Data Visualization Using Matplotlib ¶

```
Data Visualization is the process of presenting data in the form of graphs or
charts. It helps to understand large and complex amounts of data very easily.
It allows the decision-makers to make decisions very efficiently and also
allows them in identifying new trends and patterns very easily. It is also used
in high-level data analysis for Machine Learning and Exploratory Data Analysis
(EDA).  Data visualization can be done with various tools like Tableau, Power
BI, Python.
```

```
Matploptib is a low-level library of Python which is used for data
visualization. It is easy to use and emulates MATLAB like graphs and
visualization. This library is built on the top of NumPy arrays and consist of
several plots like line chart, bar chart, histogram, etc. It provides a lot of
flexibility but at the cost of writing more code.
```

PYPLOT

```
Pyplot is a Matplotlib module that provides a MATLAB-like interface. Matplotlib
is designed to be as usable as MATLAB, with the ability to use Python and the
advantage of being free and open-source. Each pyplot function makes some change
to a figure: e.g., creates a figure, creates a plotting area in a figure, plots
some lines in a plotting area, decorates the plot with labels, etc. The various
plots we can utilize using Pyplot are Line Plot, Histogram, Scatter, 3D Plot,
Image, Contour, and Polar.
```

```
# Example:

    import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

plt.show()


title()

The title() method in matplotlib module is used to specify the title of the
visualization depicted and displays the title using various attributes.

Example:

import matplotlib.pyplot as plt
```

```
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

# Adding title to the plot
plt.title("Linear graph")

plt.show()
```

Adding X Label and Y Label

In layman’s terms, the X label and the Y label are the titles given to X-axis and Y-axis respectively. These can be added to the graph by using the xlabel() and ylabel() methods.

Example:

```
import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="green")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

# Adding label on the x-axis
plt.xlabel('X-Axis')

plt.show()
```

Adding Legends

A legend is an area describing the elements of the graph. In simple terms, it reflects the data displayed in the graph’s Y-axis. It generally appears as the box containing a small sample of each color on the graph and a small description of what this data means.

The attribute bbox_to_anchor=(x, y) of legend() function is used to specify the coordinates of the legend, and the attribute ncol represents the number of columns that the legend has. Its default value is 1.

```
Example:

import matplotlib.pyplot as plt


# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="green")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

# Adding label on the x-axis
plt.xlabel('X-Axis')

# Setting the limit of y-axis
plt.ylim(0, 80)

# setting the labels of x-axis
plt.xticks(x, labels=["one", "two", "three", "four"])

# Adding legends
plt.legend(["GFG"])

plt.show()
```

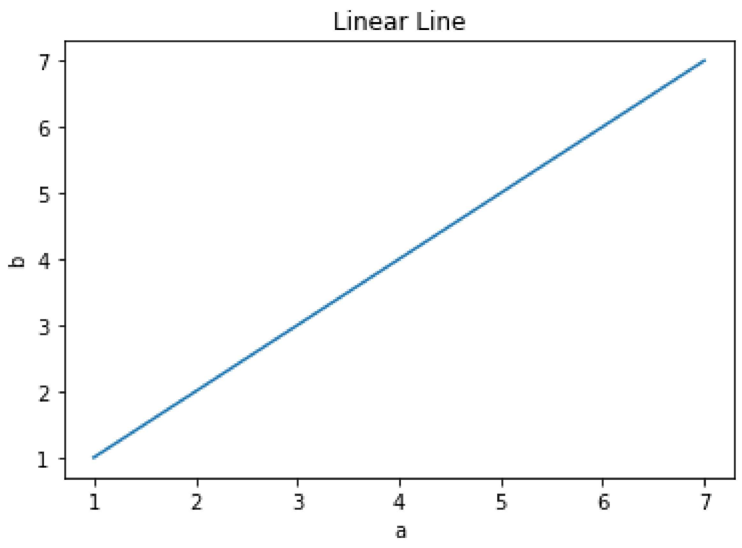
```
Just like pyplot class, axes class also provides methods for adding titles,
legends, limits, labels, etc. Let’s see a few of them –

Adding Title – ax.set_title()
Adding X Label and Y label – ax.set_xlabel(), ax.set_ylabel()
Setting Limits – ax.set_xlim(), ax.set_ylim()
Tick labels – ax.set_xticklabels(), ax.set_yticklabels()
Adding Legends – ax.legend()


Example:

# Python program to show pyplot module

import matplotlib.pyplot as plt
from matplotlib.figure import Figure

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

fig = plt.figure(figsize = (5, 4))

# Adding the axes to the figure
```

```
ax = fig.add_axes([1, 1, 1, 1])

# plotting 1st dataset to the figure
ax1 = ax.plot(x, y)

# plotting 2nd dataset to the figure
ax2 = ax.plot(y, x)

# Setting Title
ax.set_title("Linear Graph")

# Setting Label
ax.set_xlabel("X-Axis")
ax.set_ylabel("Y-Axis")

# Adding Legend
ax.legend(labels = ('line 1', 'line 2'))

plt.show()
```

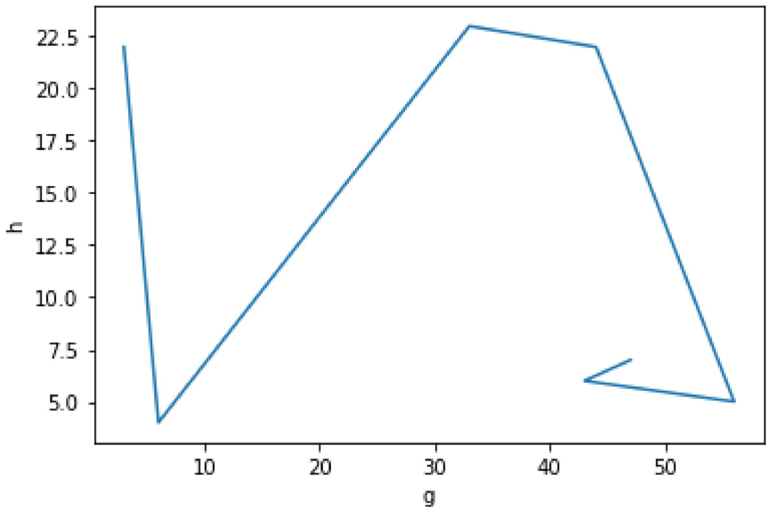
In [2]:
```
# From matplotlib import pyplot as plt

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

In [3]:
```
a=np.array([1,2,3,4,5,6,7])
b=np.array([1,2,3,4,5,6,7])
```

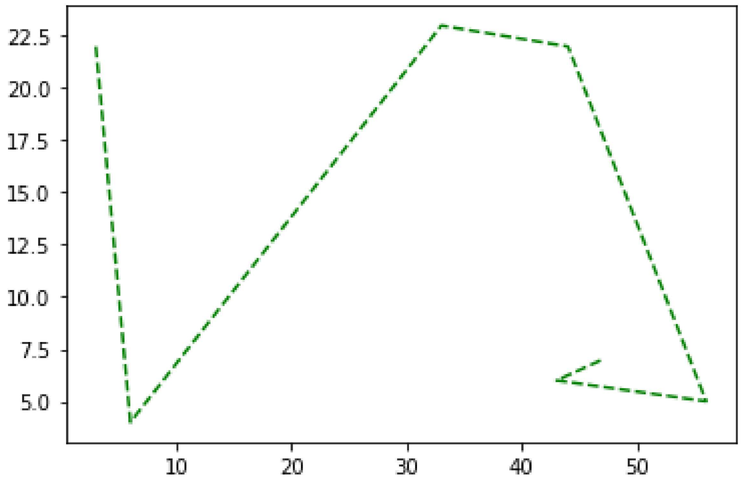
In [4]:
```
plt.title('Linear Line')
plt.xlabel("a")
plt.ylabel("b")
plt.plot(a,b)
plt.show()
```



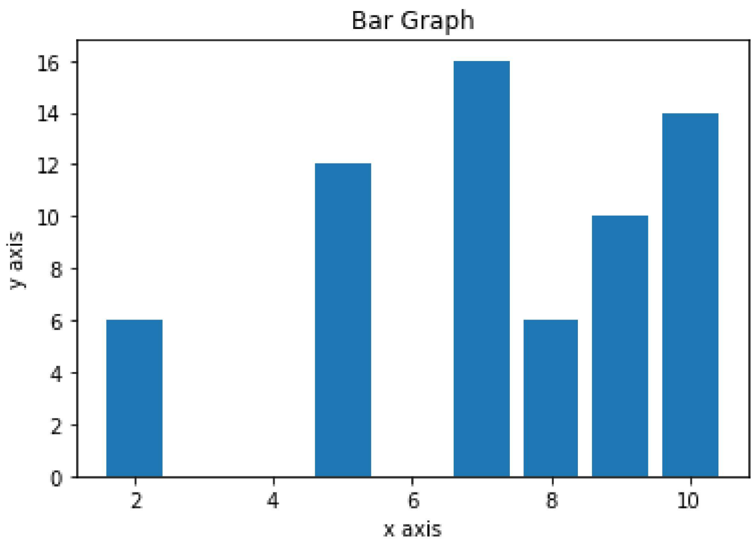
In [5]:
```
g=np.array([3,6,33,44,56,43,47])
h=np.array([22,4,23,22,5,6,7])
plt.xlabel('g')
plt.ylabel('h')
plt.plot(g,h)
plt.show()
```



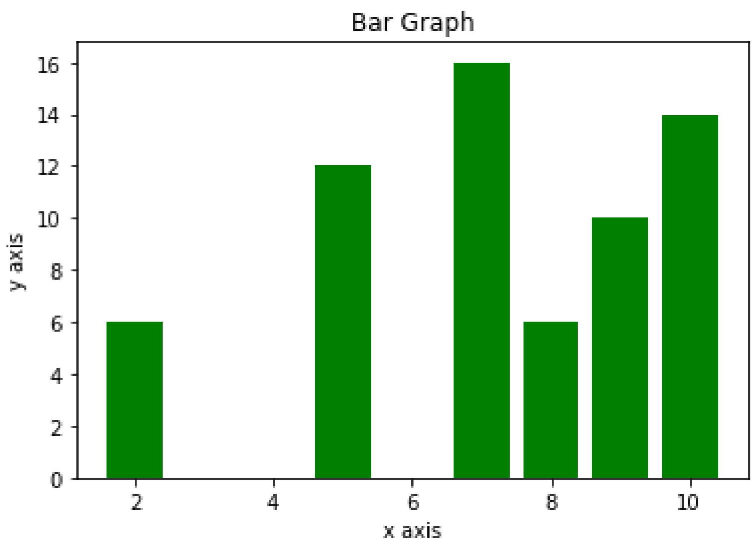
In [8]:
```
plt.plot(g,h,'--g')
plt.show()
```



In [9]:
```
# Bar Plot

