Beginner's Guide to Basic Charts in Python with Matplotlib

Matplotlib is one of the most popular Python libraries for creating visualizations. It allows you to convert raw numbers into clear, meaningful visuals. Below, we'll cover the most common charts, their purpose, and the basic code to create them.

---

1. Line Chart

What it is:

A chart that displays information as a series of points connected by straight lines.

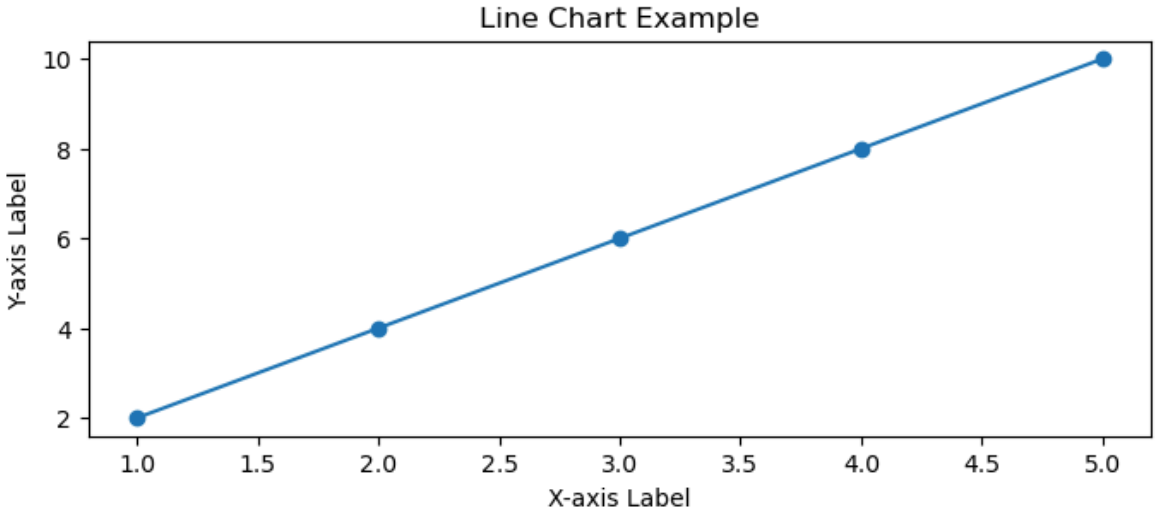
Use cases:

- Showing trends over time
- Comparing multiple data series
- Tracking continuous data

Syntax:

In [2]:
```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]
plt.figure(figsize=(8, 3))  # Width=8 inches, Height=3 inches
plt.plot(x, y, marker='o')
plt.title("Line Chart Example")
plt.xlabel("X-axis Label")
plt.ylabel("Y-axis Label")
plt.show()
```



2. Bar Chart

What it is:

A chart that represents data with rectangular bars.

