Experiment No. 10
Program to plot graph using matplotlib library
Date of Performance: 01/04/2024
Date of Submission: 08/04/2024

Experiment No. 10

Title: Program to plot graph using matplotlib library

Aim: To explore the basics Matplotlib for data visualization.

Objective: To understand how to use graphs and charts for data analysis.

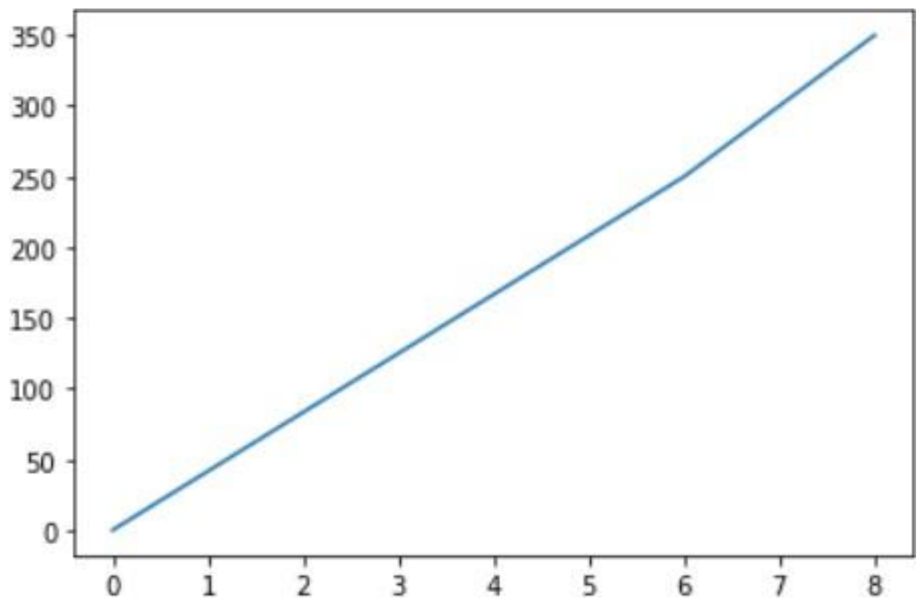
Theory:

Matplotlib is a low level graph plotting library in python that serves as a visualization utility. Matplotlib is open source and we can use it freely.

Most of the Matplotlib utilities lies under the pyplot submodule, and are usually imported under the plt alias.

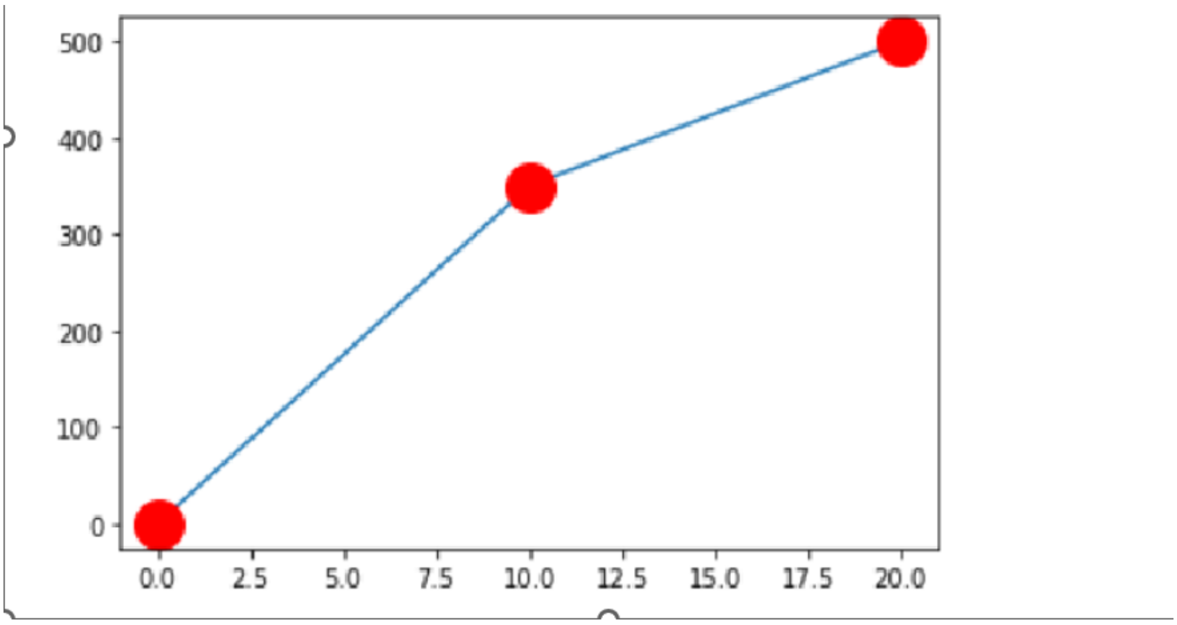
- The plot() function is used to draw points (markers) in a diagram.
- By default, the plot() function draws a line from point to point.
- The function takes parameters for specifying points in the diagram.
- Parameter 1 is an array containing the points on the x-axis.
- Parameter 2 is an array containing the points on the y-axis.Eg: (0,0), (6,250), (8,350) import matplotlib.pyplot as plt

```
import numpy as np


x = np.array([0,6,8])

y = np.array([0,250,350])


plt.plot(x,y) plt.show()
```



- The keyword argument marker is to emphasize each point with a specified marker.
- The keyword argument markersize or the shorter version, ms is to set the size of the markers
- The keyword argument markeredgecolor or the shorter mec is to set the color of the edge of the markers
- The keyword argument markerfacecolor or the shorter mfc is to set the color inside the edge of the markers
-

```
import matplotlib.pyplot as plt import numpy as np

x=np.array([0,10,20])

y=np.array([0,350,500]) plt.plot(x,y,marker='o',ms=20,mec='r',mfc='r') plt.show()
```



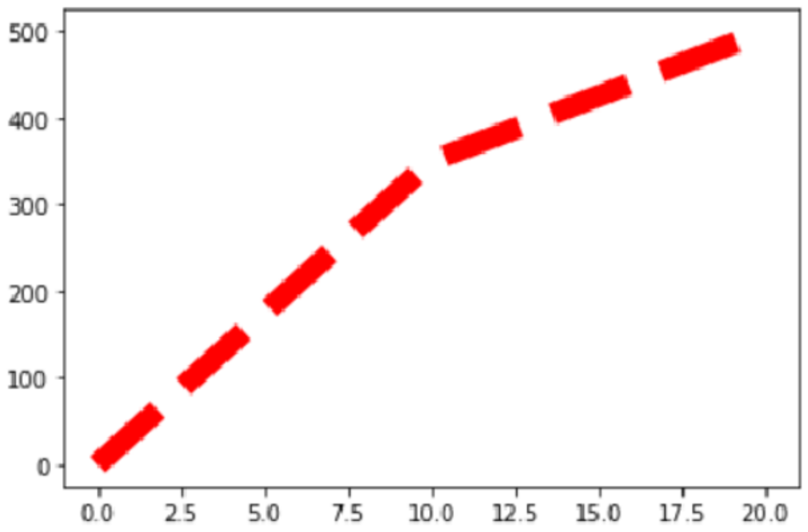
- The keyword argument linestyle, or shorter ls, to change the style of the plotted line.
- The line style can be written in a shorter syntax:
  - linestyle can be written as ls.
  - dotted can be written as :.
  - dashed can be written as --.
- the keyword argument color or the shorter c to set the color of the line

import matplotlib.pyplot as plt import numpy as np

x=np.array([0,10,20])

y=np.array([0,350,500]) plt.plot(x,y,color='red',ls='--',lw=10)

```
Out[16]: [<matplotlib.lines.Line2D at 0x24587eee970>]
```



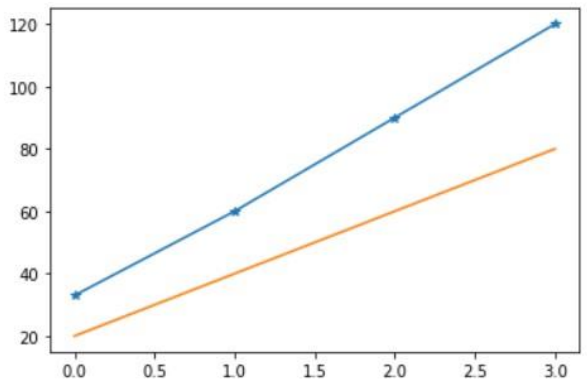
Many plotting can be done by adding more plt.plot() functions

import matplotlib.pyplot as plt import numpy as np

y1=np.array([33,60,90,120])

y2=np.array([20,40,60,80]) plt.plot(y1,marker='*') plt.plot(y2)

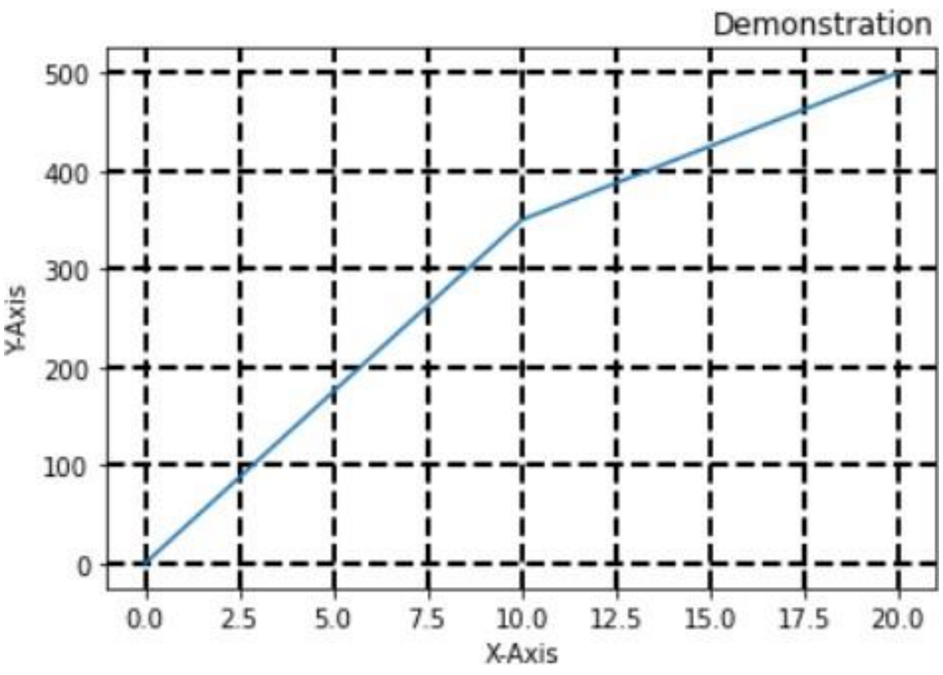
```
Out[22]: [<matplotlib.lines.Line2D at 0x2458853efa0>]
```



- WithPyplot,youcanusethe xlabel()andylabel()functionstosetalabelforthex-andy-axis.
- WithPyplot,youcanusethetitle()functiontosetatitle forthe plot.
- Youcanusethelocparameterintitle()topositionthe title.
- Legalvaluesare:'left', 'right',and'center'.Default valueis'center'.

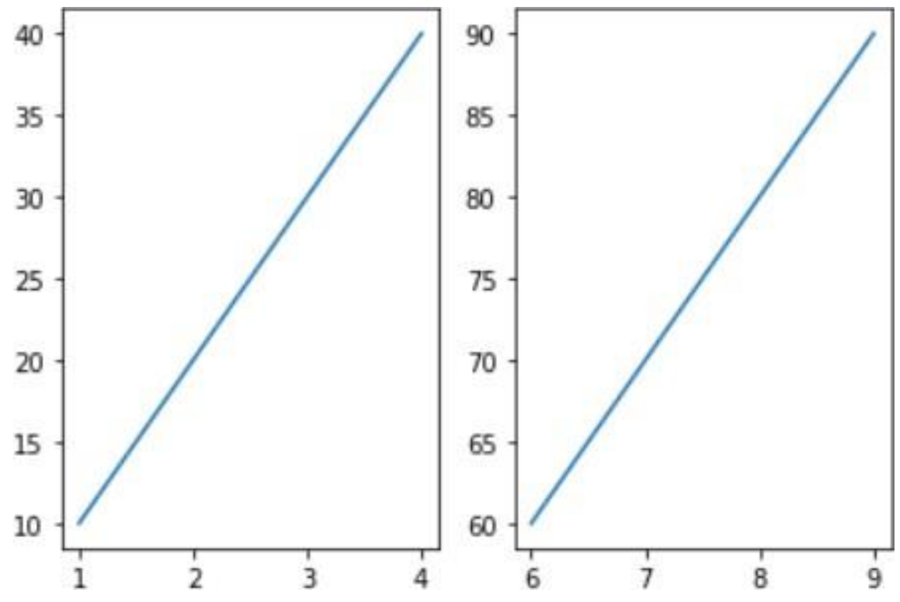
- WithPyplot,youcanusethegrid()functiontoaddgridlinesto theplot.
- Youcanuse the axisparameterinthegrid()functiontospecifywhichgridlinestodisplay.
- Legalvalues are:'x','y', and'both'.Defaultvalue is 'both'.

```
import
matplotlib.pyplot as
pltimportnumpy as np

x=np.array([0,10,20])
y=np.array([0,
350,500])plt.pl
ot(x,y)plt.xlabe
l("X-
Axis")plt.ylabe
l("Y-Axis")
plt.title('Demonstration',loc='right')p
lt.grid(color='black',linestyle='--
',linewidth=2)
```



SubPlots:
Withthesubplots()functionyoucandrawmultipleplots inonefigure.
Thesubplots()functiontakesthreeargumentsthatdescribesthelayoutofthefigure.
Thelayoutisorganizedinrowsandcolumns,whicharerepresentedbythefirstandsecondargume nt.Thethirdargument represents theindexofthe current plot.

```
x=np.array([1,2,3,4])
y=np.array([10,20,30,40])

plt.subpl
ot(1,2,1)
plt.plot(x
,y)


x=np.array([6,7,8,9])
y=np.array([60,70,80,90])

plt.subpl
ot(1,2,2)
plt.plot(x
,y)
```



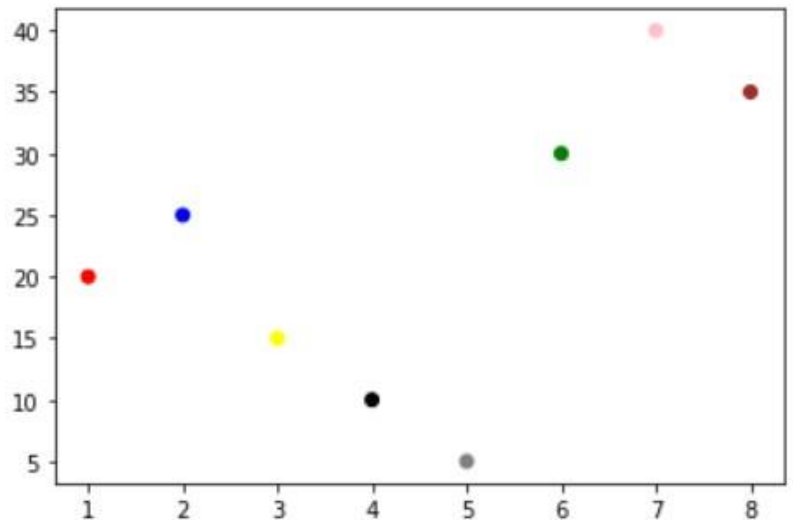
ScatterPlots:

- WithPyplot,youcanusethe scatter()functionto drawascatterplot.
- Thescatter()functionplotsonedotforeachobservation.Itneedstwoarraysofthesamel ength, oneforthevalues ofthex-axis, andoneforvaluesonthey-axis.
- Youcansetyourown colorforeach scatter plot with thecolororthe cargument.

```
import
matplotlib.pyplot as
```

```
pltimportnumpyas np

x=np.array([1,2,3,4,5,6,7,8])
y=np.array([20,25,15,10,5,30,40,35])
c=np.array(['red','blue','yellow','black','grey','green','pin
k','brown'])plt.scatter(x,y,color=c)
```

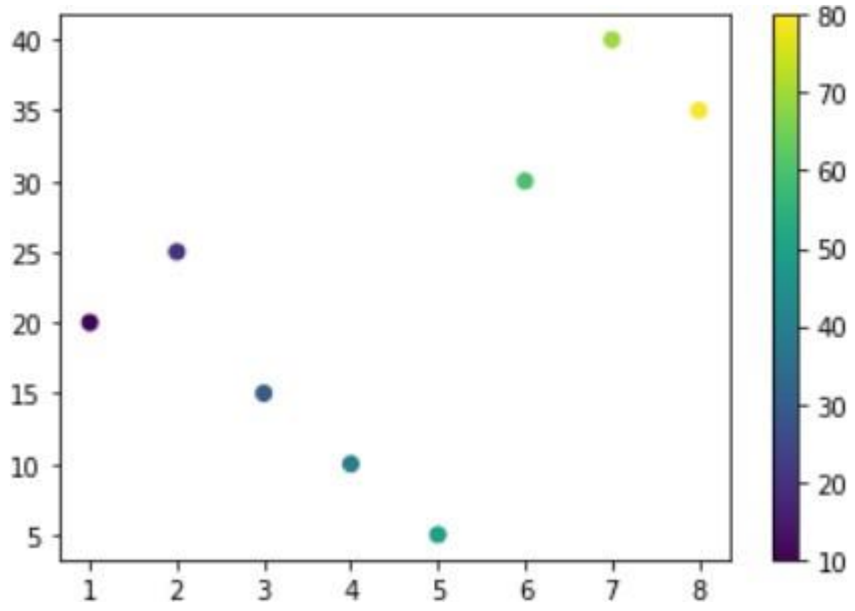
```
Out[53]: <matplotlib.collections.PathCollection at 0x24588442040>
```



ColorMaps

TheMatplotlibmodule hasa numberofavailablecolormaps.
Acolormapis like a listof colors,whereeachcolor hasa value thatranges from0to100.

```
import
matplotlib.pyplot as
pltimportnumpy as np



x=np.array([1,2,3,4,5,6,7,8])
y=np.array([20,25,15,10,5,30,40,35])
col=np.array([10,20,30,40,50,60,70,80])
plt.scatter(x,y,c=col,cma
p='viridis')plt.colorbar()
plt.show()
```



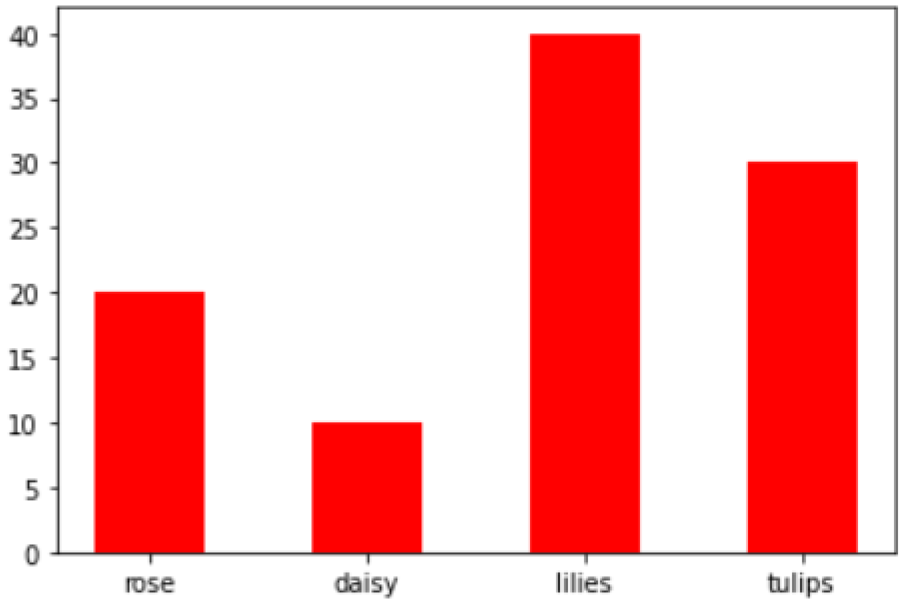
BarGraph

- WithPyplot,youcanusethe bar()functionto draw bargraphs.
- Thebar()functiontakes argumentsthatdescribesthelayoutofthebars.
- Thecategories andtheirvaluesrepresentedbythe firstandsecondargumentasarrays.
- Ifyouwantthebarstobedisplayedhorizontallyinsteadofvertically,usethebarh()function.
- Thebar()andbarh()takesthe keywordargumentcolortosetthecolorofthebars.

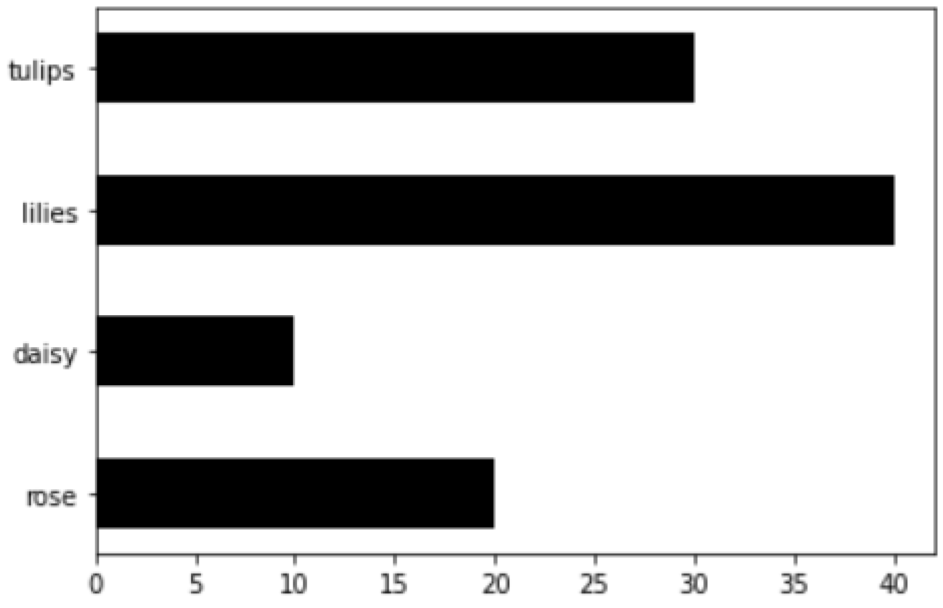
Thebar()takesthekeywordargumentwidthtosetthewidthofthe bars.

- Thebarh()takesthekeywordargumentheightto setthe heightofthebars.

```
import
matplotlib.pyplot as
pltimportnumpy as np
x=np.array(['rose','daisy','lilies','tulip
s'])
y=np.array([20,10,40,30])plt.bar(x,y
,color='red',width=0.5)
```



```
   import
   matplotlib.pyplot as
   pltimportnumpy as np
x=np.array(['rose','daisy','lilies','tulips'])
y=np.array([20,10,40,30])plt.barh(x,y,color='black',height=0.5)
```

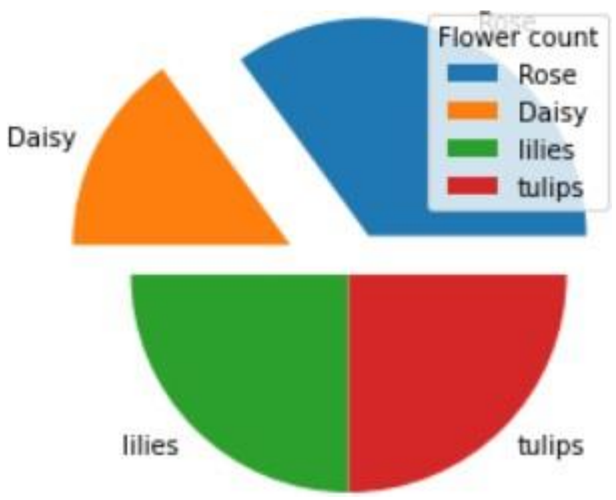


PieCharts

- WithPyplot,youcanusethepie()functiontodrawpiecharts.
- Thepiechartdrawsonepiece (called awedge) foreach value inthe array .
- By defaulttheplottingofthefirstwedgestartsfromthex-axisandmovecounterclockwise.
- Addlabelstothepie chartwith thelabel parameter.
- Thelabel parametermustbe anarraywithonelabelfor eachwedge.
- Thedefaultstartangleisatthex-axis,butyoucanchangethestartanglebyspecifyingastartangleparameter.
- Thestartangleparameterisdefinedwithanangleindegrees,defaultangleis0.
- Theexplodeparameterallowsyouto do that.
- Theexplodeparameter,ifspecified,andnotNone,mustbean arraywithone valueforeachwedge.
- Eachvalue representshowfarfrom thecentereachwedgeisdisplayed

```
import
matplotlib.pyplot as
pltimport numpy as
npy=np.array([35,15,
25,25])
l=np.array(['Rose','Daisy','lilies','tulips'])e=np.array([0.2,0.3,0,0])
```

```
plt.pie(y,labels=l,explo
de=e)plt.legend(title="
Flower
count")plt.show()
```

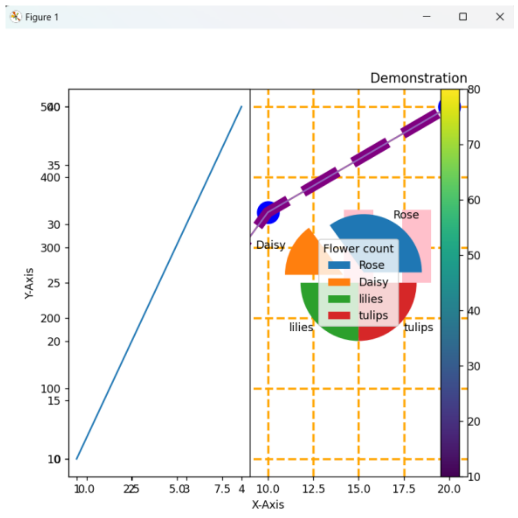


Code:

```
import matplotlib.pyplot as plt
import numpy as np

# 1. Line Plot
x1 = np.array([0, 6, 8])
y1 = np.array([0, 250, 350])
plt.plot(x1, y1)

# 2. Line Plot with Marker and Customization
x2 = np.array([0, 10, 20])
y2 = np.array([0, 350, 500])
plt.plot(x2, y2, marker='o', ms=20, mec='blue', mfc='b')

# 3. Line Plot with Color, Line Style, and Line Width
x3 = np.array([0, 10, 20])
y3 = np.array([0, 350, 500])
plt.plot(x3, y3, color='purple', ls='--', lw=10)

# 4. Multiple Line Plots
y4_1 = np.array([33, 90, 90, 120])
y4_2 = np.array([20, 70, 60, 80])
plt.plot(y4_1, marker='*')
plt.plot(y4_2)

# 5. Line Plot with Labels, Title, and Grid
x5 = np.array([0, 10, 20])
y5 = np.array([0, 350, 500])
plt.plot(x5, y5)
plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
plt.title('Demonstration', loc='right')
plt.grid(color='orange', linestyle='--', linewidth=2)

# 6. Subplots
x6_1 = np.array([1, 2, 3, 4])
y6_1 = np.array([10, 20, 30, 40])
plt.subplot(1, 2, 1)
plt.plot(x6_1, y6_1)
x6_2 = np.array([6, 7, 8, 9])
y6_2 = np.array([60, 70, 80, 90])
plt.subplot(1, 2, 2)
plt.plot(x6_2, y6_2)

# 7. Scatter Plot with Color
x7 = np.array([1, 2, 3, 4, 5, 6, 7, 8])
```

```
y7 = np.array([20, 25, 15, 10, 5, 30, 40, 35])
c7 = np.array(['red', 'blue', 'yellow', 'black', 'grey', 'green', 'pink', 'brown'])
plt.scatter(x7, y7, color=c7)

# 8. Scatter Plot with Color Map and Color Bar
x8 = np.array([1, 2, 3, 4, 5, 6, 7, 8])
y8 = np.array([20, 25, 15, 10, 5, 30, 40, 35])
col8 = np.array([10, 20, 30, 40, 50, 60, 70, 80])
plt.scatter(x8, y8, c=col8, cmap='viridis')
plt.colorbar()

# 9. Bar Plot
x9 = np.array(['rose', 'daisy', 'lilies', 'tulips'])
y9 = np.array([20, 10, 40, 30])
plt.bar(x9, y9, color='pink', width=0.5)

# 10. Pie Chart
y10 = np.array([35, 15, 25, 25])
l10 = np.array(['Rose', 'Daisy', 'lilies', 'tulips'])
e10 = np.array([0.2, 0.3, 0, 0])
plt.pie(y10, labels=l10, explode=e10)
plt.legend(title="Flower count")

# Show all plots
plt.show()
```

Output:



Conclusion:

After performing the experiment on data visualization using Matplotlib, it's evident that Matplotlib offers a versatile toolkit for creating a wide range of visualizations, including line plots, scatter plots, bar plots, and pie charts. By leveraging Matplotlib's functionalities, we can effectively explore and analyze data, gaining insights that aid decision-making processes. This experiment underscores the importance of data visualization in conveying information effectively and the utility of Matplotlib in achieving this goal.