x=[5,7,8,9,10,2]
y=[12,16,6,10,14,6]
plt.bar(x,y)
plt.title('Bar Graph')
plt.ylabel('y axis')
plt.xlabel('x axis')
plt.show()
```



In [10]:
```
x=[5,7,8,9,10,2]
y=[12,16,6,10,14,6]
plt.bar(x,y,color='g',align='center')
plt.title('Bar Graph')
plt.ylabel('y axis')
plt.xlabel('x axis')
plt.show()
```



Histogram

A histogram is basically used to represent data provided in a form of some groups.It is accurate method for the graphical representation of numerical data distribution.It is a type of bar plot where X-axis represents the bin ranges while Y-axis gives information about frequency.

Creating a Histogram

To create a histogram the first step is to create 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.The matplotlib.pyplot.hist() function is used to compute and create histogram of x.

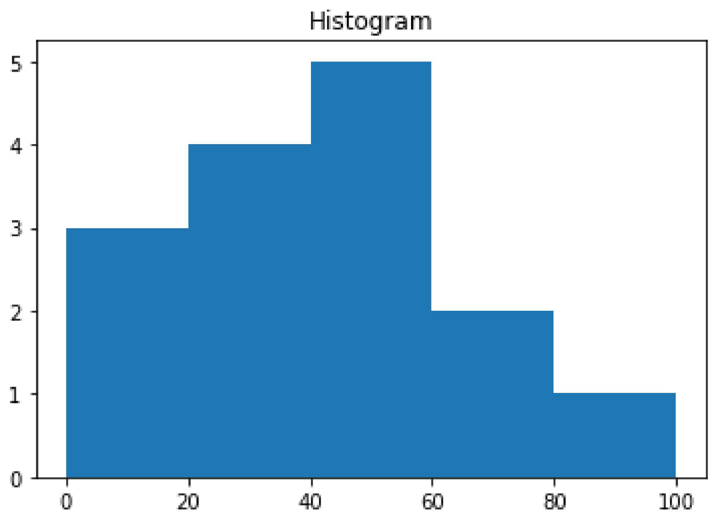
The following table shows the parameters accepted by matplotlib.pyplot.hist() function :

Attribute	parameter
X:	array or sequence of array
bins:	optional parameter contains integer or sequence or strings
density:	optional parameter contains boolean values
range:	optional parameter represents upper and lower range of bins
histtype:	optional parameter used to create type of histogram [bar, barstacked, step, stepfilled], default is “bar”
align:	optional parameter controls the plotting of histogram [left, right, mid]

