

|  |  |  |
|--|--|--|
|  <b>Marwadi</b><br>University<br><small>Marwadi Chandarana Group</small> | <b>NAAC</b><br> <b>A+</b> | <b>Marwadi University</b><br><b>Faculty of Engineering &amp; Technology</b><br><b>Department of Information and Communication Technology</b> |
| <b>Subject: Programming With Python (01CT1309)</b>   | <b>Aim:</b> Practical based on matplotlib  |  |
| <b>Experiment No: 10</b>   | <b>Date:</b>   | <b>Enrollment No: 92400133055</b>  |

## [GITHUB](#)

**Aim:** Practical based on matplotlib

### **IDE:**

What is Matplotlib?

Matplotlib is a popular plotting library in Python used for creating high-quality visualizations and graphs. It offers various tools to generate diverse plots, facilitating data analysis, exploration, and presentation. Matplotlib is flexible, supporting multiple plot types and customization options, making it valuable for scientific research, data analysis, and visual communication. It can create different types of visualization reports like line plots, scatter plots, histograms, bar charts, pie charts, box plots, and many more different plots. This library also supports 3-dimensional plotting.

Installation of Matplotlib

`pip install matplotlib`

To verify the installation, you would have to write the following code chunk:

```
import matplotlib as plt
print(plt.__version__)
```

Line Plots

Line graphs are commonly used to visualize trends over time or relationships between variables.

Example

```
import matplotlib.pyplot as plt
# data to display on plots
x = [3, 1, 3]
y = [3, 2, 1]
plt.plot(x, y)
plt.title("Line Chart")
# Adding the legends
plt.legend(["Line"])
plt.show()
```

Output



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

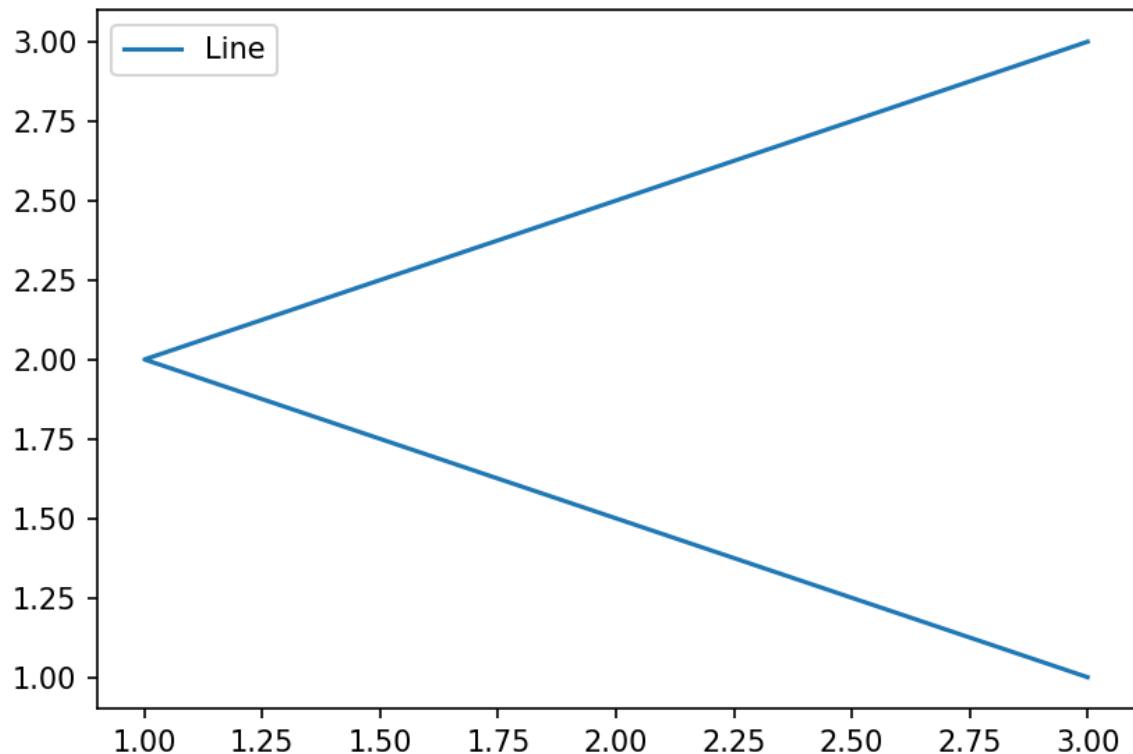
**Enrollment No: 92400133055**

```
1 import matplotlib.pyplot as plt
2 x = [3, 1, 3]
3 y = [3, 2, 1]
4 plt.plot(x, y)
5 plt.title("Line Chart")
6 plt.legend(["Line"])
7 plt.show()
```

Figure 1



Line Chart





**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

Note: xlabel, ylabel, title, plt.xlabel("X-axis") # add X-axis label, plt.ylabel("Y-axis") # add Y-axis label  
plt.title("Any suitable title") # add title, plt.show(), plt.grid(), plt.axhline(y=0, color='k'), plt.axvline(x=0, color='k')

### Stem Plots

A stem plot, also known as a stem-and-leaf plot, is a type of plot used to display data along a number line. Stem plots are particularly useful for visualizing discrete data sets, where the values are represented as “stems” extending from a baseline, with data points indicated as “leaves” along the stems.

