Q1: What is Matplotlib? Why is it used? Name five plots that can be plotted using the Pyplot module of Matplotlib.

ANS- Matplotlib is a popular Python library for creating static, animated, and interactive visualizations in various formats, such as line plots, bar charts, scatter plots, histograms, and more. It provides a flexible and extensive set of tools for creating publication-quality plots and graphs. Matplotlib is commonly used for data analysis, data visualization, and scientific plotting.

The Pyplot module of Matplotlib is a sub-library that provides a simplified interface for creating plots and visualizations. It allows users to create basic plots with minimal code and effort. Here are five types of plots that can be created using the Pyplot module of Matplotlib:

Line Plot: Used to visualize data points connected by straight lines, commonly used for time series data or data with a continuous relationship. Scatter Plot: Used to display individual data points as markers on a 2D plane, helpful for exploring relationships between two variables. Bar Chart: Used to represent categorical data with rectangular bars of varying heights or lengths, often used for comparing different categories. Histogram: Used to visualize the distribution of a single variable by dividing it into bins and counting the frequency of values in each bin. Pie Chart: Used to represent data as a circular chart, with slices (or wedges) corresponding to the proportion of each category in the whole.

Q2: What is a scatter plot? Use the following code to generate data for x and y. Using this generated data plot a scatter plot. import matplotlib.pyplot as plt plt.xlabel('x axis') plt.ylabel('y axis') plt.title('this is title') plt.scatter(x,y) plt.show()

In [ ]:
```
ANS- A scatter plot is a type of plot used to display individual data points as

To generate data for x and y and create a scatter plot, you can use Python code.
import matplotlib.pyplot as plt
import numpy as np

# Generate random data for x and y
np.random.seed(42)  # Setting a seed for reproducibility
x = np.random.rand(50)  # 50 random x values between 0 and 1
y = 2 * x + 1 + 0.2 * np.random.randn(50)  # y values with a linear relationship

# Create a scatter plot
plt.figure(figsize=(8, 6))  # Set the figure size
plt.scatter(x, y, label='Data Points', color='blue', marker='o')
plt.title('Scatter Plot Example')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid(True)

# Display the plot
plt.show()
In this example, we first generate random data for x and create y values with a
```

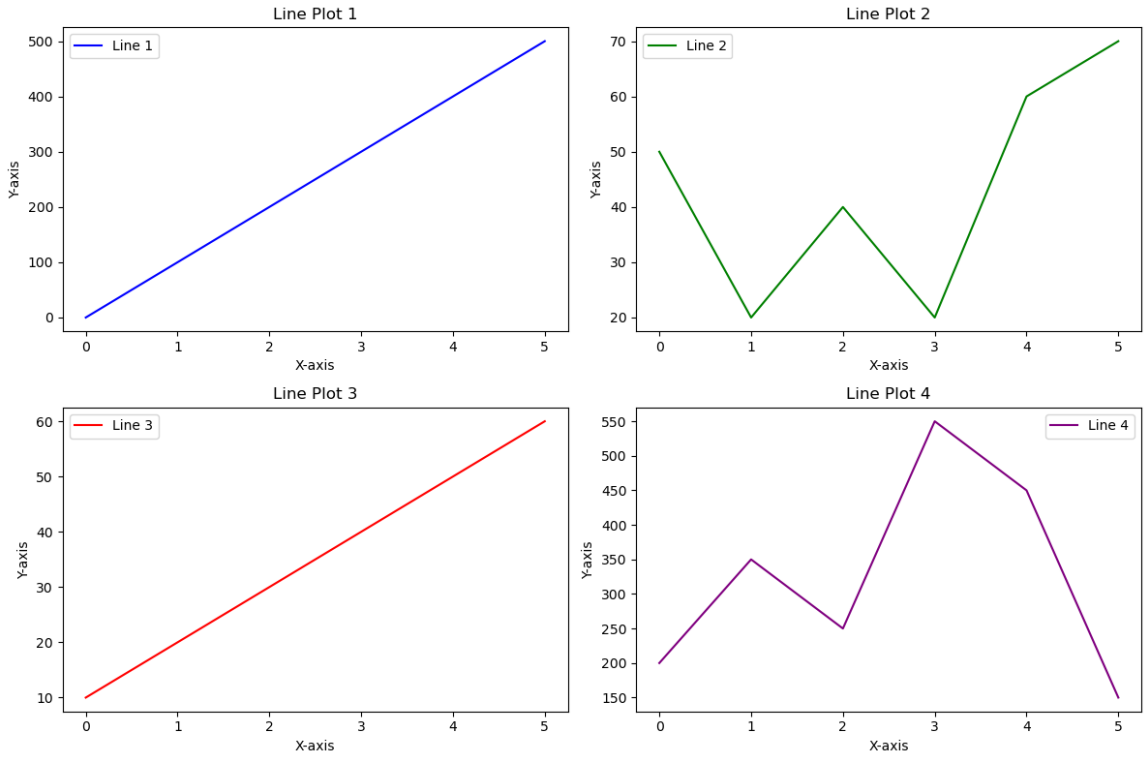
Q3: Why is the subplot() function used? Draw four line plots using the subplot() function. Use the following data: import numpy as np For line 1: x = np.array([0, 1, 2, 3, 4, 5]) and y = np.array([0, 100, 200, 300, 400, 500]) For line 2: x = np.array([0, 1, 2, 3, 4, 5]) and y = np.array([50, 20, 40, 20, 60, 70]) For line 3: x = np.array([0, 1, 2, 3, 4, 5]) and y = np.array([10, 20, 30, 40, 50, 60]) For line 4: x = np.array([0, 1, 2, 3, 4, 5]) and y = np.array([200, 350, 250, 550, 450, 150])

ANS-The subplot() function in Matplotlib is used to create multiple subplots (smaller plots) within a single figure. It allows you to arrange and display multiple plots in a grid or other custom layouts within a single figure, making it useful for visualizing multiple datasets or comparing different plots side by side.

Here's an example of how to use the subplot() function to create four line plots:

In [3]:
```
import matplotlib.pyplot as plt
import numpy as np

# Define the data for the four lines
x = np.array([0, 1, 2, 3, 4, 5])

y1 = np.array([0, 100, 200, 300, 400, 500])
y2 = np.array([50, 20, 40, 20, 60, 70])
y3 = np.array([10, 20, 30, 40, 50, 60])
y4 = np.array([200, 350, 250, 550, 450, 150])

# Create subplots
plt.figure(figsize=(12, 8))  # Set the figure size

# Subplot 1
plt.subplot(2, 2, 1)  # 2 rows, 2 columns, 1st subplot
plt.plot(x, y1, label='Line 1', color='blue')
plt.title('Line Plot 1')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()

# Subplot 2
plt.subplot(2, 2, 2)  # 2 rows, 2 columns, 2nd subplot
plt.plot(x, y2, label='Line 2', color='green')
plt.title('Line Plot 2')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()

# Subplot 3
plt.subplot(2, 2, 3)  # 2 rows, 2 columns, 3rd subplot
plt.plot(x, y3, label='Line 3', color='red')
plt.title('Line Plot 3')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()

# Subplot 4
plt.subplot(2, 2, 4)  # 2 rows, 2 columns, 4th subplot
plt.plot(x, y4, label='Line 4', color='purple')
plt.title('Line Plot 4')
```

```
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()

# Adjust layout to prevent overlapping titles and labels
plt.tight_layout()

# Show the subplots
plt.show()
```



In [ ]:
```
Q4: What is a bar plot? Why is it used? Using the following data plot a bar plot
import numpy as np
company = np.array(["Apple", "Microsoft", "Google", "AMD"])
profit = np.array([3000, 8000, 1000, 10000])
```

ANS- A bar plot is a graphical representation of categorical data using rectangular bars, where the length or height of each bar is proportional to the value it represents. Bar plots are used to visualize and compare the values of different categories or groups. They are particularly useful for showing the distribution of discrete data or for making comparisons between discrete categories.

Here's how to create both a bar plot and a horizontal bar plot using your provided data:B

In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np

company = np.array(["Apple", "Microsoft", "Google", "AMD"])
profit = np.array([3000, 8000, 1000, 10000])

# Create a bar plot
plt.figure(figsize=(8, 6))
plt.bar(company, profit, color='skyblue')
plt.title('Bar Plot of Company Profits')
plt.xlabel('Company')
plt.ylabel('Profit (in billions)')
```

```
plt.show()

# Create a horizontal bar plot
plt.figure(figsize=(8, 6))
plt.barh(company, profit, color='lightcoral')
plt.title('Horizontal Bar Plot of Company Profits')
plt.xlabel('Profit (in billions)')
plt.ylabel('Company')
plt.show()V
```

Q5: What is a box plot? Why is it used? Using the following data plot a box plot. box1 = np.random.normal(100, 10, 200) box2 = np.random.normal(90, 20, 200)

In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np

# Generate random data for two boxes
box1 = np.random.normal(100, 10, 200)
box2 = np.random.normal(90, 20, 200)

# Create a box plot
plt.figure(figsize=(8, 6))
plt.boxplot([box1, box2], labels=['Box 1', 'Box 2'], notch=True, patch_artist=Tr
plt.title('Box Plot of Two Distributions')
plt.xlabel('Box')
plt.ylabel('Value')
plt.grid(True)

# Add colors to the boxes
colors = ['lightblue', 'lightgreen']
for patch, color in zip(plt.gca()['boxes'], colors):
    patch.set_facecolor(color)

# Show the box plot
plt.show()
```

In [ ]:
```

```

In [ ]:
```

```