Data vizualization

Data visualization refers to the techniques used to communicate data or information by encoding it as visual objects (e.g., points, lines or bars) contained in graphics.

Types of graph

Different types of graph can be depict different type of data. Most common graph types are as follows:

1. Line graph or regular plot

2. Scatter plot

3. Bar graph and histograms

4. Contour plot

5. Imshow

6. Quiver plot

7. Pie-chart

8. Grid

9. Multiplot/Subplot

10. Polar plot

11. 3D plot

11. 3D plot

12. Text plot

Regular plot / Line graph

Lines are made by connecting vertices with x,y coordinates. Their are two list of x and y coordinate sets that define a line graph. If their is only one list that corresponds to the y coordinates, the subsequent x coordinates are assumned as [0,...,n-1]. There can also be some extra parameters to line graph like color, vertex marker, label , linewidth etc.

The following color abbreviations are supported:

character	color
'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

The following markers abbreviations are supported:

character	description
'-'	solid line style

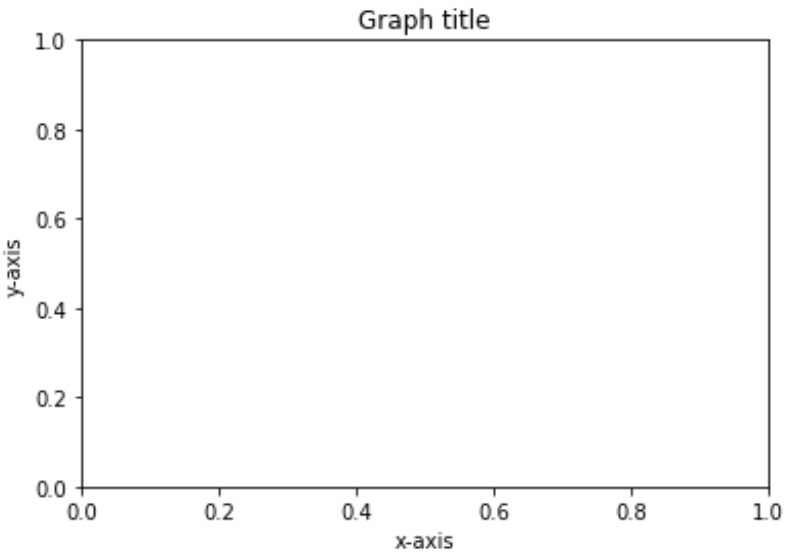
'--'	dashed line style
'-.'	dash-dot line style
':'	dotted line style
'.'	point marker
','	pixel marker
'o'	circle marker
'v'	triangle_down marker
'^'	triangle_up marker
'<'	triangle_left marker
'>'	triangle_right marker
'1'	tri_down marker
'2'	tri_up marker
'3'	tri_left marker
'4'	tri_right marker
's'	square marker
'p'	pentagon marker
'*'	star marker
'h'	hexagon1 marker
'H'	hexagon2 marker
'+'	plus marker
'x'	x marker
'D'	diamond marker
'd'	thin_diamond marker
'|'	vline marker
'_'	hline marker

Numpy array can also reprsent the list of coordinate. Numpy library offers many functions for drawing mathematical graphs.

```
import numpy as np

X = np.linspace(-np.pi, np.pi, 256,endpoint=True)
C,S = np.cos(X), np.sin(X)
```

In [57]:

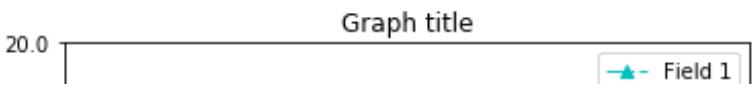
```
# A basic graph structure
# pip install matplotlib

import matplotlib.pyplot as plt
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.title("Graph title")

plt.show()
```

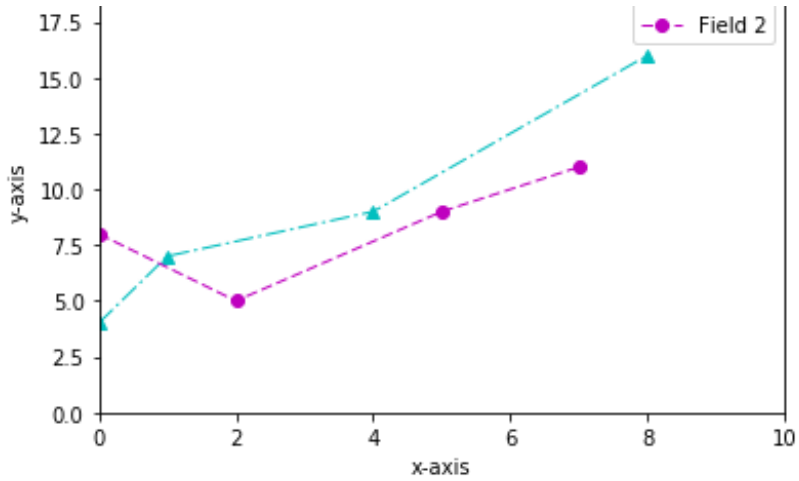


1. Line graph

In [94]:

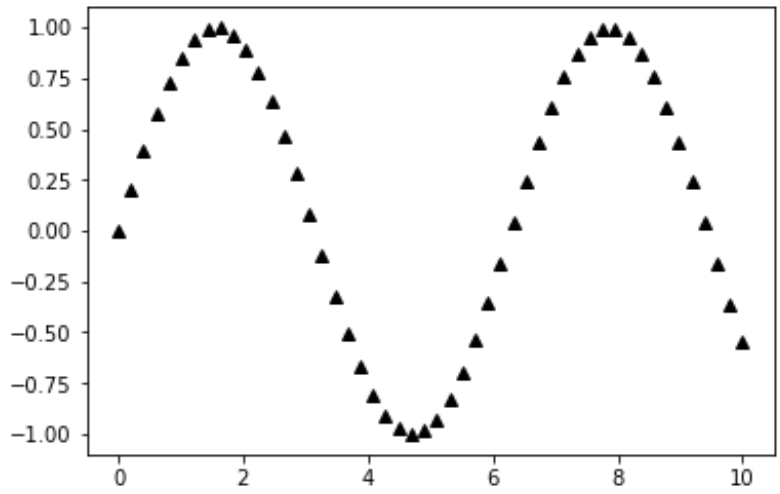
```
import matplotlib.pyplot as plt
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.title("Graph title")

plt.plot([0,1,4,8],[4,7,9,16],'c^-.',linewidth=1,label="Field 1")
plt.plot([0,2,5,7],[8,5,9,11],'mo--',linewidth=1,label="Field 2")

plt.axis([0,10,0,20])
plt.legend()
plt.show()
```





In [92]:

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

x = np.linspace(0, 10)
plt.plot(x, np.sin(x),'k^')

plt.show()
```

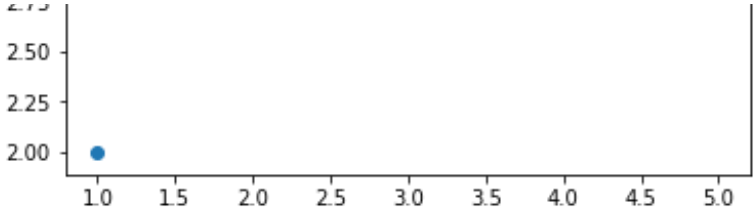


2. Scatter plot

It plots discrete x,y coordinates on the graph. It is like plotting several vertex with a marker.

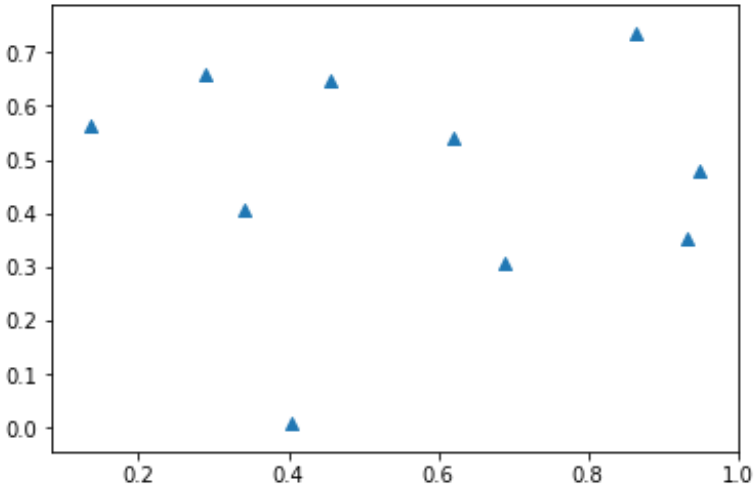
In [98]:

```
import matplotlib.pyplot as plt
plt.scatter([1,5,3],[2,3,4])
plt.show()
```



In [113]:

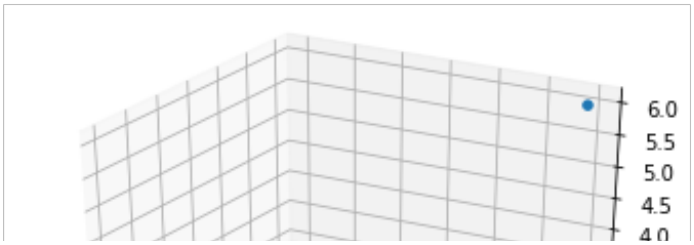
```
import numpy as np
import matplotlib.pyplot as plt


n = 10
x = np.random.rand(n)
y = np.random.rand(n)

plt.scatter(x,y,marker='^')
plt.show()
```

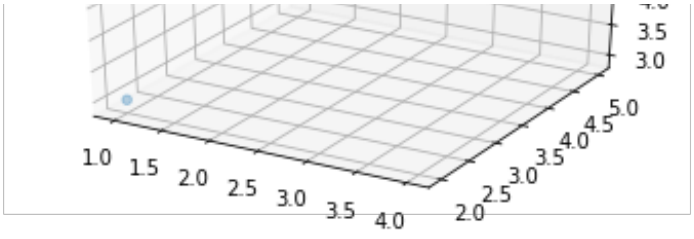


3D Scatter plot

In [153]:

```
import matplotlib.pyplot as plt

ax = plt.figure().add_subplot(111, projection='3d')

xs = [1, 4]
ys = [2, 5]
zs = [3, 6]

ax.scatter(xs, ys, zs)

plt.show()
```



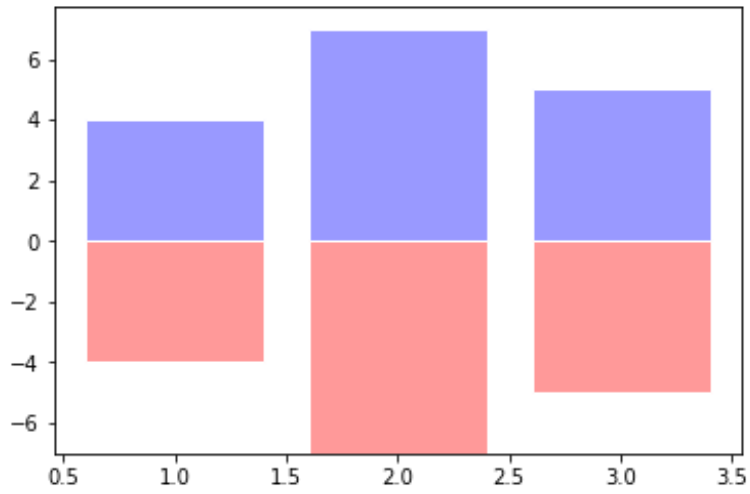


3a. Bar graph

It represents a list of value against a list of x coordinates.

In [160]:

```
import matplotlib.pyplot as plt

plt.bar([1,2,3],[4,7,5], facecolor='#9999ff', edgecolor='white')
plt.bar([1,2,3],[-4,-7,-5], facecolor='#ff9999', edgecolor='white')

plt.show()
```

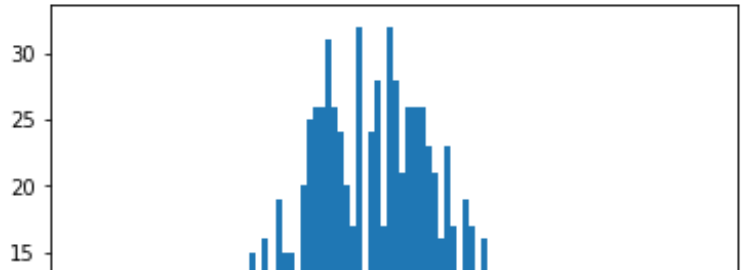


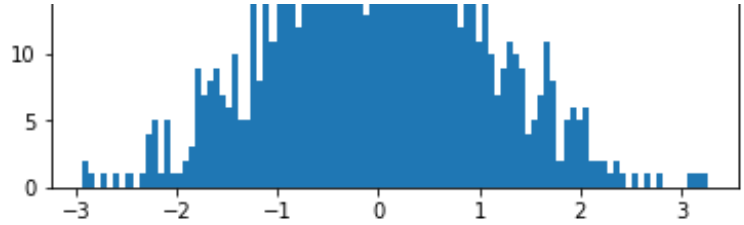
3b. Histogram

It represents list of value against a range or bins.

In [176]:

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

x = np.random.randn(1000)
plt.hist(x,100)
plt.show()
```

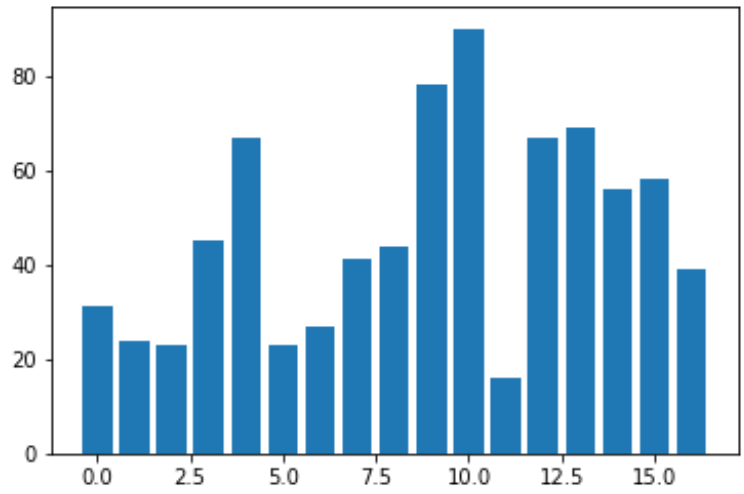




Bar graph vs histagram

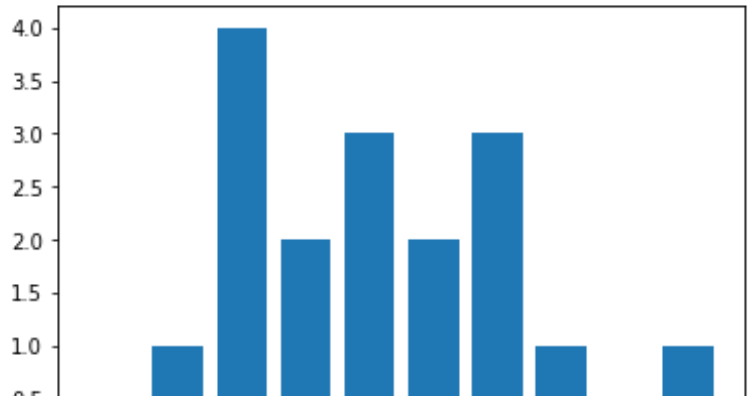
In [186]:

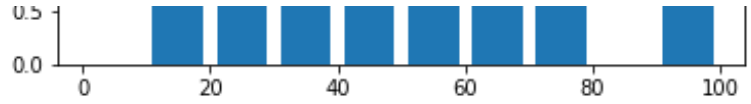
```
# bar graph
import matplotlib.pyplot as plt

population_ages = [31,24,23,45,67,23,27,41,44,78,90,16,67,69,56,58,39]
ids = [i for i in range(len(population_ages))]
plt.bar(ids,population_ages)

plt.show()
```



In [193]:

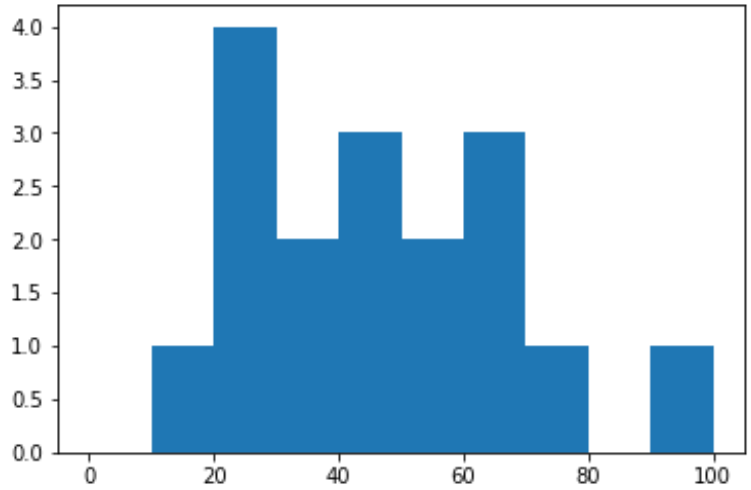
```
# histagram
import matplotlib.pyplot as plt

population_ages = [31,24,23,45,67,23,27,41,44,78,90,16,67,69,56,58,39]
bins = [0,10,20,30,40,50,60,70,80,90,100]
plt.hist(population_ages,bins,histtype="bar",rwidth=0.8)

plt.show()
```





In [194]:

```
import matplotlib.pyplot as plt

population_ages = [31,24,23,45,67,23,27,41,44,78,90,16,67,69,56,58,39]
bins = [0,10,20,30,40,50,60,70,80,90,100]
plt.hist(population_ages,bins,histtype="stepfilled",rwidth=0.8)

plt.show()
```

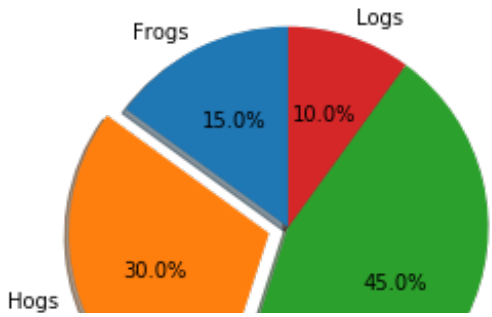


5. pie chart

Here the slices are ordered and plotted counter-clockwise.

In [195]:

```
import matplotlib.pyplot as plt

labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
sizes = [15, 30, 45, 10]
explode = (0, 0.1, 0, 0)  # only "explode" the 2nd slice (i.e. 'Hogs')

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circ
le.

plt.show()
```




Dogs