#### Example

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0.1, 2 * np.pi, 41)
y = np.exp(np.sin(x))
plt.stem(x, y, use_line_collection = True)
plt.show()
```

Output

```
9  import matplotlib.pyplot as plt
10 import numpy as np
11 x = np.linspace(0.1, 2 * np.pi, 41)
12 y = np.exp(np.sin(x))
13 plt.stem(x, y, use_line_collection = True)
14 plt.show()
15 import matplotlib.pyplot as plt
16 import numpy as np
17 x = np.linspace(0.1, 2 * np.pi, 41)
18 y = np.exp(np.sin(x))
19 plt.stem(x, y, use_line_collection = True)
20 plt.show()
```



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

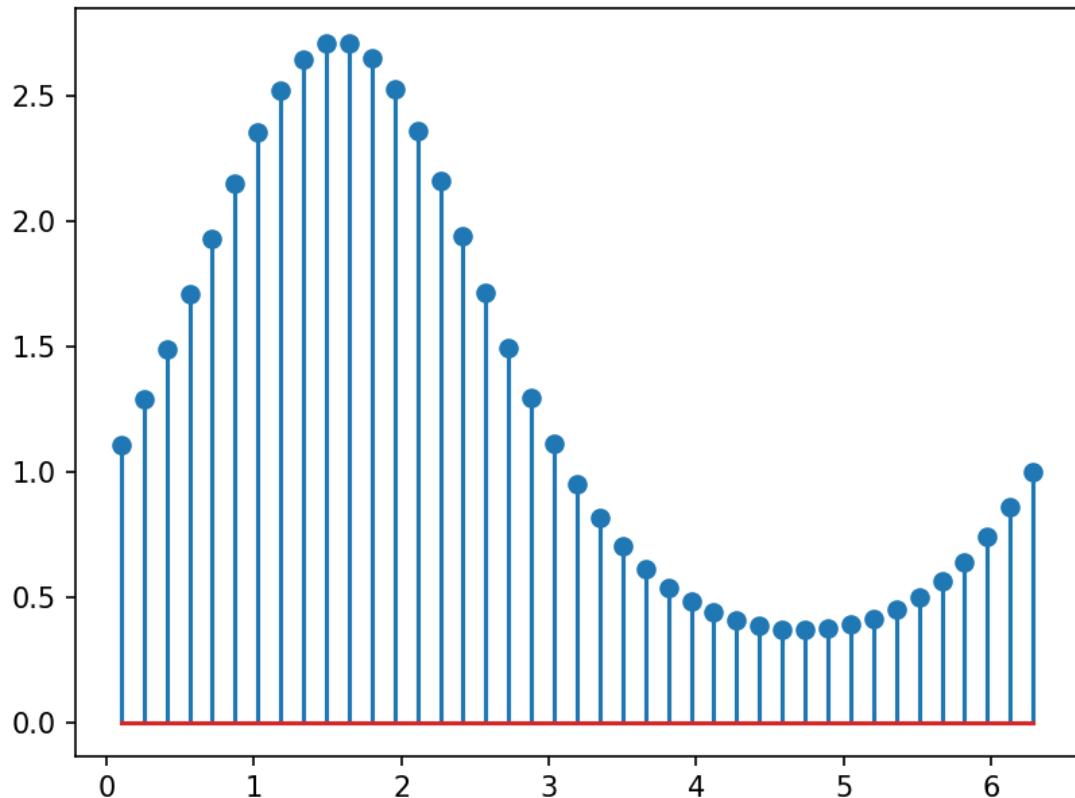
**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

Figure 1

— □ ×



## Bar Charts

A **bar plot** or **bar chart** is a graph that represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent. The bar plots can be plotted horizontally or vertically. A bar chart describes the comparisons between the discrete categories. It can be created using the **bar()** method.

## Example



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

