Program:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# --- Create sample data ---

np.random.seed(0)

# DataFrame 1

df1 = pd.DataFrame({

'ID': range(1, 6),

'Age': np.random.randint(20, 40, 5),

'Score': np.random.randint(50, 100, 5)

})

# DataFrame 2 (to merge)

df2 = pd.DataFrame({

'ID': [3, 4, 5, 6],

'City': ['NY', 'LA', 'Chicago', 'Houston']

})

print("DataFrame 1:\n", df1)

print("DataFrame 2:\n", df2)

# --- Pandas Operations ---

# 1. Merge

merged\_df = pd.merge(df1, df2, on='ID', how='left')

print("\nMerged DataFrame:\n", merged\_df)

# 2. Sorting by Age

sorted\_df = merged\_df.sort\_values(by='Age')

print("\nSorted by Age:\n", sorted\_df)

# 3. Filtering (Age > 25 and Score > 60)

filtered\_df = merged\_df[(merged\_df['Age'] > 25) & (merged\_df['Score'] > 60)]

print("\nFiltered DataFrame:\n", filtered\_df)

# 4. Adding a new column

merged\_df['Passed'] = merged\_df['Score'] > 60

print("\nAdded 'Passed' Column:\n", merged\_df)

# 5. Grouping and aggregation

grouped = merged\_df.groupby('Passed')['Score'].agg(['mean', 'count'])

print("\nGrouped aggregation:\n", grouped)

# 6. Sorting by multiple columns

multi\_sorted = merged\_df.sort\_values(by=['Passed', 'Score'], ascending=[False, True])

print("\nMulti-column sort:\n", multi\_sorted)

# 7. Drop a column

dropped\_df = merged\_df.drop(columns=['City'])

print("\nDropped 'City' column:\n", dropped\_df)

# 8. Fill missing values (NaN in City)

merged\_df['City'] = merged\_df['City'].fillna('Unknown')

# --- Visualization with matplotlib ---

plt.figure(figsize=(15, 12))

# 1. Line plot (Age vs Score)

plt.subplot(3, 3, 1)

plt.plot(merged\_df['Age'], merged\_df['Score'], marker='o', linestyle='-')

plt.title('Line Plot: Age vs Score')

plt.xlabel('Age')

plt.ylabel('Score')

# 2. Scatter plot (Age vs Score colored by Passed)

plt.subplot(3, 3, 2)

colors = merged\_df['Passed'].map({True: 'green', False: 'red'})

plt.scatter(merged\_df['Age'], merged\_df['Score'], c=colors)

plt.title('Scatter Plot: Age vs Score by Passed')

plt.xlabel('Age')

plt.ylabel('Score')

# 3. Bar chart (Count of Passed/Failed)

plt.subplot(3, 3, 3)

passed\_counts = merged\_df['Passed'].value\_counts()

plt.bar(passed\_counts.index.astype(str), passed\_counts.values, color=['red', 'green'])

plt.title('Bar Chart: Passed Counts')

plt.xlabel('Passed')

plt.ylabel('Count')

# 4. Histogram (Age distribution)

plt.subplot(3, 3, 4)

plt.hist(merged\_df['Age'], bins=5, color='blue', alpha=0.7)

plt.title('Histogram: Age Distribution')

# 5. Pie chart (Passed vs Failed)

plt.subplot(3, 3, 5)

plt.pie(passed\_counts, labels=passed\_counts.index.astype(str), autopct='%1.1f%%', colors=['red', 'green'])

plt.title('Pie Chart: Pass vs Fail')

# 6. Box plot (Scores by Pass/Fail)

plt.subplot(3, 3, 6)

merged\_df.boxplot(column='Score', by='Passed')

plt.title('Box Plot: Scores by Passed')

plt.suptitle('')

# 7. Horizontal bar chart (Mean score by Pass)

plt.subplot(3, 3, 7)

grouped['mean'].plot(kind='barh', color=['red', 'green'])

plt.title('Horizontal Bar: Mean Score by Passed')

# 8. Area plot (Cumulative Score by ID)

plt.subplot(3, 3, 8)

merged\_df.set\_index('ID')['Score'].cumsum().plot(kind='area', color='cyan', alpha=0.5)

plt.title('Area Plot: Cumulative Score by ID')

# 9. Heatmap (Score correlation matrix)

plt.subplot(3, 3, 9)

corr = merged\_df[['Age', 'Score', 'Passed']].corr()

plt.imshow(corr, cmap='coolwarm', interpolation='none')

plt.colorbar()

plt.xticks(range(len(corr)), corr.columns, rotation=45)

plt.yticks(range(len(corr)), corr.columns)

plt.title('Heatmap: Correlation Matrix')

plt.tight\_layout()

plt.show()

Output:

