Use Case 1: Company Sales Branches Comparison

Team members:

1	VTU25115	V HARSHITA
2	VTU25130	KONIKI VASU
3	VTU25132	MUNTIMADUGU SAI SRAVAN
4	VTU25156	RAMIREDDY UDAY SIMHA REDDY
5	VTU25207	YERUVA MANIDEEP REDDY
6	VTU25259	VAJJA MANIDHAR SAI
7	VTU25292	DAMARLA AKSHAY
8	VTU25361	BAJANTHRI SIVA NAGA LOKESH
9	VTU25424	R SATHYA
10	VTU25518	GUDI HANISH
11	VTU25562	SURA GOWRI SHANKAR REDDY

Aim:

To visualize and compare the sales performance of different company branches using Python data visualization libraries such as Matplotlib and Seaborn. This helps in understanding which branch performs best and identifying sales trends.

Algorithm:

Step 1: Import necessary libraries (pandas, numpy, matplotlib, sklearn).

Step 2: Create or load a dataset containing branch names and their sales values.

Step 3: Store data in a Data Frame.

Step 4: Use a bar chart to compare total sales among branches.

Step 5: Use a line plot or pie chart to visualize trends or share distribution.

Step 6: Add titles, axis labels, and legends for clarity.

Step 7: Display the visualization.

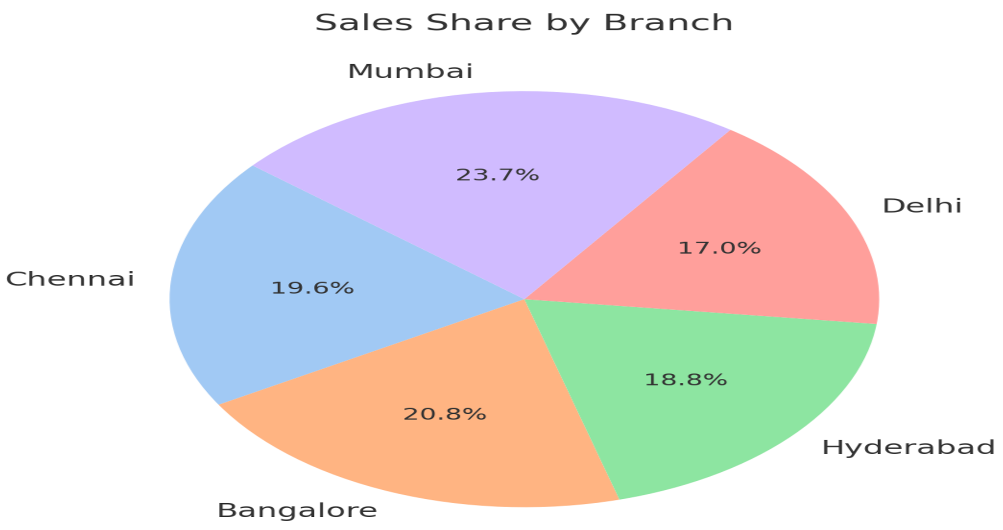
Program:

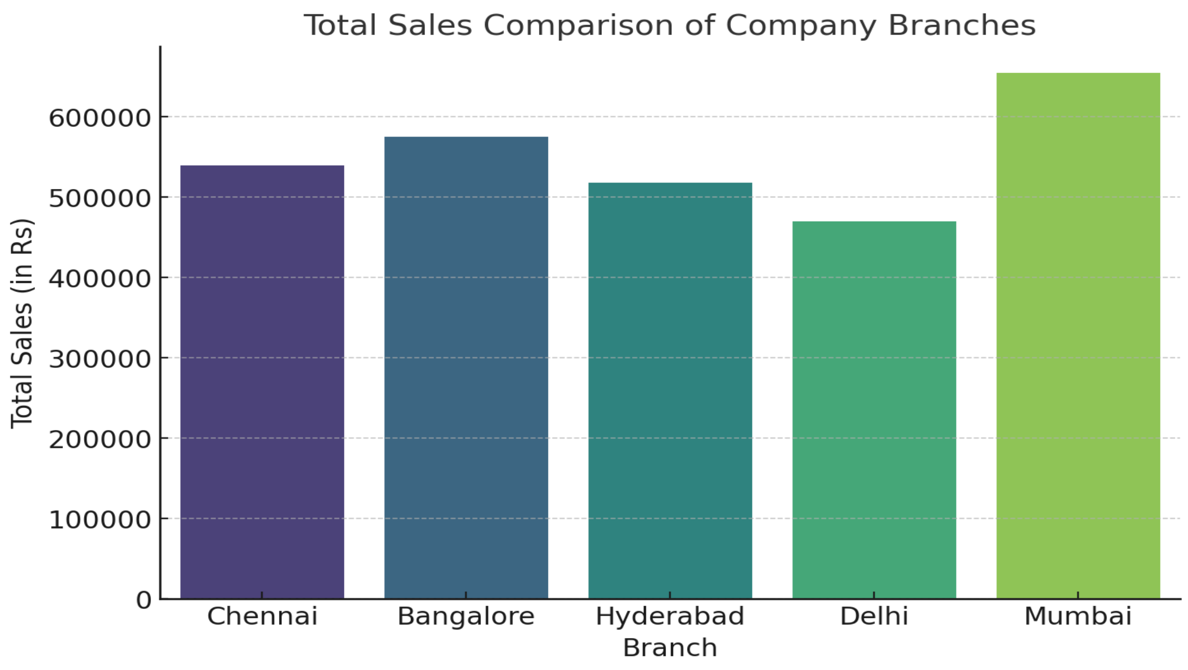
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data = {
    'Branch': ['Chennai', 'Bangalore', 'Hyderabad', 'Delhi', 'Mumbai'],
    'Quarter1_Sales': [120000, 135000, 125000, 110000, 150000],
    'Quarter2_Sales': [130000, 140000, 128000, 115000, 160000],
    'Quarter3_Sales': [140000, 145000, 130000, 120000, 170000],
    'Quarter4_Sales': [150000, 155000, 135000, 125000, 175000]
}
df = pd.DataFrame(data)
df['Total_Sales'] = df.iloc[:, 1:].sum(axis=1)
print("Company Sales Data:\n", df)
plt.figure(figsize=(8,5))
sns.barplot(x='Branch', y='Total_Sales', data=df, palette='viridis')
plt.title('Total Sales Comparison of Company Branches', fontsize=14)
plt.xlabel('Branch')
plt.ylabel('Total Sales (in Rs)')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
plt.figure(figsize=(6,6))
plt.pie(df['Total_Sales'], labels=df['Branch'], autopct='%1.1f%%', startangle=140,
colors=sns.color_palette('pastel'))

plt.title('Sales Share by Branch', fontsize=14)

plt.show()
```

Sample Output:





Result:

The sales comparison shows that the Mumbai branch has the highest sales, while Delhi has the lowest. Visualization using bar and pie charts clearly highlights each branch's performance.

Use Case 2: Creating a Dashboard using COVID-19 data

Team members:

1	VTU25115	V HARSHITA
2	VTU25130	KONIKI VASU
3	VTU25132	MUNTIMADUGU SAI SRAVAN
4	VTU25156	RAMIREDDY UDAY SIMHA REDDY
5	VTU25207	YERUVA MANIDEEP REDDY
6	VTU25259	VAJJA MANIDHAR SAI
7	VTU25292	DAMARLA AKSHAY
8	VTU25361	BAJANTHRI SIVA NAGA LOKESH
9	VTU25424	R SATHYA
10	VTU25518	GUDI HANISH
11	VTU25562	SURA GOWRI SHANKAR REDDY

Aim:

To create an interactive dashboard to visualize COVID-19 data such as confirmed cases, recoveries, and deaths across different countries or regions using Python libraries like Plotly and Pandas.

Algorithm:

Step 1: Import required libraries (pandas, plotly.express).

Step 2: Load the COVID-19 dataset (CSV file or online source).

Step 3: Preprocess the data by removing missing values and selecting required columns.

Step 4: Create visualizations such as bar charts, line charts, and choropleth maps using Plotly.

Step 5: Combine multiple visualizations into a simple dashboard layout.

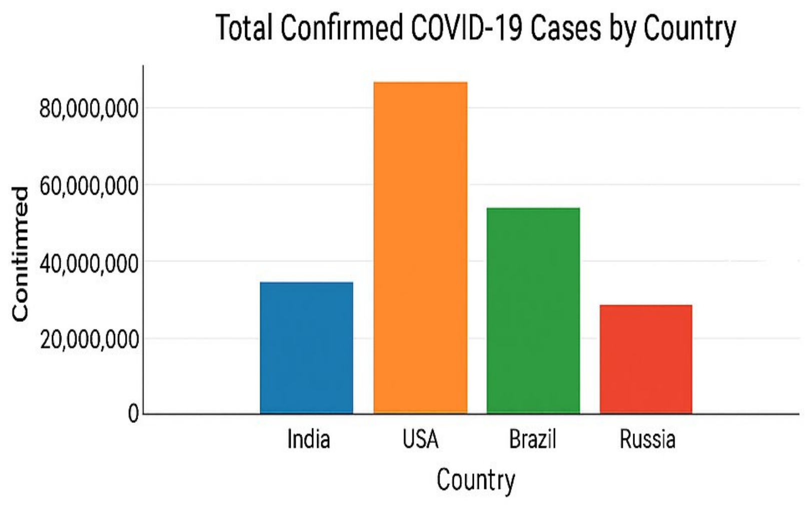
Step 6: Visualize results using bar charts and pie charts.

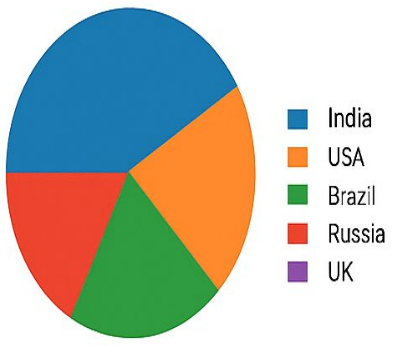
Step 7: Display the interactive dashboard.

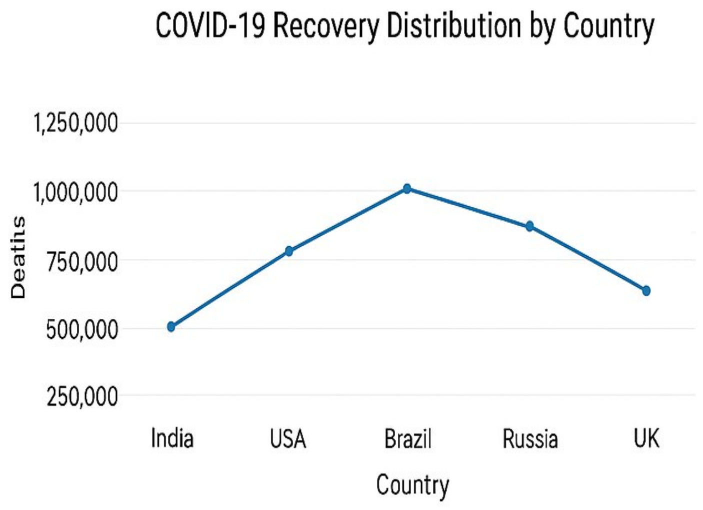
Program:

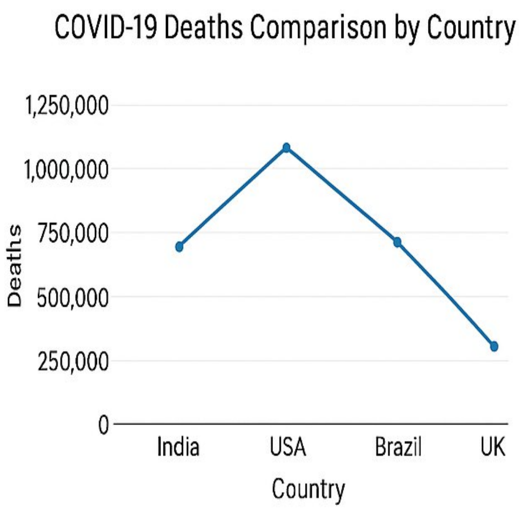
```
import pandas as pd
import plotly.express as px
data = {
    'Country': ['India', 'USA', 'Brazil', 'Russia', 'UK'],
    'Confirmed': [44500000, 97000000, 36000000, 22000000, 24000000],
    'Recovered': [43800000, 95000000, 35000000, 21000000, 23000000],
    'Deaths': [530000, 1200000, 700000, 400000, 500000]
}
df = pd.DataFrame(data)
print("COVID-19 Dataset:\n", df)
fig1 = px.bar(df, x='Country', y='Confirmed', color='Country',
              title='Total Confirmed COVID-19 Cases by Country')
fig1.show()
fig2 = px.pie(df, values='Recovered', names='Country',
            title='Recovered Cases Distribution')
fig2.show()
fig3 = px.line(df, x='Country', y='Deaths', markers=True,
             title='COVID-19 Deaths by Country')
fig3.show()
```

Output:









Result:

A COVID-19 data dashboard was successfully created using Plotly in Python.

The dashboard visually represents confirmed, recovered, and death cases across countries, allowing easy analysis of global COVID-19 trends.