```
import matplotlib.pyplot as plt
```

```
# data to display on plots
```

```
x = [3, 1, 3, 12, 2, 4, 4]
```

```
y = [3, 2, 1, 4, 5, 6, 7]
```

```
# This will plot a simple bar chart
```

```
plt.bar(x, y)
```

```
# Title to the plot
```

```
plt.title("Bar Chart")
```

```
# Adding the legends
```

```
plt.legend(["bar"])
```

```
plt.show()
```

```
Output
```

```
22 import matplotlib.pyplot as plt
23 x = [3, 1, 3, 12, 2, 4, 4]
24 y = [3, 2, 1, 4, 5, 6, 7] |
25 plt.bar(x, y)
26 plt.title("Bar Chart")
27 plt.legend(["bar"])
28 plt.show()
```



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

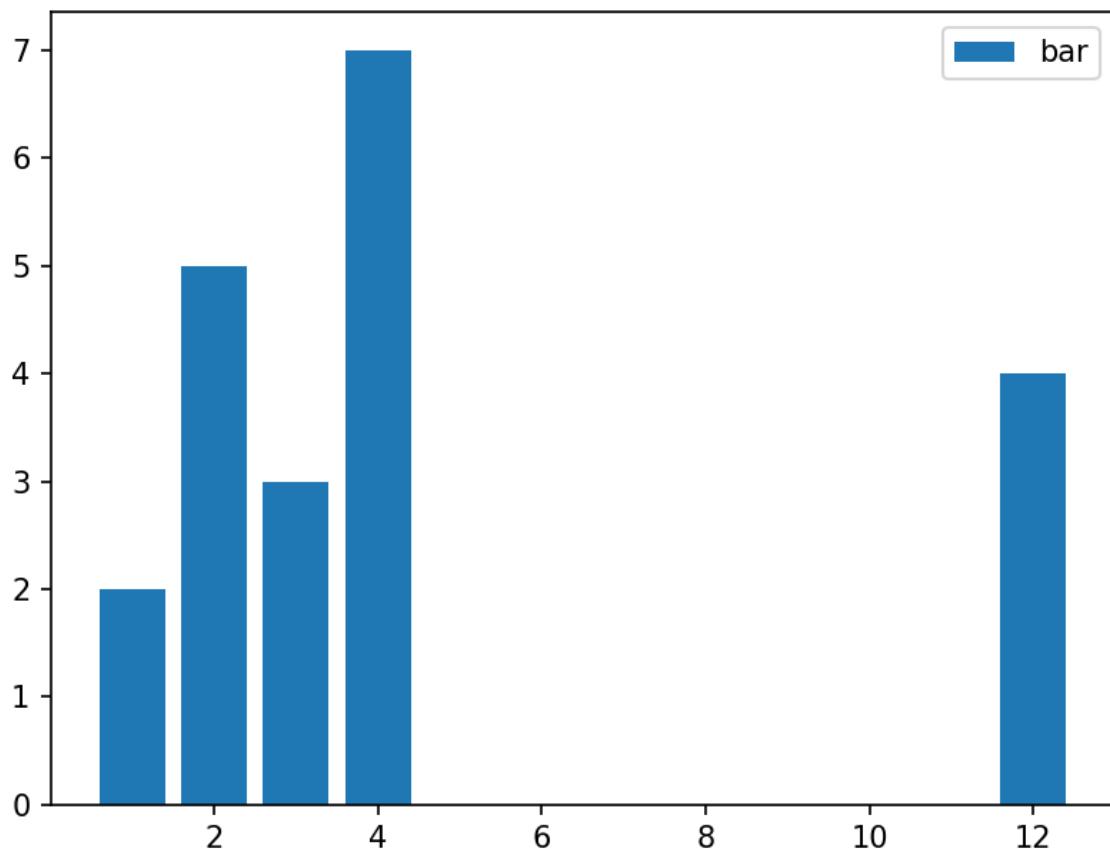
**Date:**

**Enrollment No: 92400133055**

Figure 1



Bar Chart



### Histogram Plot

A histogram is basically used to represent data in the form of some groups. It is a type of bar plot where the X-axis represents the bin ranges while the Y-axis gives information about frequency. To create a histogram the first step is to create a bin of the ranges, then distribute the whole range of the values into a series of intervals, and count the values which fall into each of the intervals. Bins are clearly identified as consecutive, non-overlapping intervals of variables.



**Marwadi University**  
**Faculty of Engineering & Technology**  
**Department of Information and Communication Technology**

**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

```
import matplotlib.pyplot as plt
# data to display on plots
x = [1, 2, 3, 4, 5, 6, 7, 4]
# This will plot a simple histogram
plt.hist(x, bins = [1, 2, 3, 4, 5, 6, 7])
# Title to the plot
plt.title("Histogram")
# Adding the legends
plt.legend(["bar"])
plt.show()
Output
```