```
weights:    optional parameter contains array of weights having same dimensions
as x
bottom:     location of the basline of each bin
rwidth:     optional parameter which is relative width of the bars with respect
to bin width
color:      optional parameter used to set color or sequence of color specs
label:      optional parameter string or sequence of string to match with
multiple datasets
log:        optional parameter used to set histogram axis on log scale


Example:

from matplotlib import pyplot as plt
import numpy as np


# Creating dataset
a = np.array([22, 87, 5, 43, 56,
              73, 55, 54, 11,
              20, 51, 5, 79, 31,
              27])

# Creating histogram
fig, ax = plt.subplots(figsize =(10, 7))
ax.hist(a, bins = [0, 25, 50, 75, 100])

# Show plot
plt.show()
```

In [11]:
```python
import matplotlib.pyplot as plt
import numpy as np
```

In [12]:
```
a=np.array([22, 87, 5, 43, 56,
              73, 55, 54, 11,
              20, 51, 5, 79, 31,
              27])
plt.hist(a, bins=[0, 20, 40, 60, 80, 100])
plt.title("Histogram")
plt.show()
```



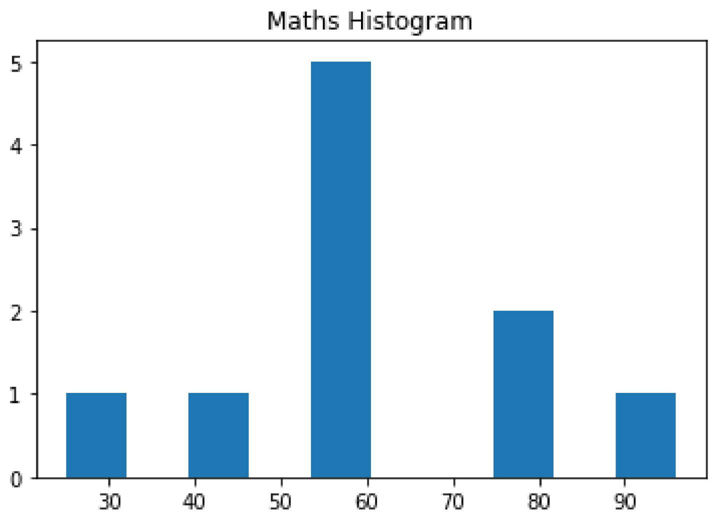
In [13]:
```
import pandas as pd
```

In [14]:
```
df=pd.read_csv('student_marks.csv')
print(df)
df.columns
```

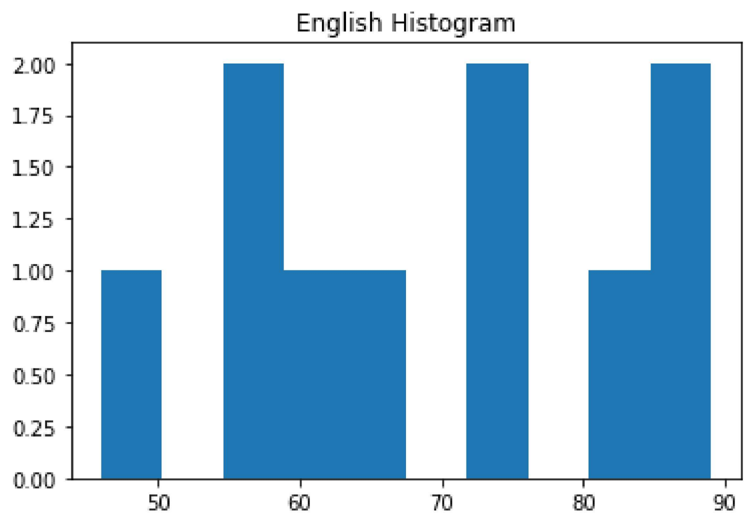
```
  Unnamed: 0 Gender         DOB  Maths  Physics  Chemistry  English  Biology  \
0       John      M  05-04-1988     55       45         56       87       21
1     Suresh      M  04-05-1987     75       96         78       64       90
2     Ramesh      M  25-05-1989     25       54         89       76       95
3    Jessica      F  12-08-1990     78       96         86       63       54
4   Jennifer      F  02-09-1989     58       96         78       46       96
5       Annu      F  05-04-1988     45       87         52       89       55
6      pooja      F  04-05-1987     55       64         61       58       75
7     Ritesh      M  25-05-1989     54       76         87       56       25
8      Farha      F  12-08-1990     55       63         89       75       78
9     Mukesh      M  02-09-1989     96       46         77       83       58

   Economics  History  Civics
0         52       89      65
1         61       58       2
2         87       56      74
3         89       75      45
4         77       83      53
5         89       87      52
6         58       64      61
7         56       76      87
8         75       63      89
9         83       46      77
```

Out[14]:
```
Index(['Unnamed: 0', 'Gender', 'DOB', 'Maths', 'Physics', 'Chemistry',
       'English', 'Biology', 'Economics', 'History', 'Civics'],
      dtype='object')
```

In [15]:
```
plt.hist(df['Maths'], bins=10)
plt.title("Maths Histogram")
plt.show()
```

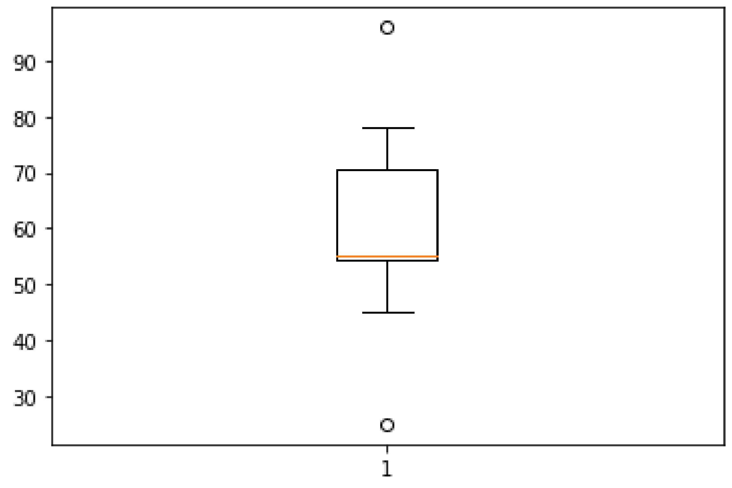


In [17]:
```
plt.hist(df['English'], bins=10)
plt.title("English Histogram")
plt.show()
```



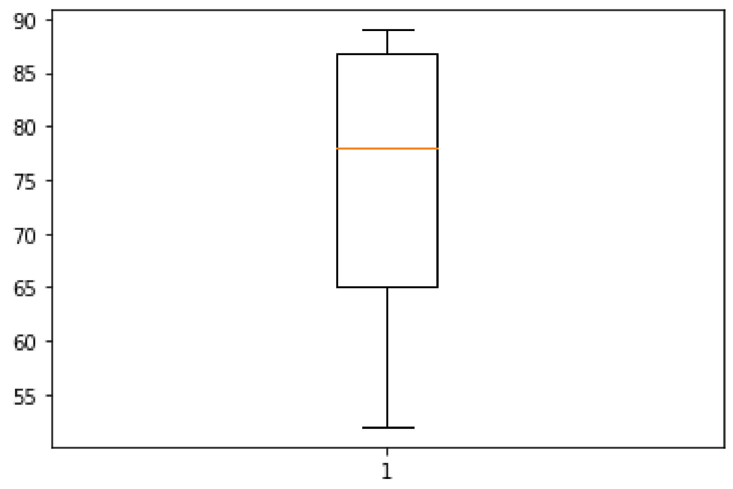
In [18]:
```
# Box Plot

plt.boxplot(df['Maths'])
```

Out[18]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x20e96c720a0>,
  <matplotlib.lines.Line2D at 0x20e96c72430>],
 'caps': [<matplotlib.lines.Line2D at 0x20e96c727c0>,
  <matplotlib.lines.Line2D at 0x20e96c72b50>],
 'boxes': [<matplotlib.lines.Line2D at 0x20e96c63cd0>],
 'medians': [<matplotlib.lines.Line2D at 0x20e96c72ee0>],
 'fliers': [<matplotlib.lines.Line2D at 0x20e96c802b0>],
 'means': []}
```



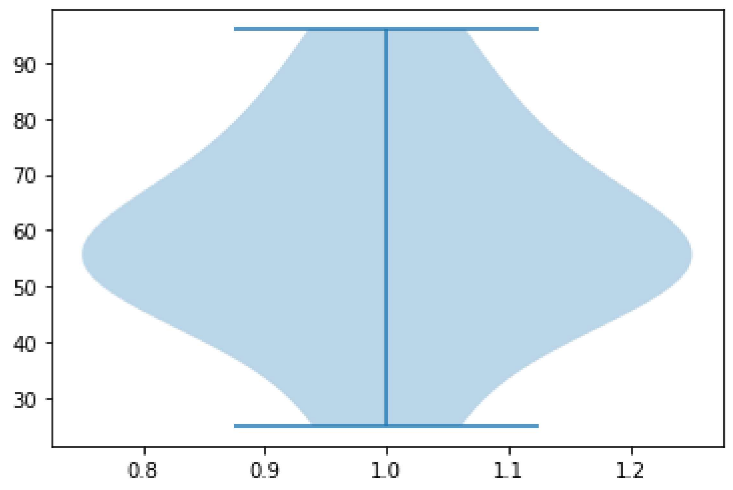
In [19]:
```
plt.boxplot(df['Chemistry'])
```

Out[19]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x20e96c0fe50>,
  <matplotlib.lines.Line2D at 0x20e96c15310>],
 'caps': [<matplotlib.lines.Line2D at 0x20e96c15df0>,
  <matplotlib.lines.Line2D at 0x20e96c15130>],
 'boxes': [<matplotlib.lines.Line2D at 0x20e96c0f940>],
 'medians': [<matplotlib.lines.Line2D at 0x20e96c15400>],
 'fliers': [<matplotlib.lines.Line2D at 0x20e96be74f0>],
 'means': []}
```



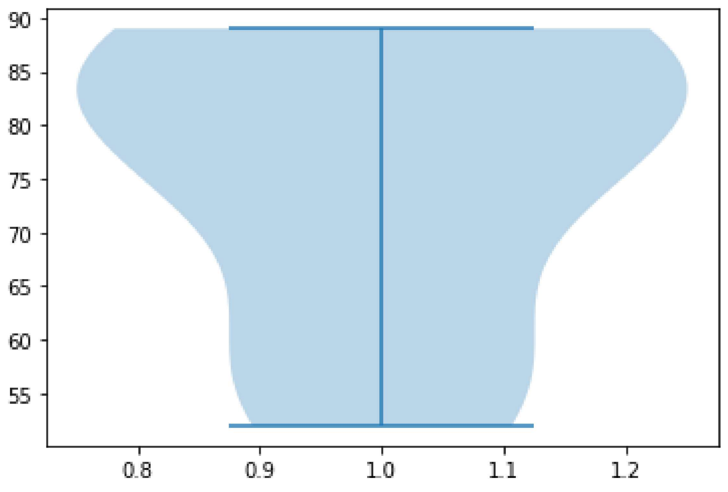
In [20]:
```
# Violine Plot

