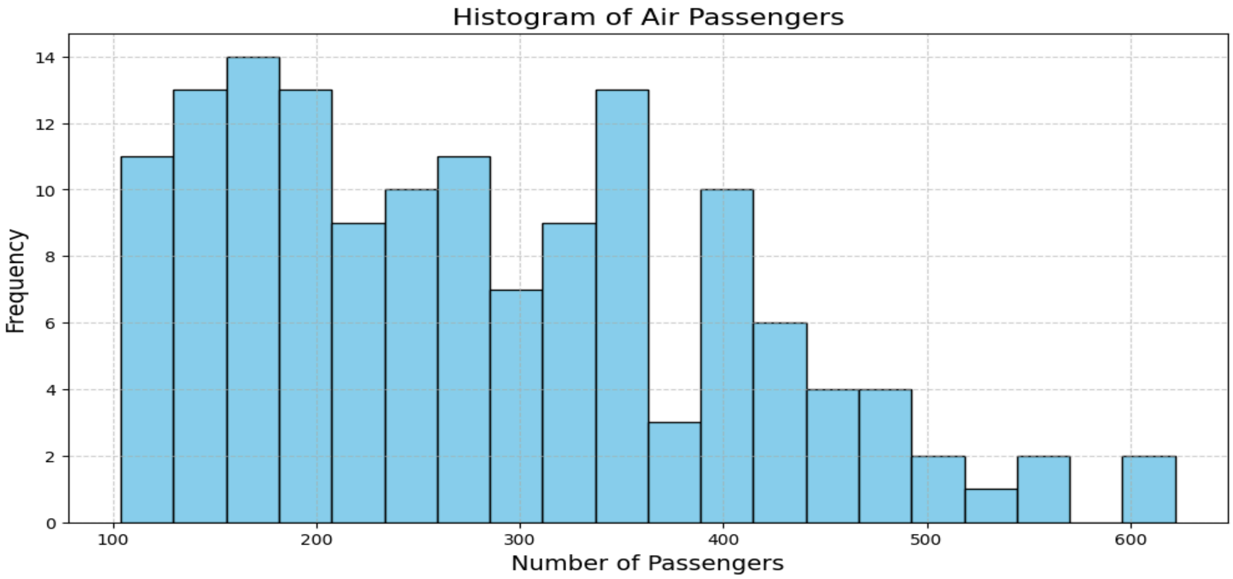
5. Outlier Detection

```
plt.figure(figsize=(12,2))
sns.boxplot(data['Passengers'], whis=1.5)
plt.title("Outlier Detection using Box Plot")
plt.show()
```

6. Histogram Plot

```
data['Passengers'].hist(figsize=(12,4), bins=20, color='skyblue', edgecolor='black')
plt.title("Histogram of Passenger Counts")
plt.xlabel("Number of Passengers")
plt.ylabel("Frequency")
plt.grid()
plt.show()
```

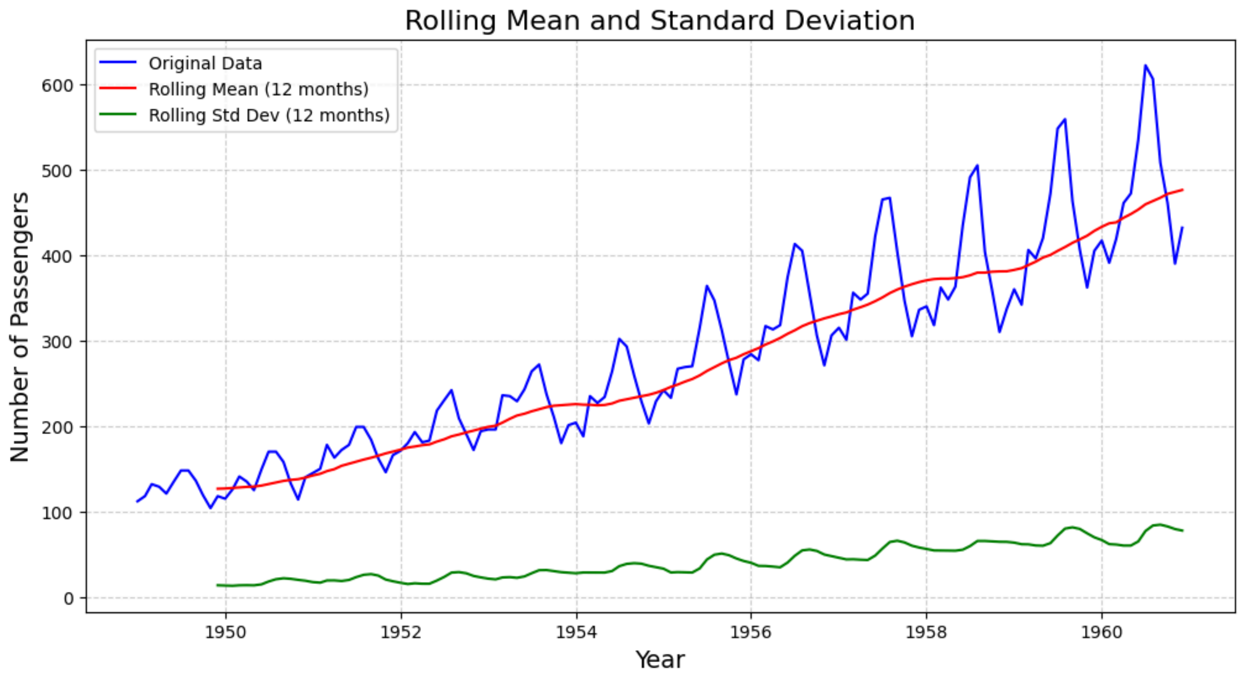
Output



7. Rolling Mean and Standard Deviation Plot

```
rolling_mean = data['Passengers'].rolling(window=12).mean()
rolling_std = data['Passengers'].rolling(window=12).std()

plt.figure(figsize=(12,6))
plt.plot(data['Passengers'], label='Original Data', color='blue')
plt.plot(rolling_mean, label='Rolling Mean (12 months)', color='red')
plt.plot(rolling_std, label='Rolling Std Dev (12 months)', color='green')
plt.title("Rolling Mean and Standard Deviation")
plt.legend()
plt.grid()
plt.show()
```

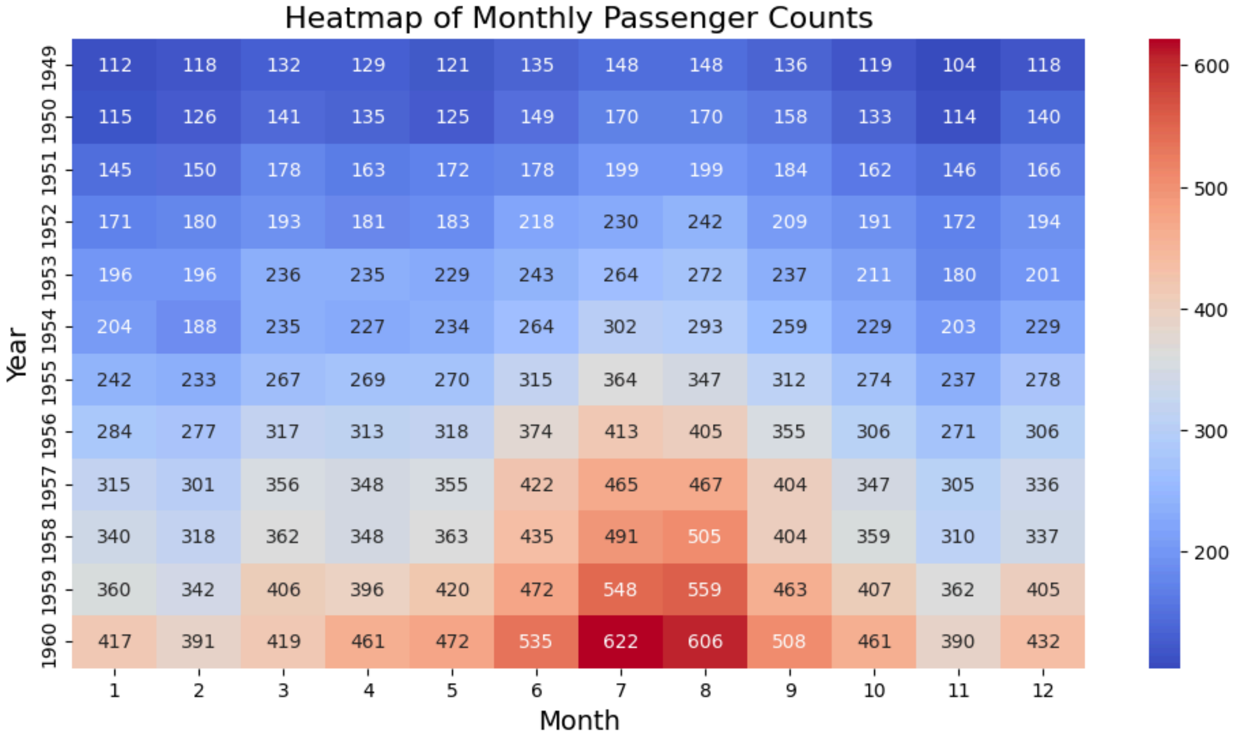
Output



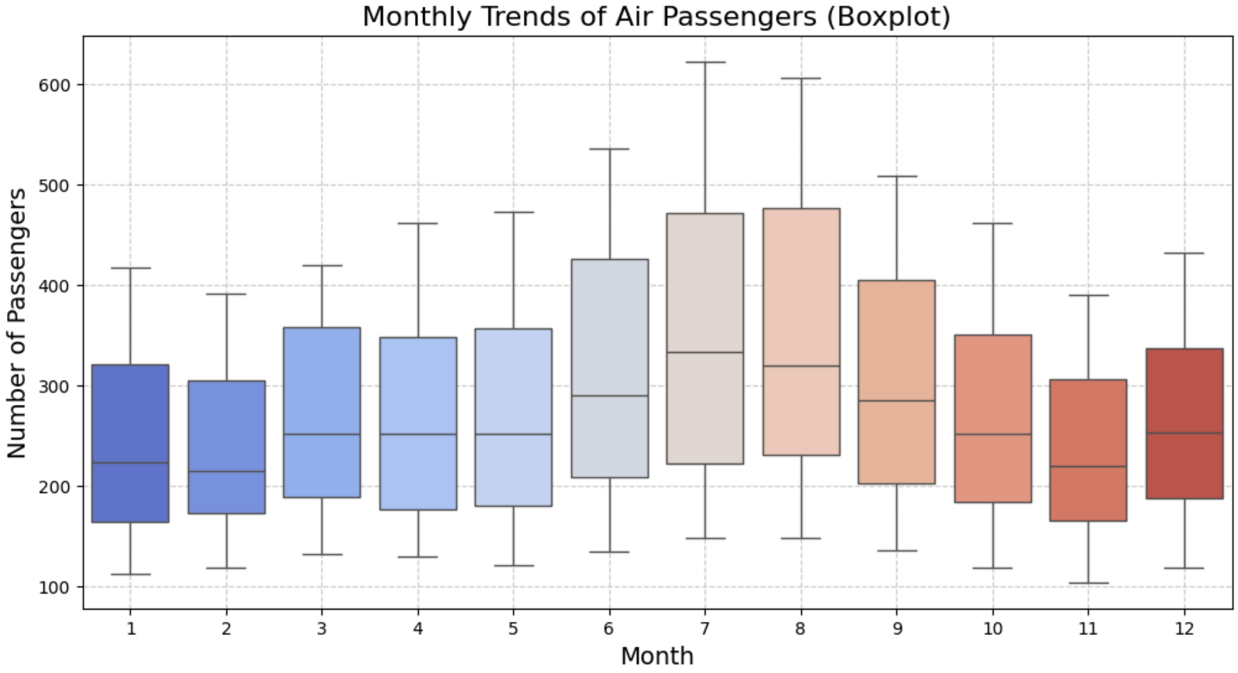
Heatmap for Monthly Passenger Count

```
data['Year'] = data.index.year
data['Month'] = data.index.month

# Pivot the data to get years as rows and months as columns
pivot_table = data.pivot_table(values='Passengers', index='Year', columns='Month',
aggfunc='sum')

# Plot the heatmap
plt.figure(figsize=(12,6))
sns.heatmap(pivot_table, cmap="coolwarm", annot=True, fmt=".0f", linewidths=0.5)
plt.title("Monthly Passenger Count Heatmap")
plt.xlabel("Month")
plt.ylabel("Year")
plt.show()
```

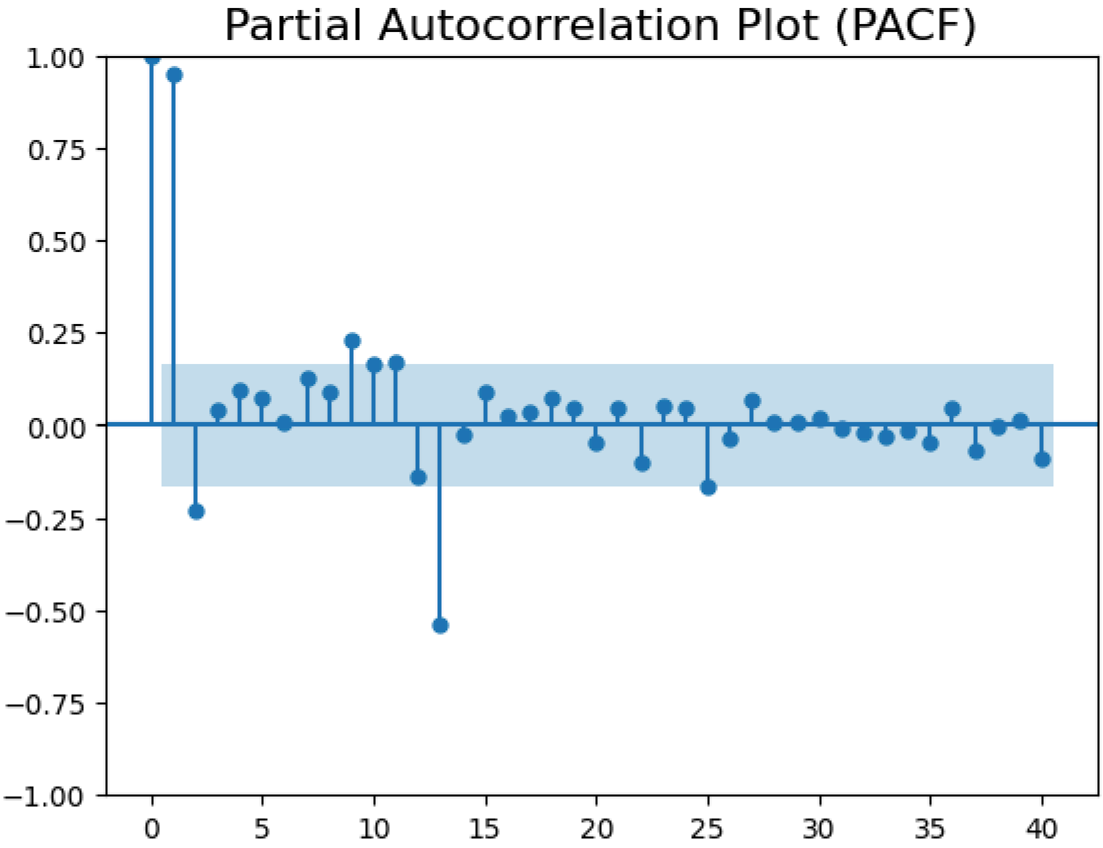
Output



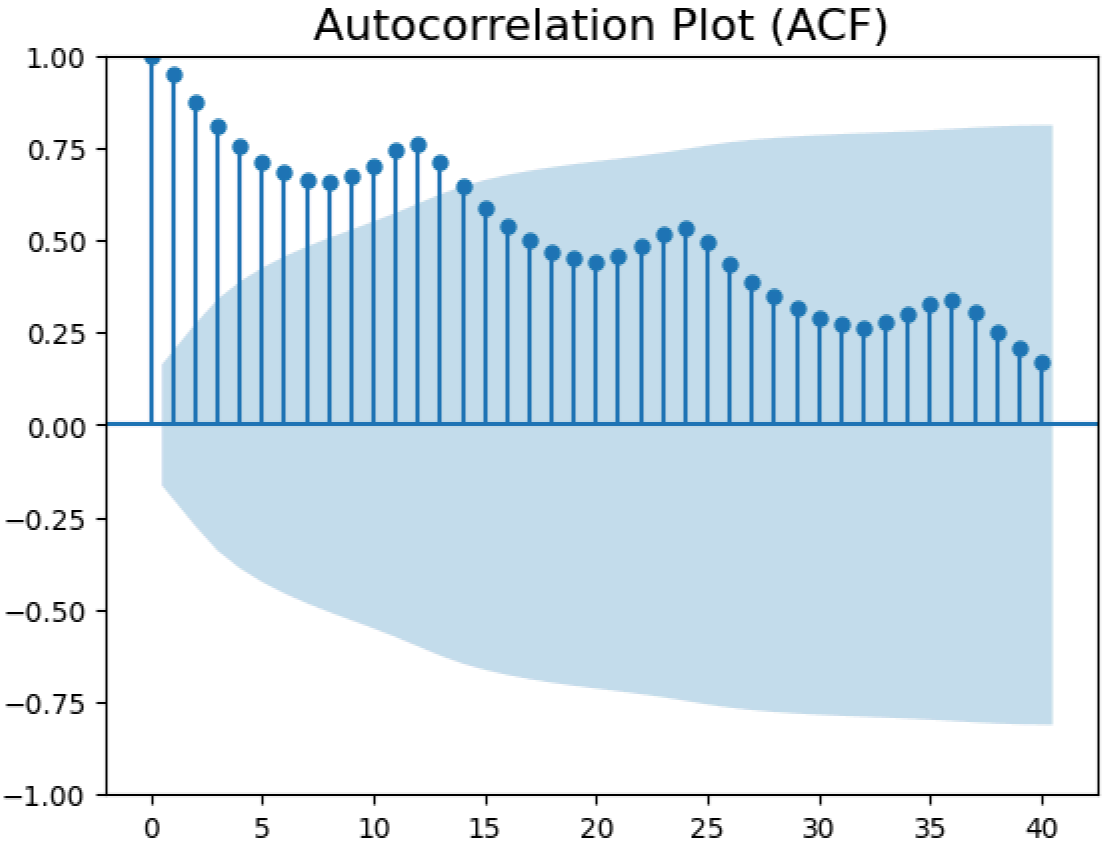
Box plot



Partial auto correlation



Autocorrelation


Autocorrelation Plot (ACF)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

+