```
30 import matplotlib.pyplot as plt
31 x = [1, 2, 3, 4, 5, 6, 7, 4]
32 plt.hist(x, bins = [1, 2, 3, 4, 5, 6, 7])
33 plt.title("Histogram")
34 plt.legend(["bar"])
35 plt.show()
```



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

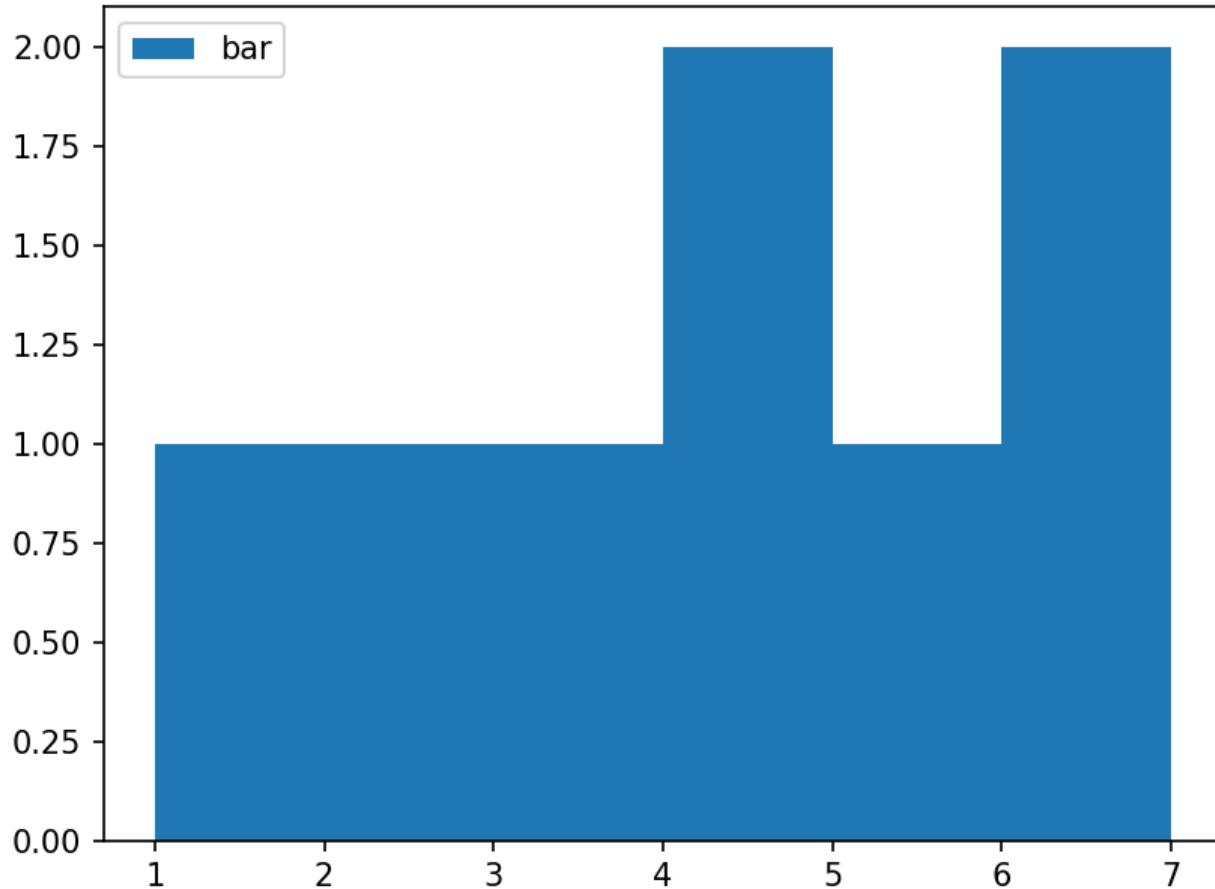
**Date:**

**Enrollment No: 92400133055**

Figure 1

— □ ×

Histogram



### Scatter Plot

Scatter plots are ideal for visualizing the relationship between two continuous variables.



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

**Example**

```
import matplotlib.pyplot as plt
# data to display on plots
x = [3, 1, 3, 12, 2, 4, 4]
y = [3, 2, 1, 4, 5, 6, 7]
# This will plot a simple scatter chart
plt.scatter(x, y)
# Adding legend to the plot
plt.legend("A")
# Title to the plot
plt.title("Scatter chart")
plt.show()
```

Output

```
37 import matplotlib.pyplot as plt
38 x = [3, 1, 3, 12, 2, 4, 4]
39 y = [3, 2, 1, 4, 5, 6, 7]
40 plt.scatter(x, y)
41 plt.legend("A")
42 plt.title("Scatter chart")
43 plt.show()
```



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

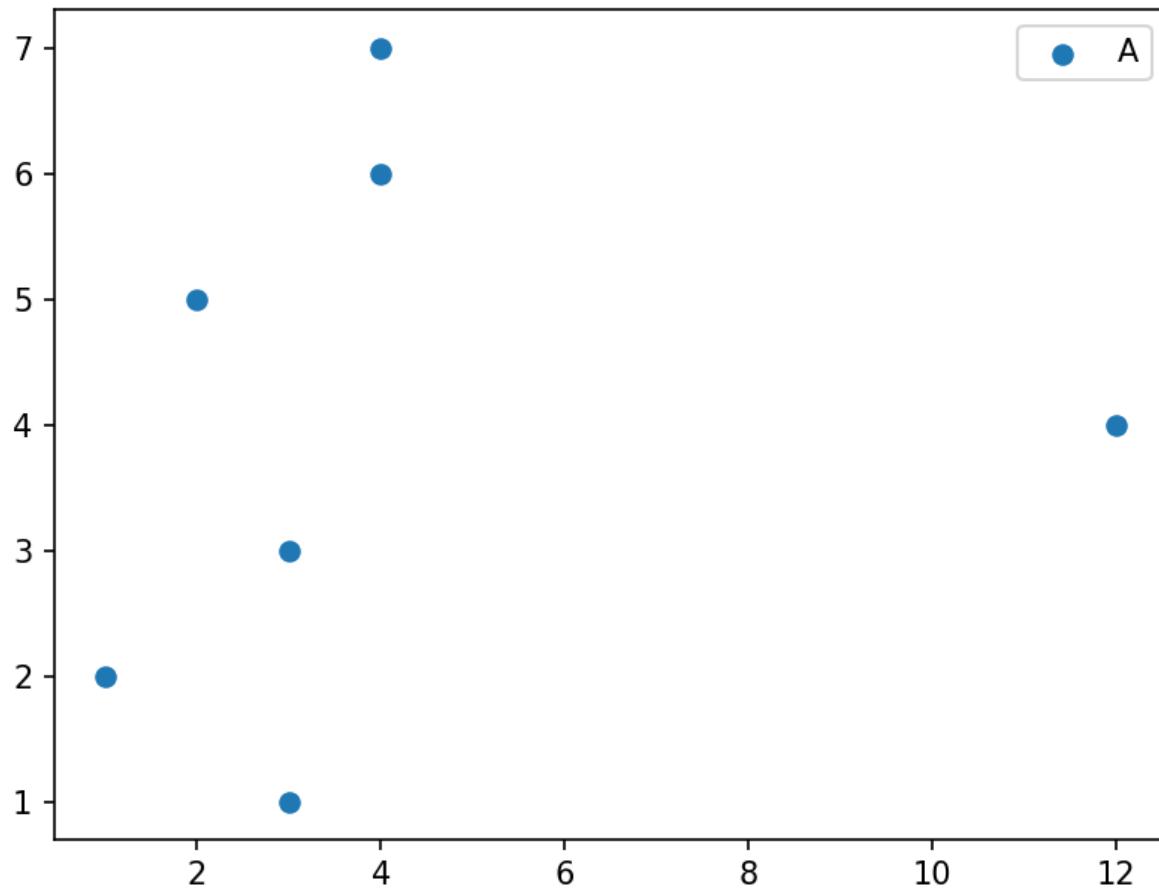
**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

Figure 1

Scatter chart



### Box Plot

A box plot, also known as a box-and-whisker plot, provides a visual summary of the distribution of a dataset. It represents key statistical measures such as the median, quartiles, and potential outliers in a concise and intuitive manner. Box plots are particularly useful for comparing distributions across different groups or identifying anomalies in the data.

### Example



**Marwadi University**  
**Faculty of Engineering & Technology**  
**Department of Information and Communication Technology**

**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

```
import matplotlib.pyplot as plt
import numpy as np
# Creating dataset
np.random.seed(10)
data = np.random.normal(100, 20, 200)
fig = plt.figure(figsize =(10, 7))
# Creating plot
plt.boxplot(data)
```

# show plot

plt.show()

Output

```
45 import matplotlib.pyplot as plt
46 import numpy as np
47 np.random.seed(10)
48 data = np.random.normal(100, 20, 200)
49 fig = plt.figure(figsize =(10, 7))
50 plt.boxplot(data)
51 plt.show()
```



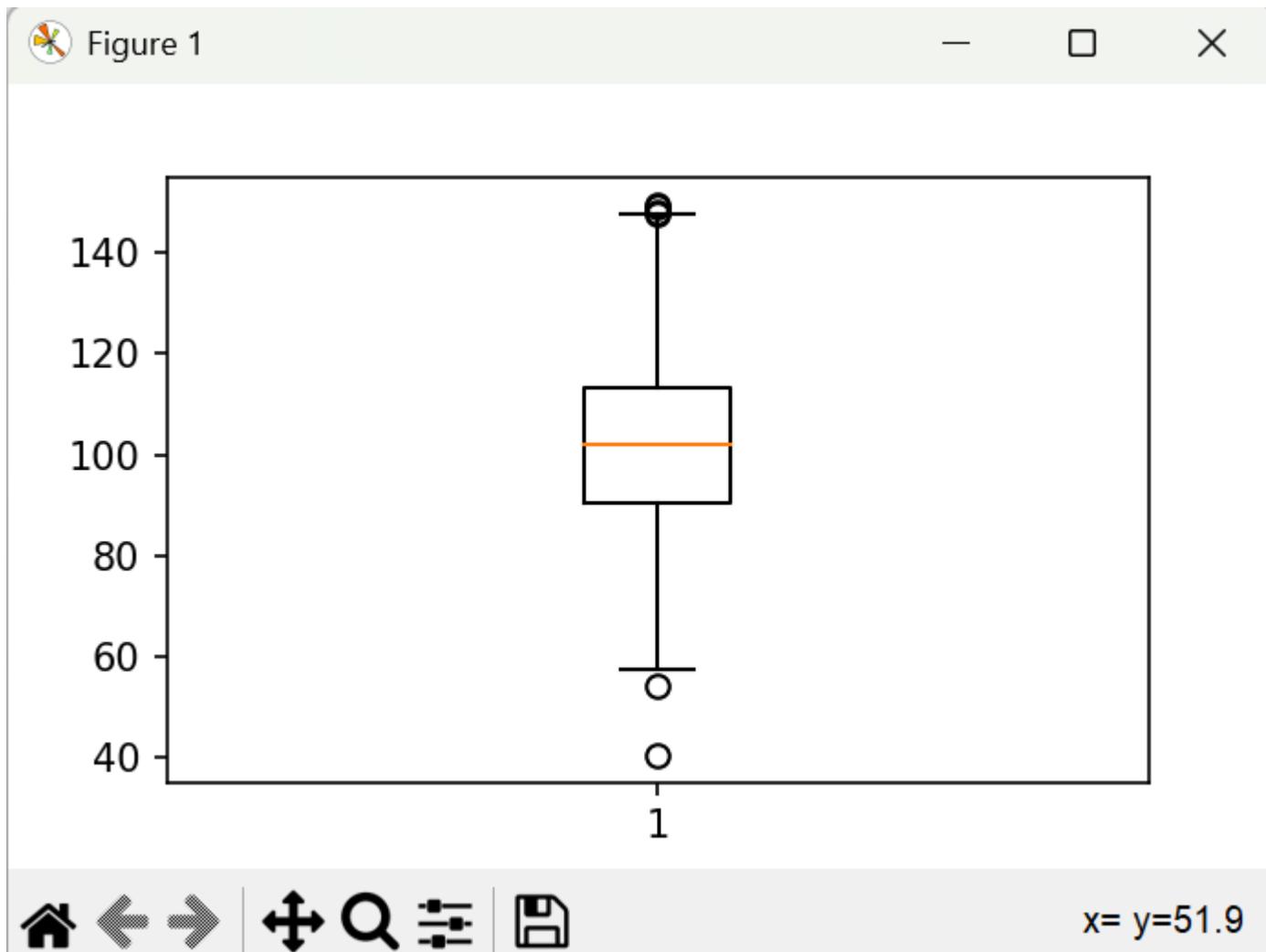
**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**



### Pie chart

A Pie Chart is a circular statistical plot that can display only one series of data. The area of the chart is the total percentage of the given data. The area of slices of the pie represents the percentage of the parts of the data. The slices of pie are called wedges. The area of the wedge is determined by the length of the arc of the wedge.

### Example



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

```
import matplotlib.pyplot as plt
# data to display on plots
x = [1, 2, 3, 4]
# this will explode the 1st wedge
# i.e. will separate the 1st wedge
# from the chart
e =(0.1, 0, 0, 0)
# This will plot a simple pie chart
plt.pie(x, explode = e)
# Title to the plot
plt.title("Pie chart")
plt.show()
Output
```

```
53 import matplotlib.pyplot as plt |
54 x = [1, 2, 3, 4]
55 e =(0.1, 0, 0, 0)
56 plt.pie(x, explode = e)
57 plt.title("Pie chart")
58 plt.show()
```

Figure 1

— □ ×

Pie chart





**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

### Error Plot

Error plots display the variability or uncertainty associated with each data point in a dataset. They are commonly used in scientific research, engineering, and statistical analysis to visualize measurement errors, confidence intervals, standard deviations, or other statistical properties of the data. By incorporating error bars into plots, we can convey not only the central tendency of the data but also the range of possible values around each point.

### Example

```
# importing matplotlib
import matplotlib.pyplot as plt
# making a simple plot
x =[1, 2, 3, 4, 5, 6, 7]
y =[1, 2, 1, 2, 1, 2, 1]
# creating error
y_error = 0.2
# plotting graph
plt.plot(x, y)
plt.errorbar(x, y,yerr = y_error,fmt ='o')
```

Output

```
# importing matplotlib
import matplotlib.pyplot as plt
# making a simple plot
x =[1, 2, 3, 4, 5, 6, 7]
y =[1, 2, 1, 2, 1, 2, 1]
# creating error
y_error = 0.2
# plotting graph
plt.plot(x, y)
plt.errorbar(x, y,yerr = y_error,fmt ='o')
```



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

Note:

lets the creation of a 3D plot in Matplotlib. We can create different types of 3D plots like scatter plots, contour plots, surface plots, etc. Let's create a simple 3D line plot.

Example

```
import matplotlib.pyplot as plt
```

```
x = [1, 2, 3, 4, 5]
```

```
y = [1, 4, 9, 16, 25]
```

```
z = [1, 8, 27, 64, 125]
```

```
# Creating the figure object
```

```
fig = plt.figure()
```

```
# keeping the projection = 3d
```

```
# creates the 3d plot
```

```
ax = plt.axes(projection = '3d')
```

```
ax.plot3D(z, y, x)
```

Output

```
72 import matplotlib.pyplot as plt
73 x = [1, 2, 3, 4, 5]
74 y = [1, 4, 9, 16, 25]
75 z = [1, 8, 27, 64, 125]
76 # Creating the figure object
77 fig = plt.figure()
78 # keeping the projection = 3d
79 # creates the 3d plot
80 ax = plt.axes(projection = '3d')
81 ax.plot3D(z, y, x)
```



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

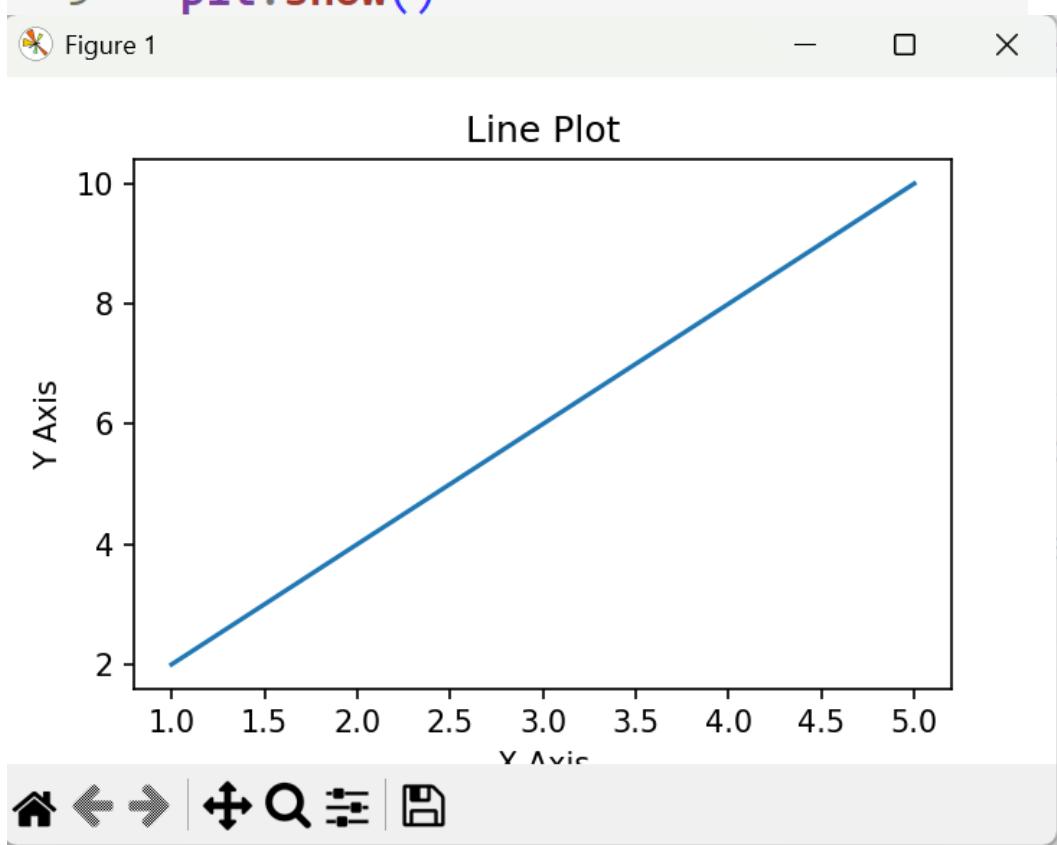
**Enrollment No: 92400133055**

**Post Lab Exercise:**

- a. Write a Python program to draw a line with suitable label in the x axis, y axis and a title.

```
1 # Question 1
2 import matplotlib.pyplot as plt
3 x = [1, 2, 3, 4, 5]
4 y = [2, 4, 6, 8, 10]
5 plt.plot(x, y)
6 plt.xlabel("X Axis")
7 plt.ylabel("Y Axis")
8 plt.title("Line Plot")
9 plt.show()
```

Figure 1





**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

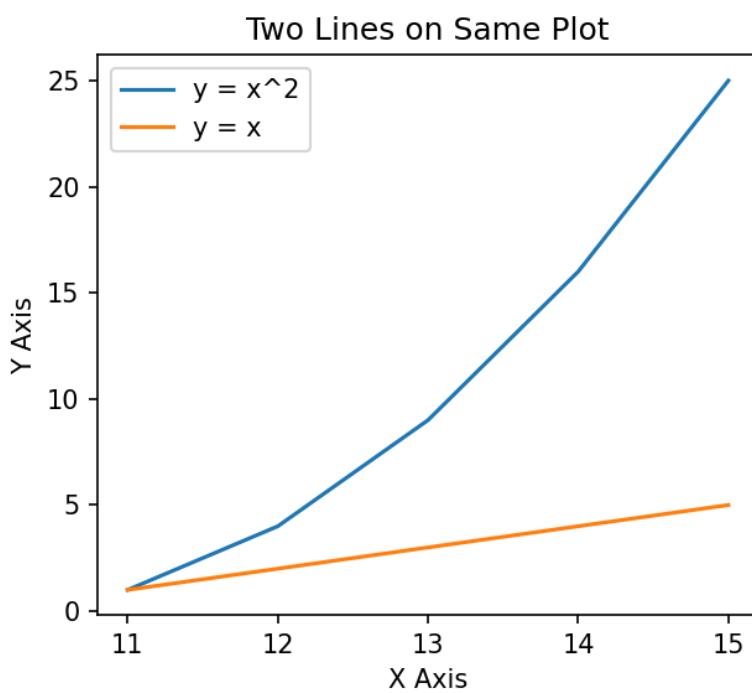
**Enrollment No: 92400133055**

- b. Write a Python program to plot two or more lines on same plot with suitable legends of each line.

```
11  #Question 2
12  import matplotlib.pyplot as plt
13  x = [11, 12, 13, 14, 15]
14  y1 = [1, 4, 9, 16, 25]
15  y2 = [1, 2, 3, 4, 5]
16  plt.plot(x, y1, label="y = x^2")
17  plt.plot(x, y2, label="y = x")
18  plt.xlabel("X Axis")
19  plt.ylabel("Y Axis")
20  plt.title("Two Lines on Same Plot")
21  plt.legend()
22  plt.show()
```

Figure 1

— □ ×





**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

- c. Write a Python programming to display a bar chart of the popularity of programming Languages.

Sample data:

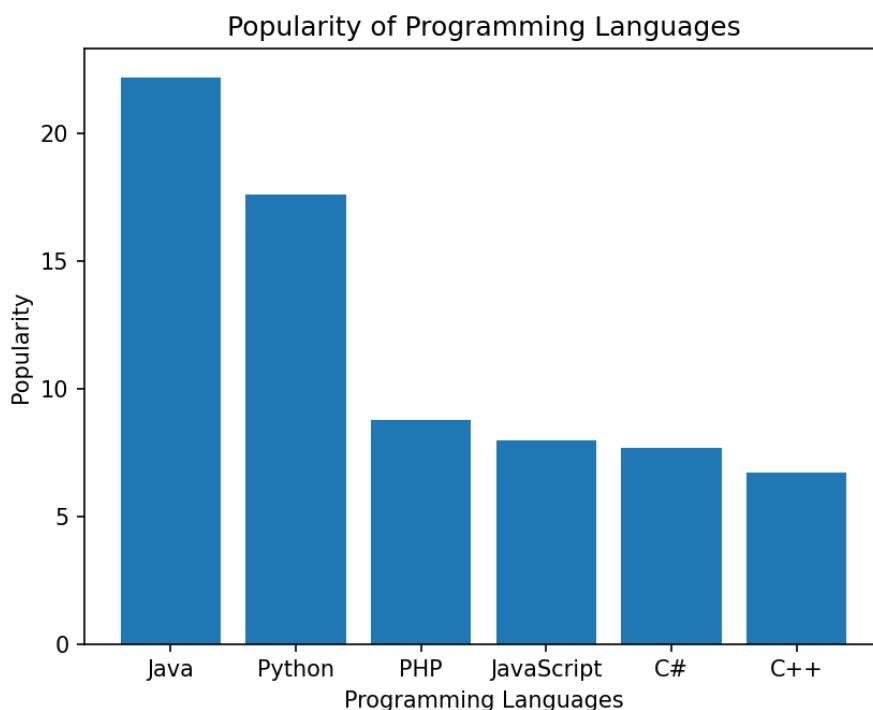
Programming languages: Java, Python, PHP, JavaScript, C#, C++

Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7

```
24 #Question 3
25 import matplotlib.pyplot as plt
26 languages = ["Java", "Python", "PHP", "JavaScript", "C#", "C++"]
27 popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
28 plt.bar(languages, popularity)
29 plt.xlabel("Programming Languages")
30 plt.ylabel("Popularity")
31 plt.title("Popularity of Programming Languages")
32 plt.show()
```

Figure 1

— □ ×





**Marwadi University**  
**Faculty of Engineering & Technology**  
**Department of Information and Communication Technology**

**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on matplotlib

**Experiment No: 10**

**Date:**

**Enrollment No: 92400133055**

More Practice

Reference : <https://medium.com/@DebaprasannBhoi/python-matplotlib-exercises-cc46c994563c>