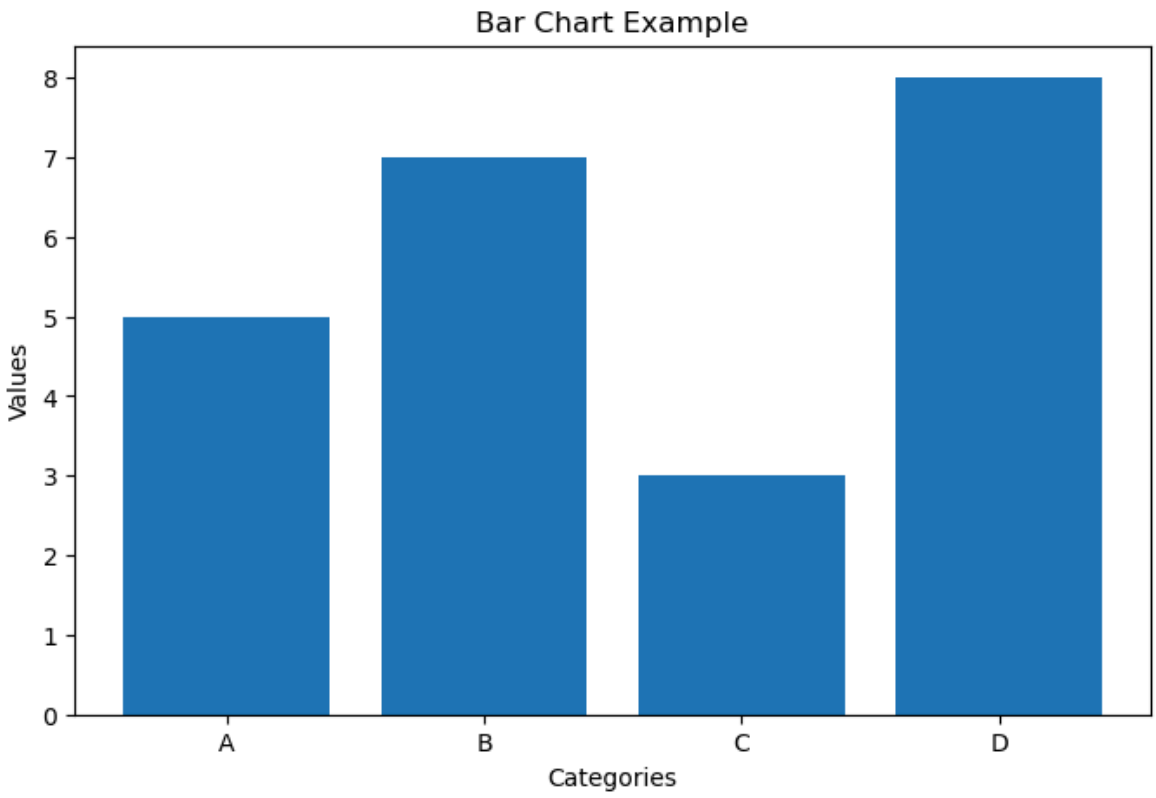
Use cases:

- Comparing quantities across categories
- Displaying discrete values
- Ranking items

Syntax:

In [4]:
```
categories = ['A', 'B', 'C', 'D']
values = [5, 7, 3, 8]

plt.figure(figsize=(8, 5))  # Width=8 inches, Height=5 inches
plt.bar(categories, values)
plt.title("Bar Chart Example")
plt.xlabel("Categories")
plt.ylabel("Values")
plt.show()
```



3. Horizontal Bar Chart

What it is:

Same as a bar chart, but bars run horizontally.

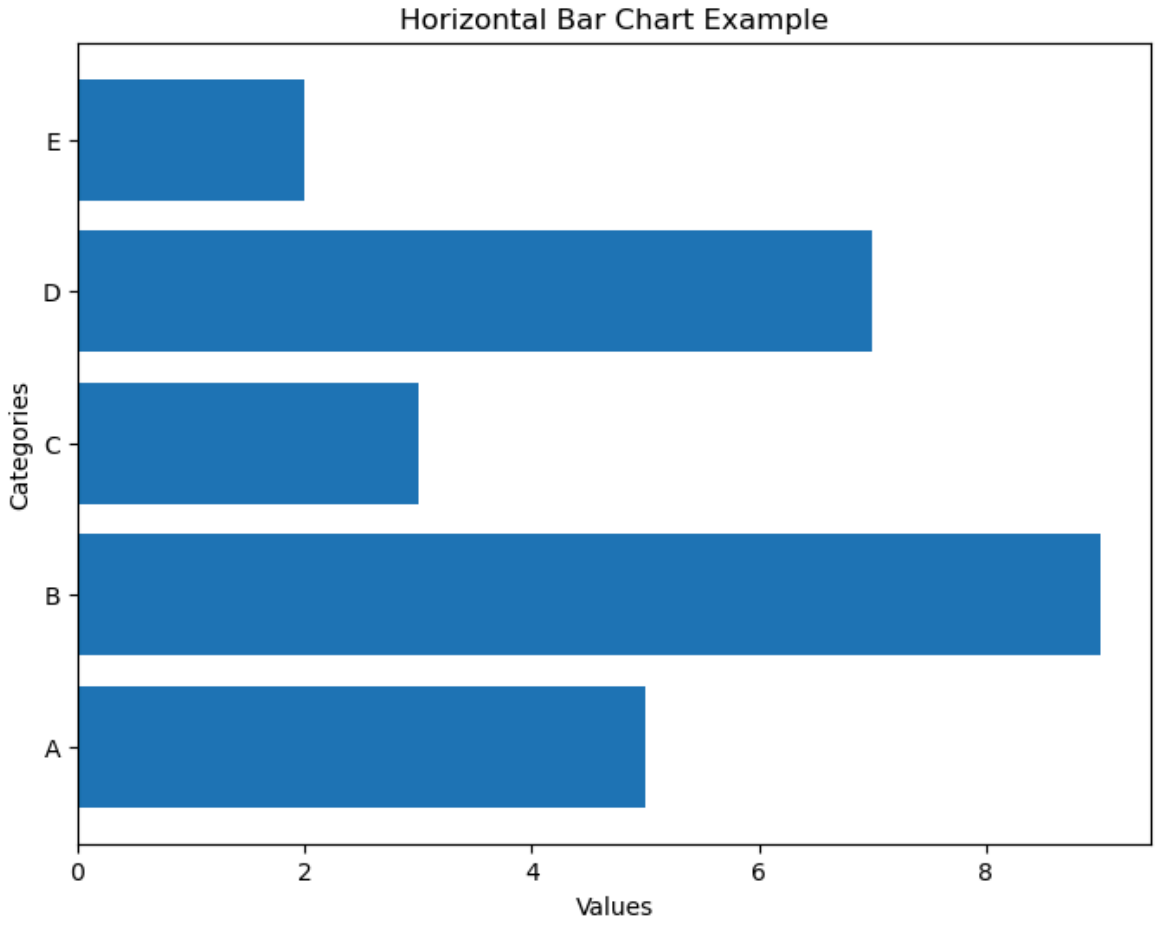
Use cases:

- Comparing many categories (easier to read than vertical bars)
- Visualizing rankings

Syntax:

In [8]:
```
categories = ['A', 'B', 'C', 'D','E']
values = [5, 9, 3, 7,2]

plt.figure(figsize=(8, 6))  # Width=8 inches, Height=6 inches
plt.barh(categories, values)
plt.title("Horizontal Bar Chart Example")
plt.xlabel("Values")
plt.ylabel("Categories")
plt.show()
```



4. Pie Chart

What it is:

A circular chart divided into slices representing proportions.

Use cases:

- Showing percentage breakdowns
- Representing parts of a whole

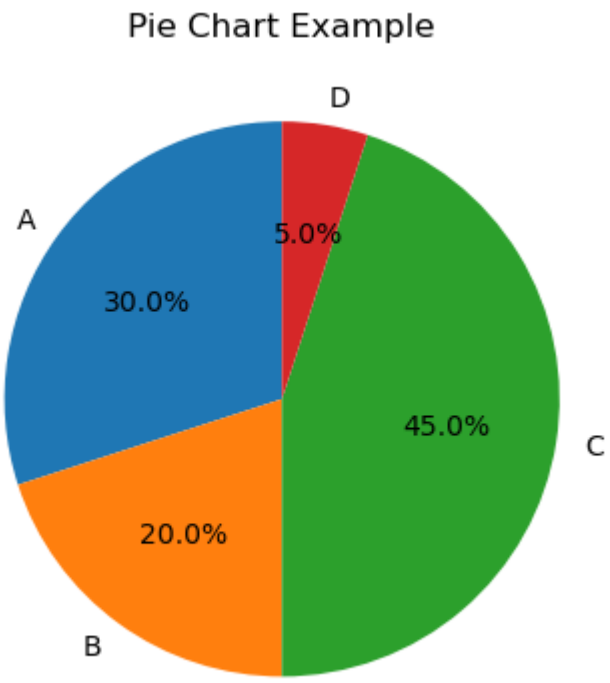
Syntax:

In [11]:
```
sizes = [30, 20, 45, 5]
labels = ['A', 'B', 'C', 'D']

plt.figure(figsize=(8, 4.5))  # Width=8 inches, Height=4.55 inches
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
plt.title("Pie Chart Example")
plt.show()

# autopct='%1.1f%%'
#This controls how percentages are displayed on the chart.
#'%1.1f%%' means:  1.1f → show one digit before and one digit after the decimal.
#%% → print the percent sign.

#startangle=90
#Rotates the pie chart's start position by 90 degrees.
#By default, the first slice starts at the right (0°), but with startangle=90, i
#This can make the chart look more balanced or better aligned
```



5. Scatter Plot

What it is:

A plot that uses Cartesian coordinates to display values for two variables.

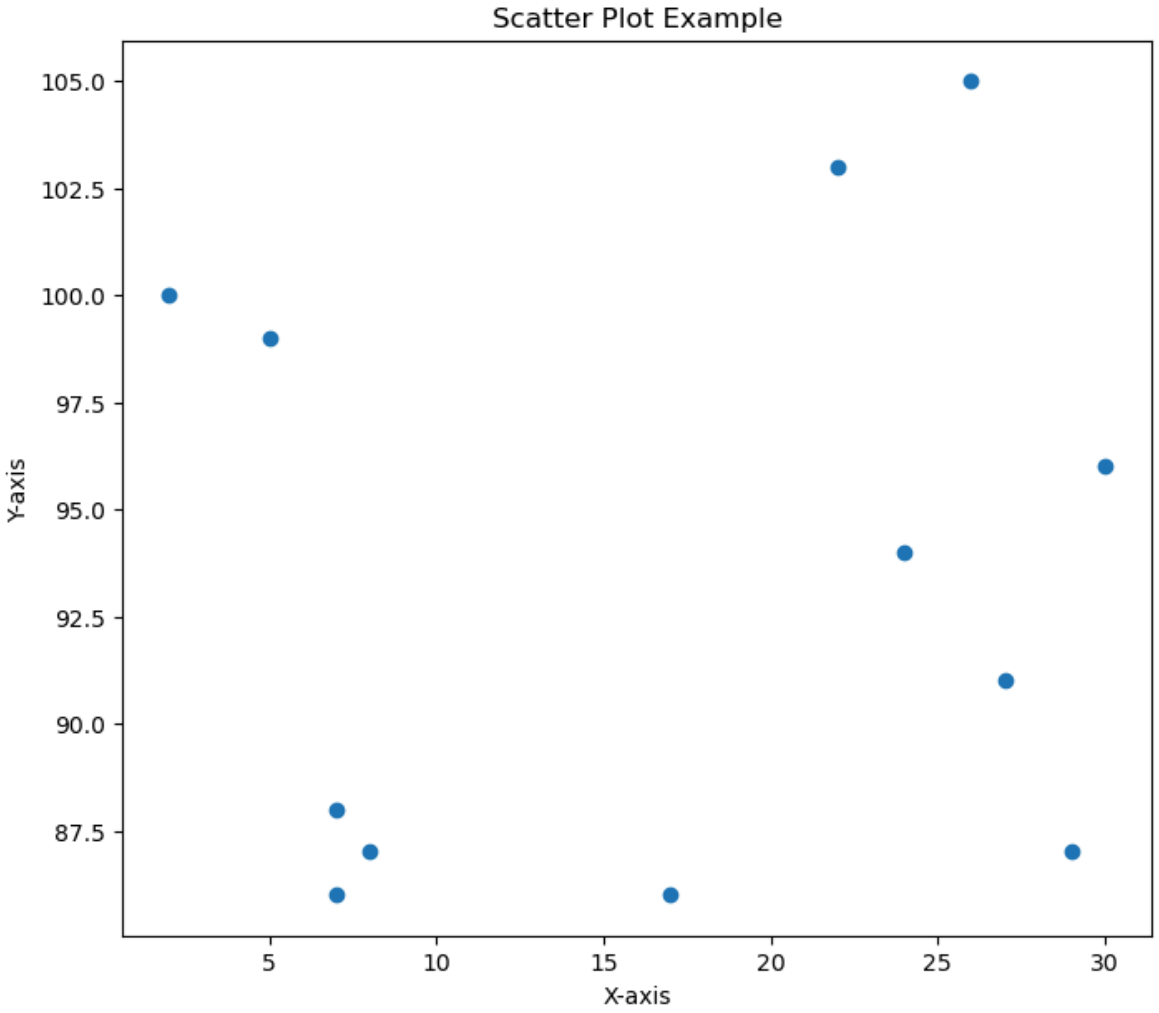
Use cases:

- Identifying relationships or correlations
- Spotting trends and clusters

Syntax:

In [48]:
```
x = [5, 7, 8, 7, 2, 17, 22, 29, 24, 30, 27, 26]
y = [99, 86, 87, 88, 100, 86, 103, 87, 94, 96, 91, 105]

plt.figure(figsize=(8, 7))  # Width=8 inches, Height=7 inches
plt.scatter(x, y)
plt.title("Scatter Plot Example")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```



---

6. Histogram

What it is:

A chart showing the distribution of numerical data.

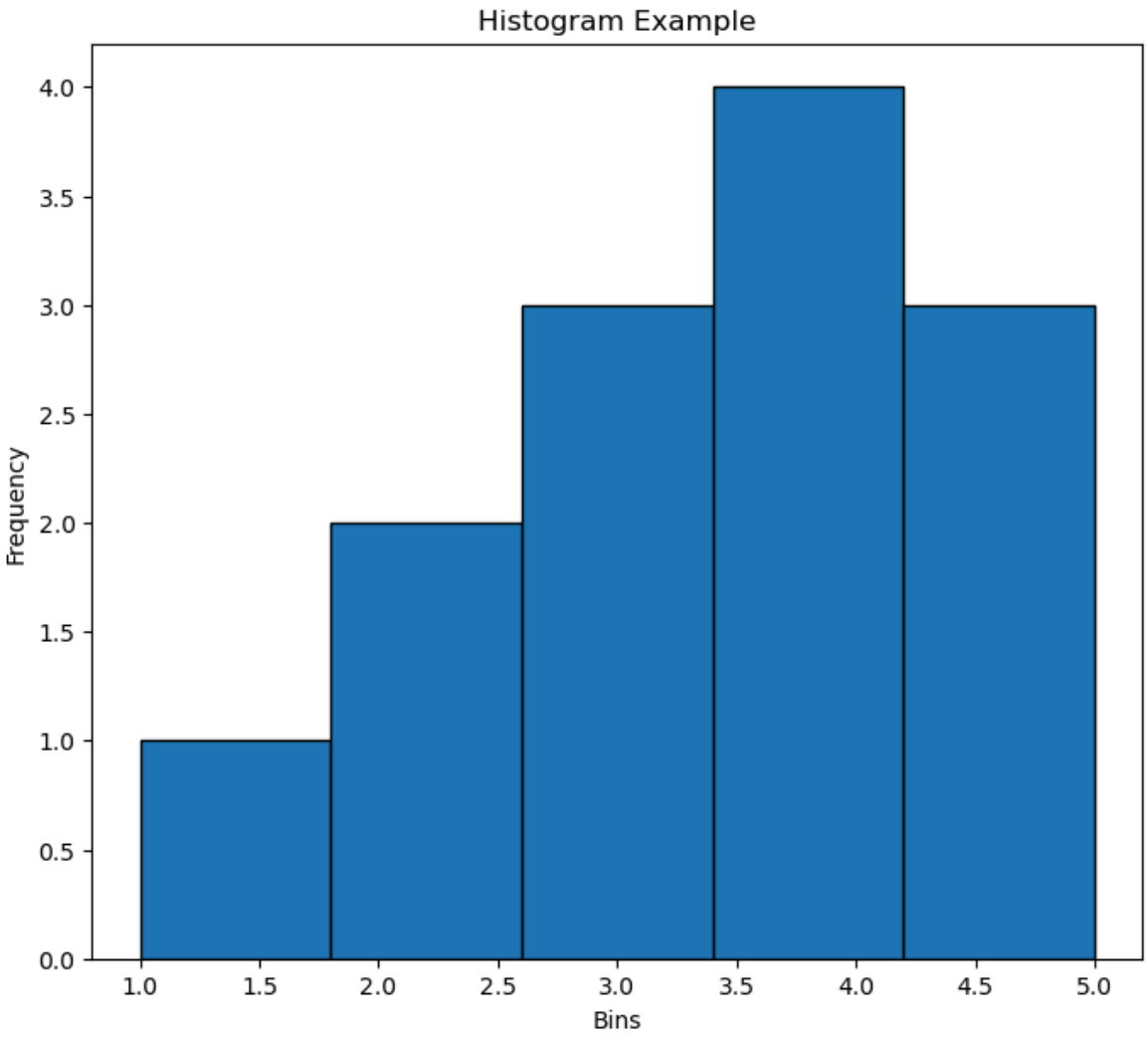
Use cases:

- Analyzing frequency distribution
- Finding patterns in continuous data

Syntax:

In [50]:
```
data = [1,2,2,3,3,3,4,4,4,4,5,5,5]

plt.figure(figsize=(8, 7))  # Width=8 inches, Height=7 inches
plt.hist(data, bins=5,edgecolor='black')
plt.title("Histogram Example")
plt.xlabel("Bins")
plt.ylabel("Frequency")
plt.show()
```



---

7. Stack Plot

What it is:

A plot that shows different data series stacked on top of each other.

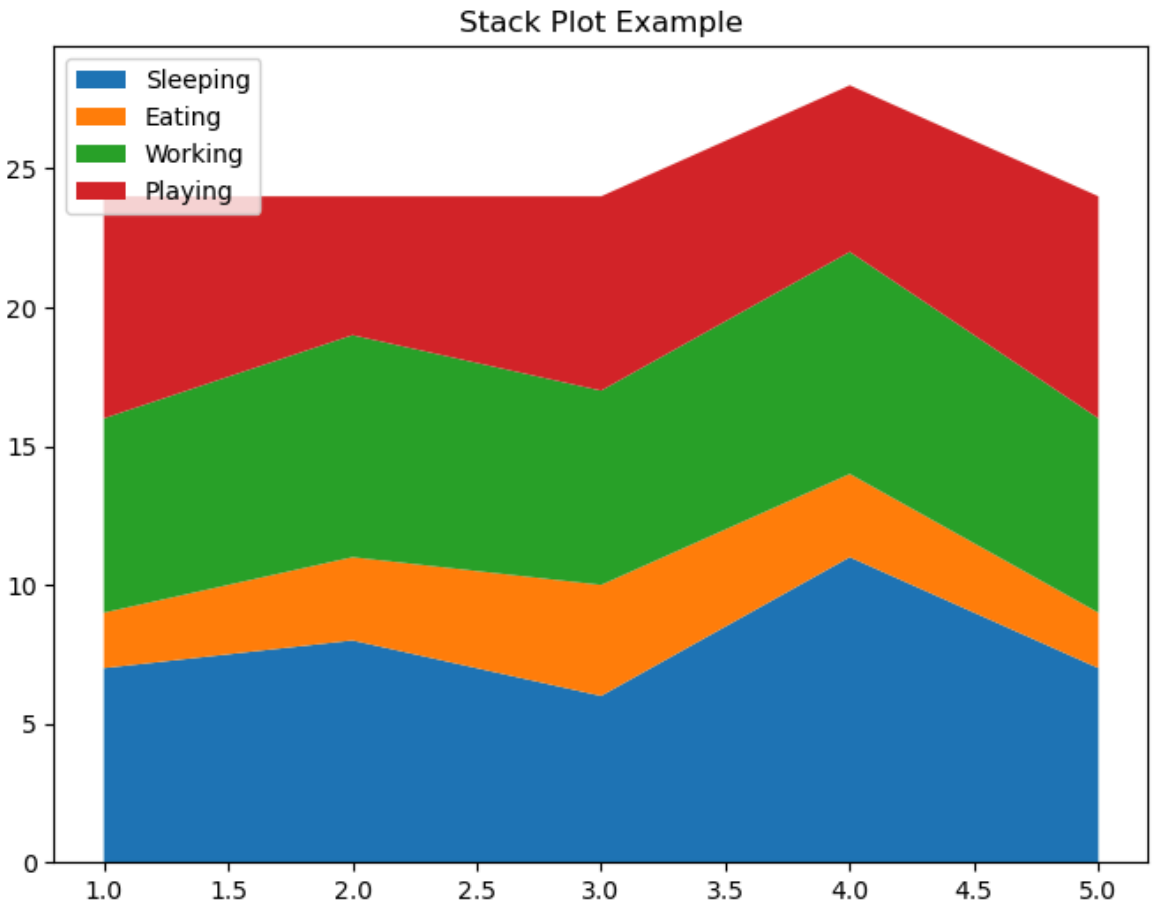
Use cases:

- Showing cumulative data over time
- Comparing contributions to a total

Syntax:

In [52]:
```
days = [1, 2, 3, 4, 5]
sleeping = [7, 8, 6, 11, 7]
eating = [2, 3, 4, 3, 2]
working = [7, 8, 7, 8, 7]
playing = [8, 5, 7, 6, 8]

plt.figure(figsize=(8, 6))  # Width=8 inches, Height=6 inches
plt.stackplot(days, sleeping, eating, working, playing, labels=['Sleeping','Eati
plt.legend(loc='upper left')
plt.title("Stack Plot Example")
plt.show()
```



Final Tips:

- Always label axes and add a title for clarity.
- Use `plt.legend()` for multiple datasets.
- Customize colors, markers, and line styles to improve readability.
- Keep it simple — don’t overload the chart with unnecessary elements.