plt.violinplot(df['Maths'])
```

Out[20]:
```
{'bodies': [<matplotlib.collections.PolyCollection at 0x20e95b6e460>],
 'cmaxes': <matplotlib.collections.LineCollection at 0x20e95b6e3a0>,
 'cmins': <matplotlib.collections.LineCollection at 0x20e95b5d3a0>,
 'cbars': <matplotlib.collections.LineCollection at 0x20e95b5d970>}
```

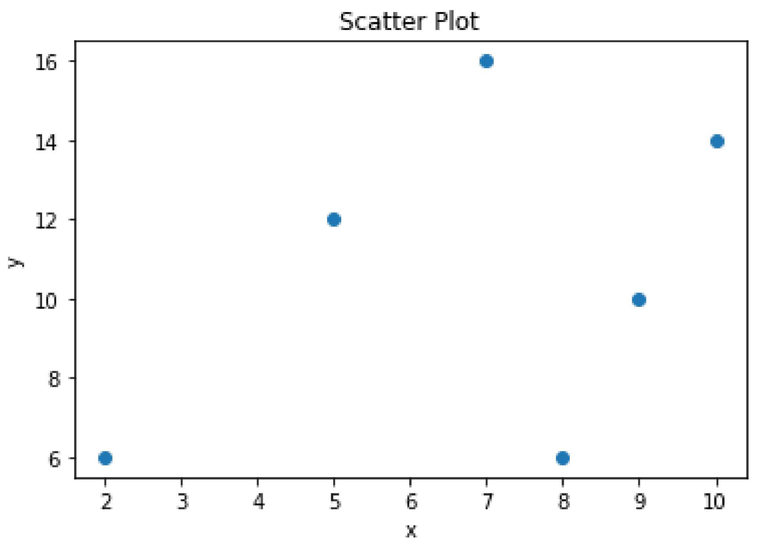


In [21]:
```
plt.violinplot(df['Chemistry'])
```

Out[21]:
```
{'bodies': [<matplotlib.collections.PolyCollection at 0x20e93facd90>],
 'cmaxes': <matplotlib.collections.LineCollection at 0x20e93e41940>,
 'cmins': <matplotlib.collections.LineCollection at 0x20e93e43fd0>,
 'cbars': <matplotlib.collections.LineCollection at 0x20e93e43130>}
```

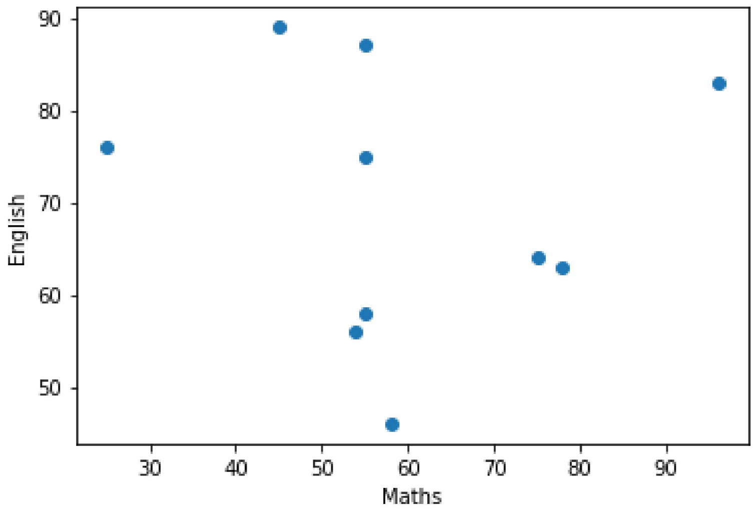


In [22]:
```
# Scatter Plot

x=[5,7,8,9,10,2]
y=[12,16,6,10,14,6]
plt.scatter(x,y)
plt.title('Scatter Plot')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```



In [23]:
```
x=df['Maths']
y=df['English']
```

In [24]:
```
plt.scatter(x,y)
plt.xlabel('Maths')
plt.ylabel('English')
plt.show()
```



In [ ]: