In [2]:

```
import numpy as np
import pandas as pd
```

In [3]:

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

In [4]:

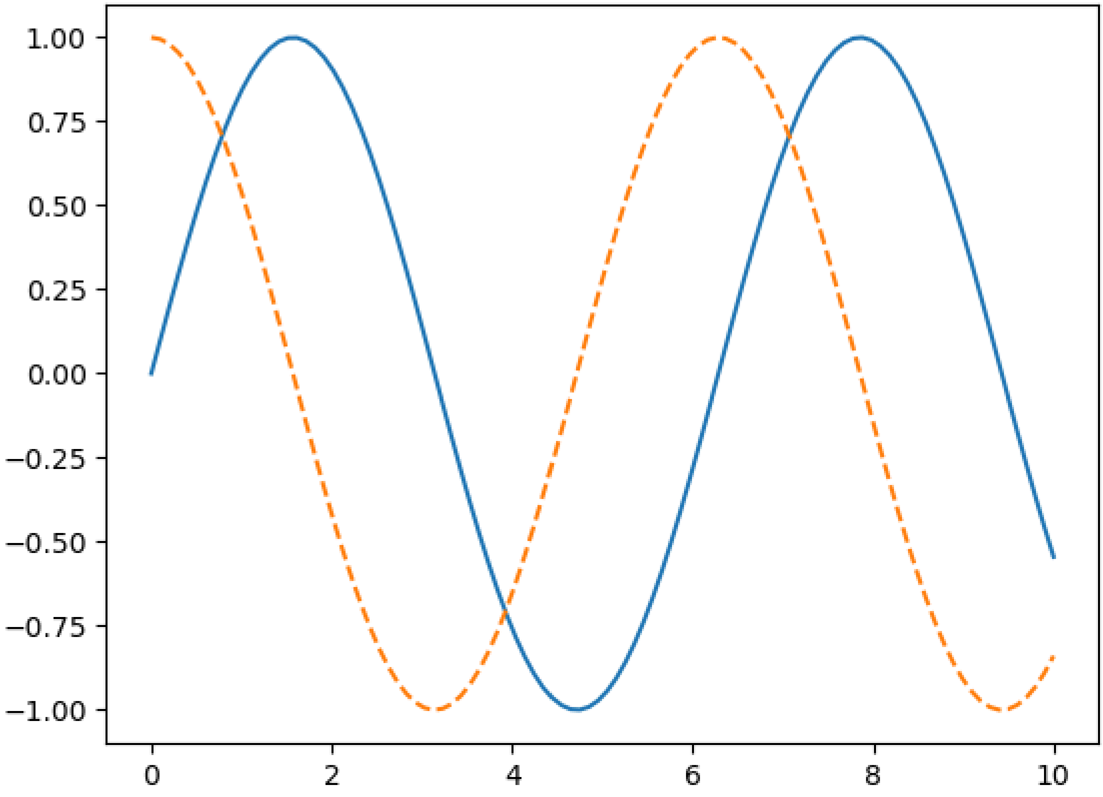
```
#Display plots in matplotlib
```

In [5]:

```
%matplotlib inline

x1 = np.linspace(0,10,100)

#create a plot figure
fig = plt.figure()

plt.plot(x1, np.sin(x1), '-')
plt.plot(x1, np.cos(x1), '--')
```

Out[5]:

```
[<matplotlib.lines.Line2D at 0x26bb7d77f20>]
```



In [6]:

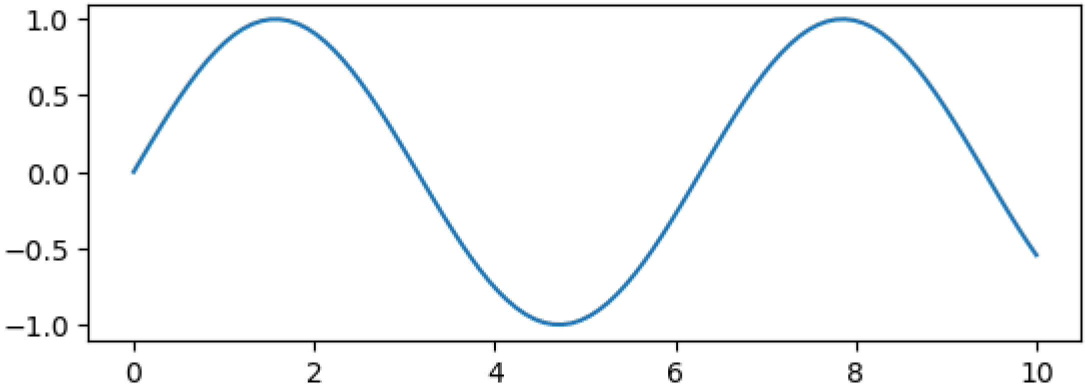
```
#Pyplot API
```

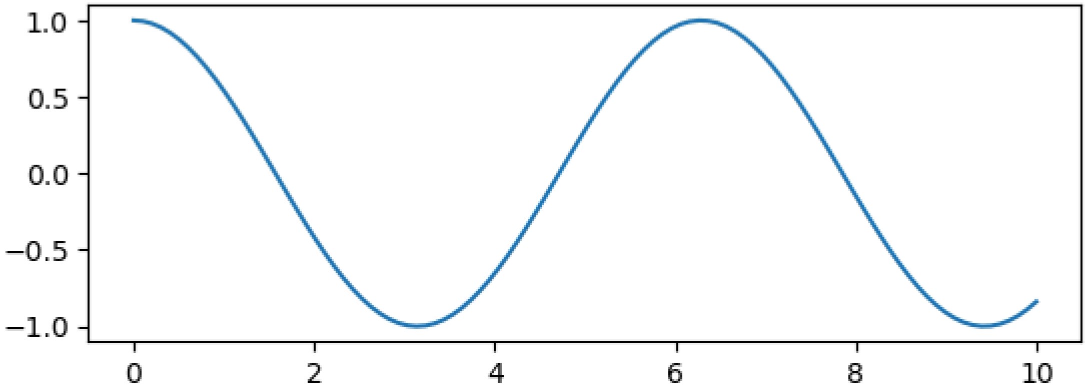
In [7]:

```
plt.figure()
plt.subplot(2,1,1)
plt.plot(x1, np.sin(x1))
plt.subplot(2,1,2)
plt.plot(x1,np.cos(x1))
```

Out[7]:

```
[<matplotlib.lines.Line2D at 0x26bb7e9aea0>]
```



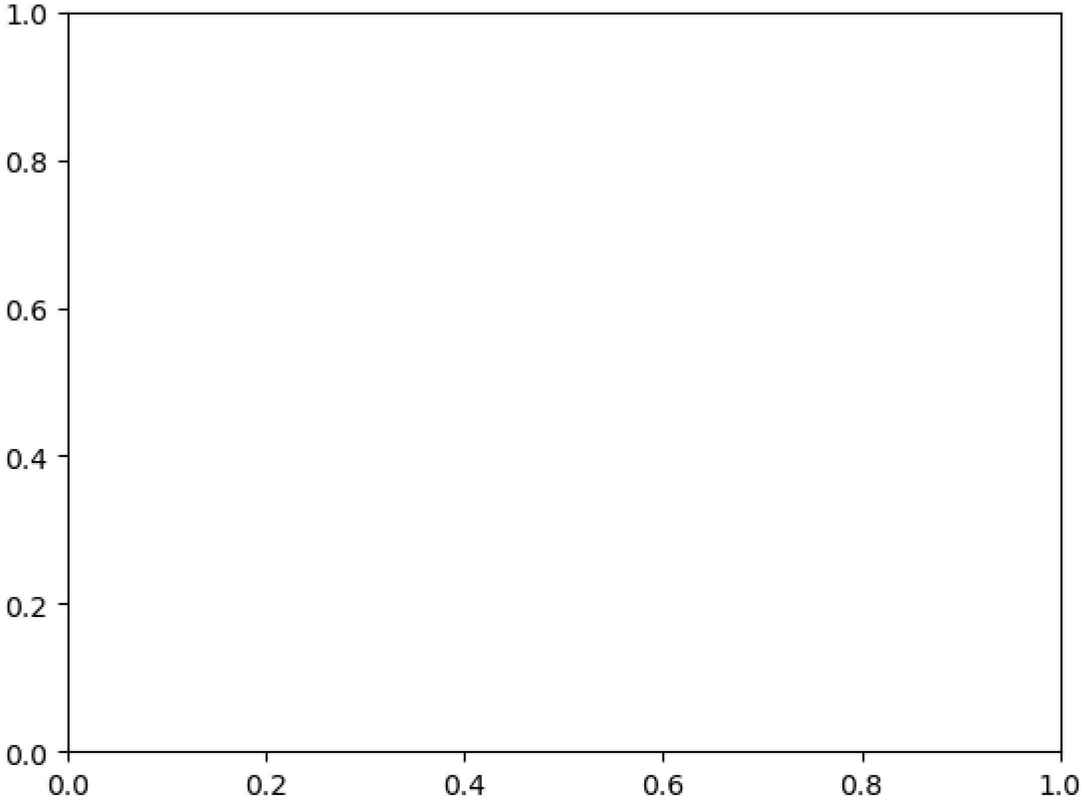


In [8]:
```
print(plt.gcf()) #gcf-Get Current Figure
```

```
Figure(640x480)
<Figure size 640x480 with 0 Axes>
```

In [9]:
```
print(plt.gca()) #gca-Get Current Axes
```

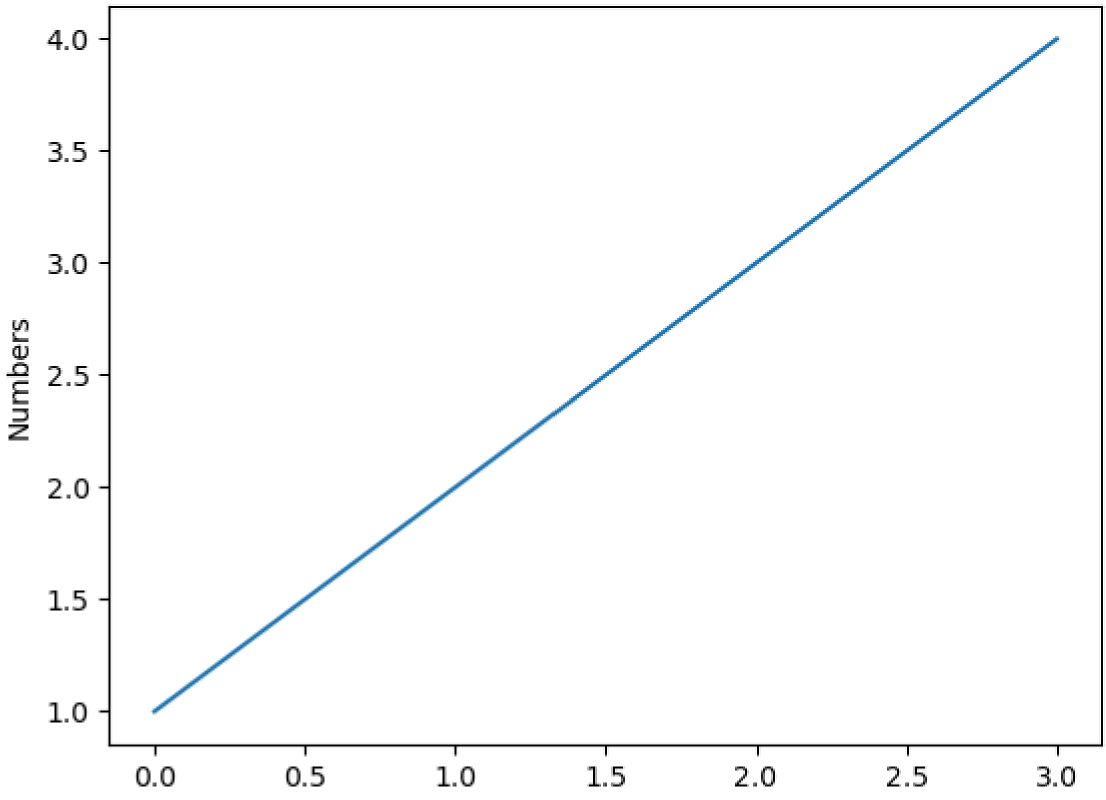
```
Axes(0.125,0.11;0.775x0.77)
```



In [10]:
```
#Visualization With Pyplot
```

In [11]:

```
plt.plot([1,2,3,4])
plt.ylabel('Numbers')
plt.show()
```

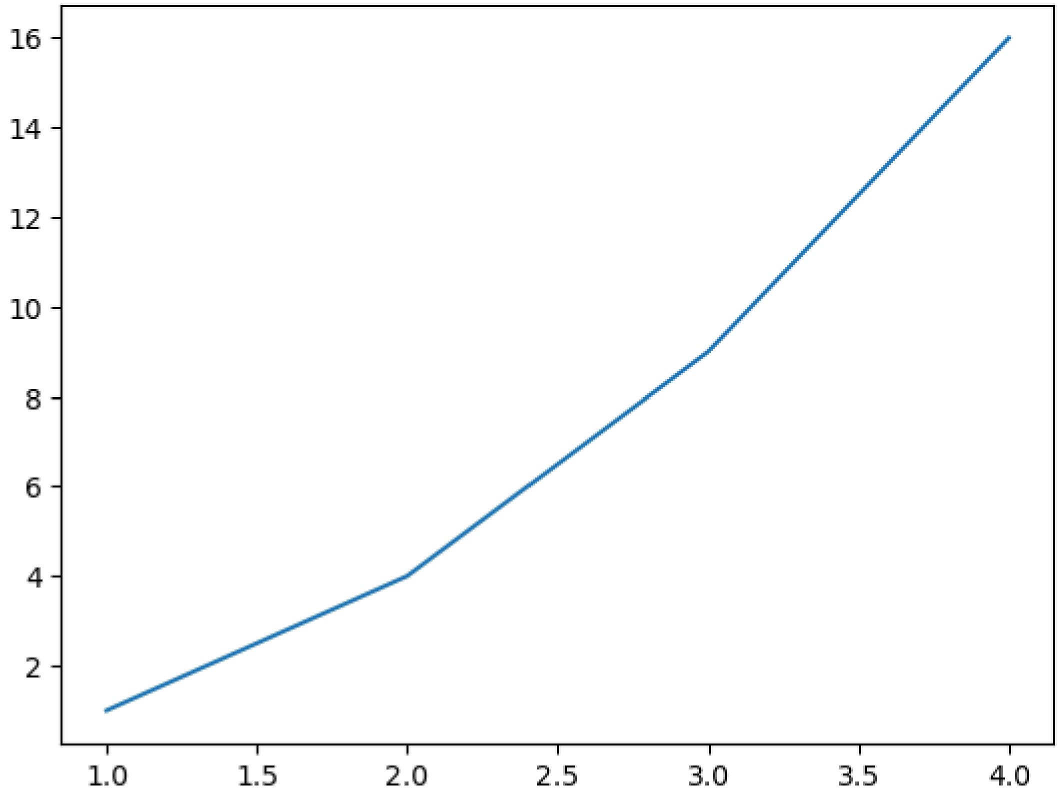


In [12]:

```
#plot()- this function is used to create 2d line plots.
```

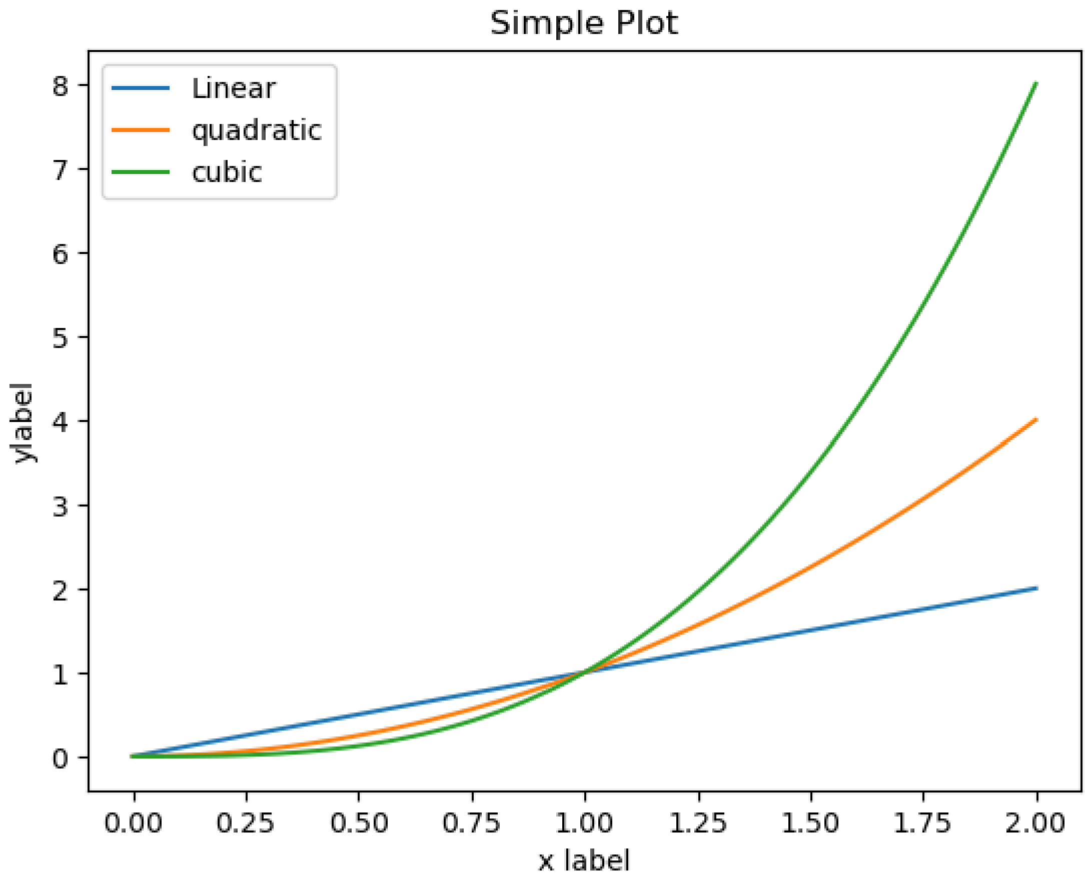
In [13]:

```
plt.plot([1,2,3,4],[1,4,9,16])
plt.show()
```



In [14]:

```
x = np.linspace(0,2,100)

plt.plot(x, x, label = 'Linear')
plt.plot(x, x ** 2, label = 'quadratic')
plt.plot(x, x**3, label = 'cubic')

plt.xlabel('x label')
plt.ylabel('ylabel')

plt.title('Simple Plot')

plt.legend()

plt.show()
```

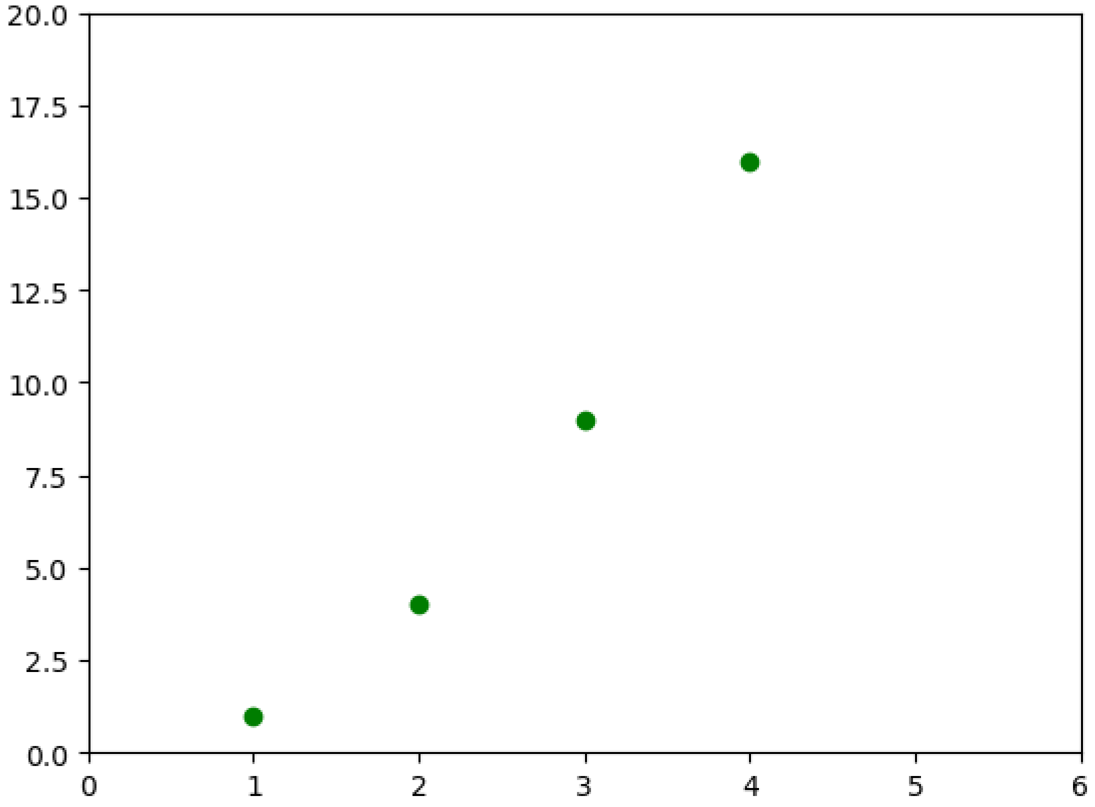


In [15]:

```
#Formatting the style of plot
```

In [16]:

```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'go')
plt.axis([0, 6, 0, 20])
plt.show()
```

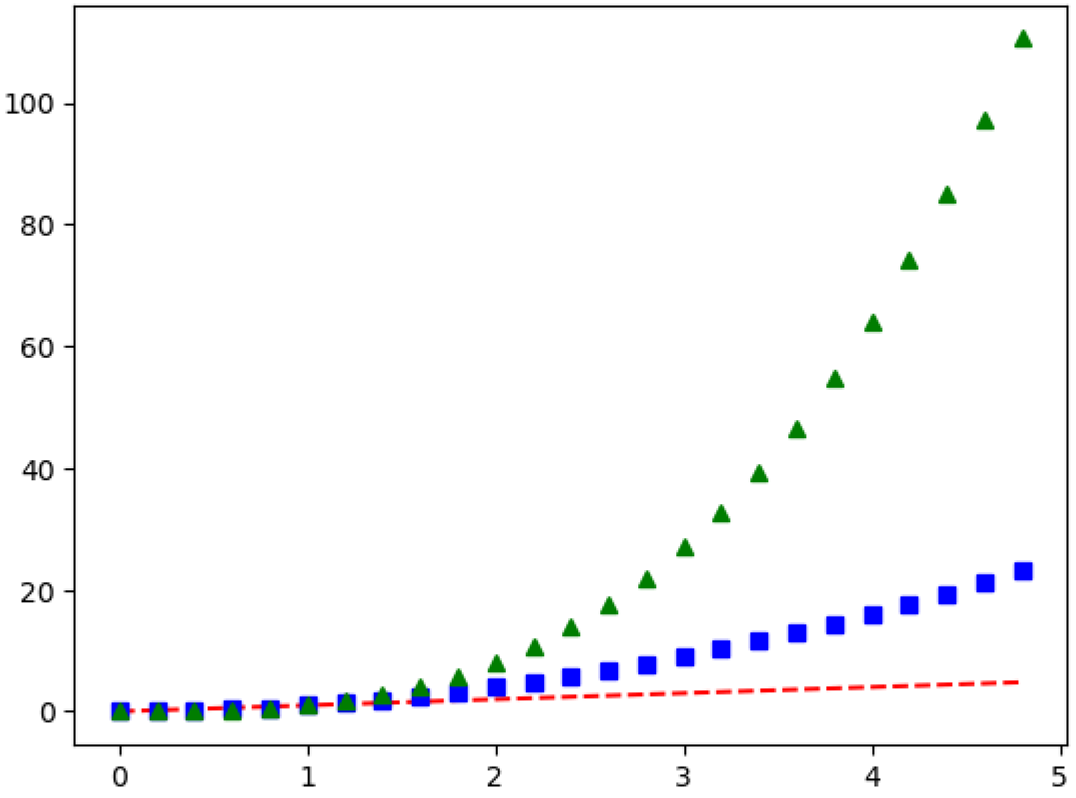


In [17]:

```
#Working with NumPy arrays
```

In [18]:

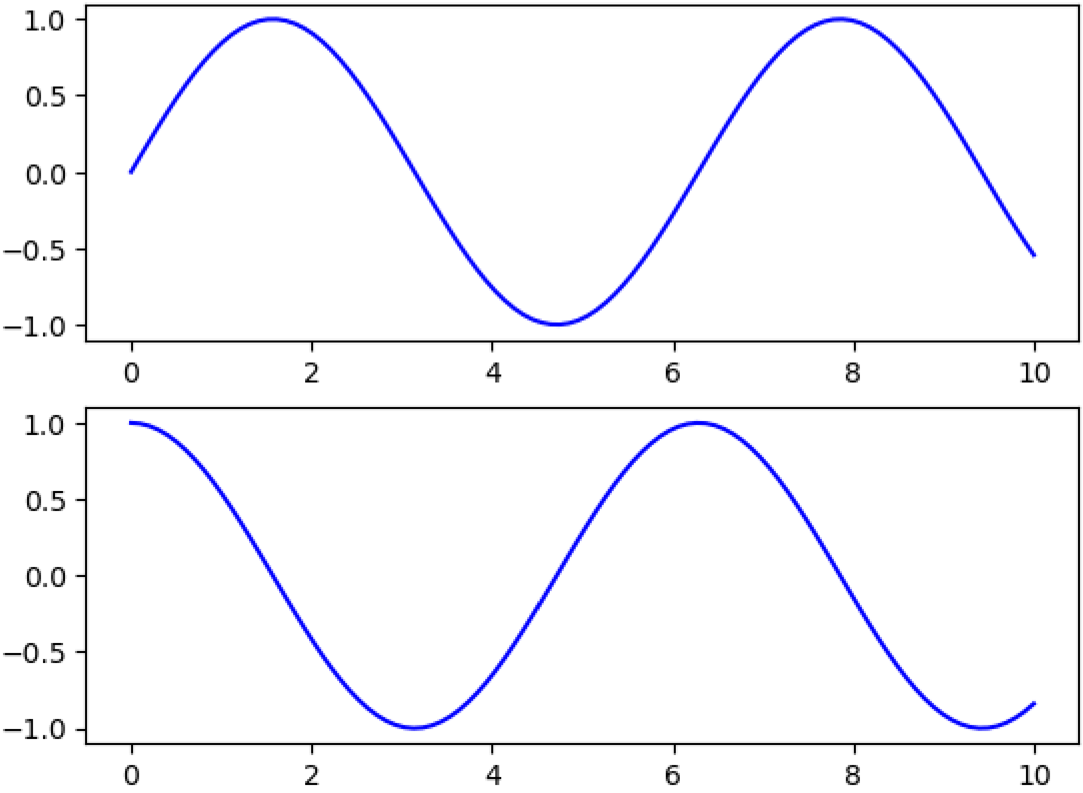
```
t = np.arange(0., 5., 0.2)

# red dashes, blue squares and green triangles
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```



In [19]:
```
#Object-Oriented API
```

In [20]:
```
# First create a grid of plots
# ax will be an array of two Axes objects
fig, ax = plt.subplots(2)

# Call plot() method on the appropriate object
ax[0].plot(x1, np.sin(x1), 'b-')
ax[1].plot(x1, np.cos(x1), 'b-')
```

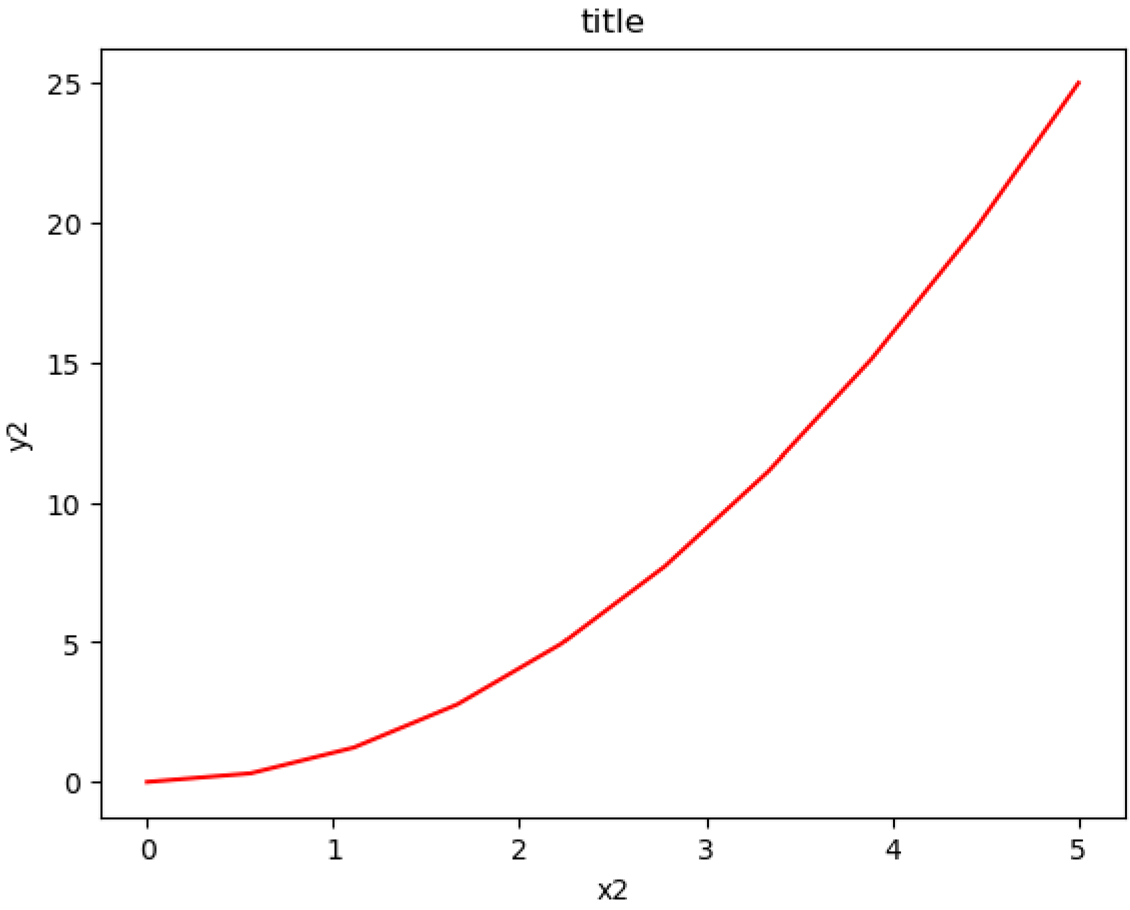
Out[20]:
```
[<matplotlib.lines.Line2D at 0x26bb885e180>]
```



In [21]:
```
#Objects and Reference
```

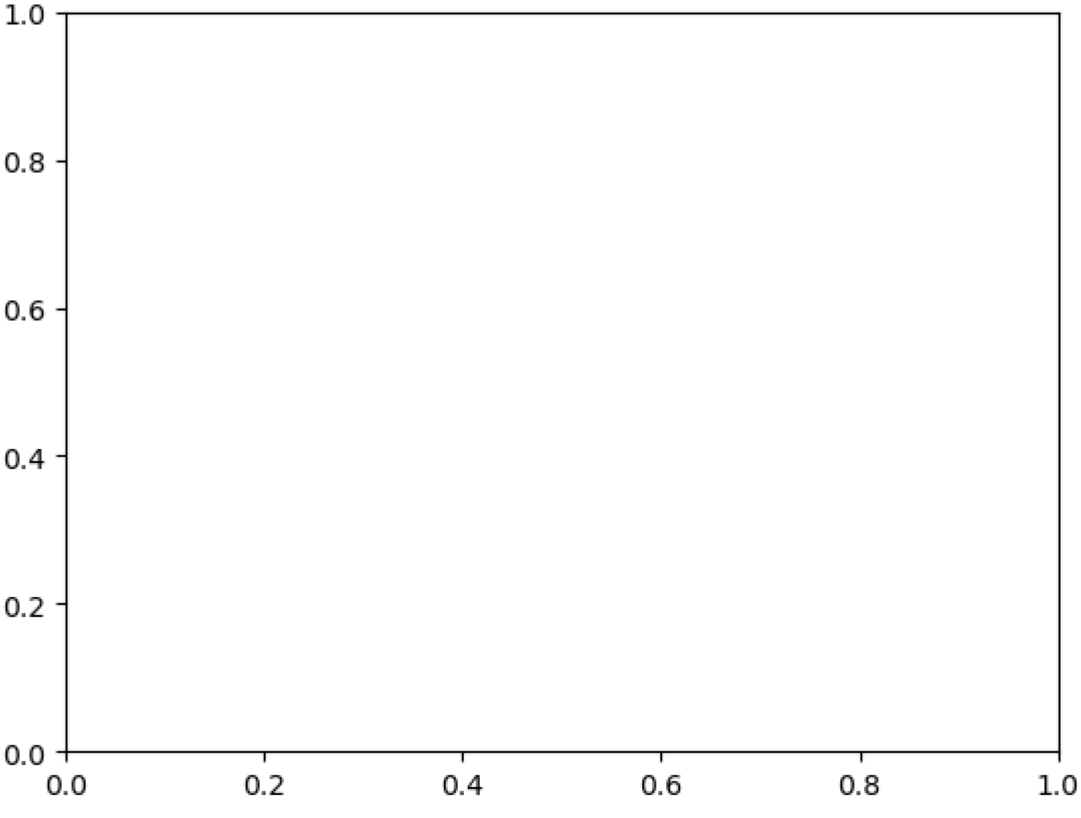
In [22]:
```
fig = plt.figure()

x2 = np.linspace(0,5,10)
y2 = x2 ** 2

axes = fig.add_axes([0.1, 0.1, 0.8, 0.8])

axes.plot(x2, y2, 'r')

axes.set_xlabel('x2')
axes.set_ylabel('y2')
axes.set_title('title')
```

Out[22]:
```
Text(0.5, 1.0, 'title')
```



```
In [23]: #Figure and Axes
```

```
In [24]: fig = plt.figure()
         ax = plt.axes()
```

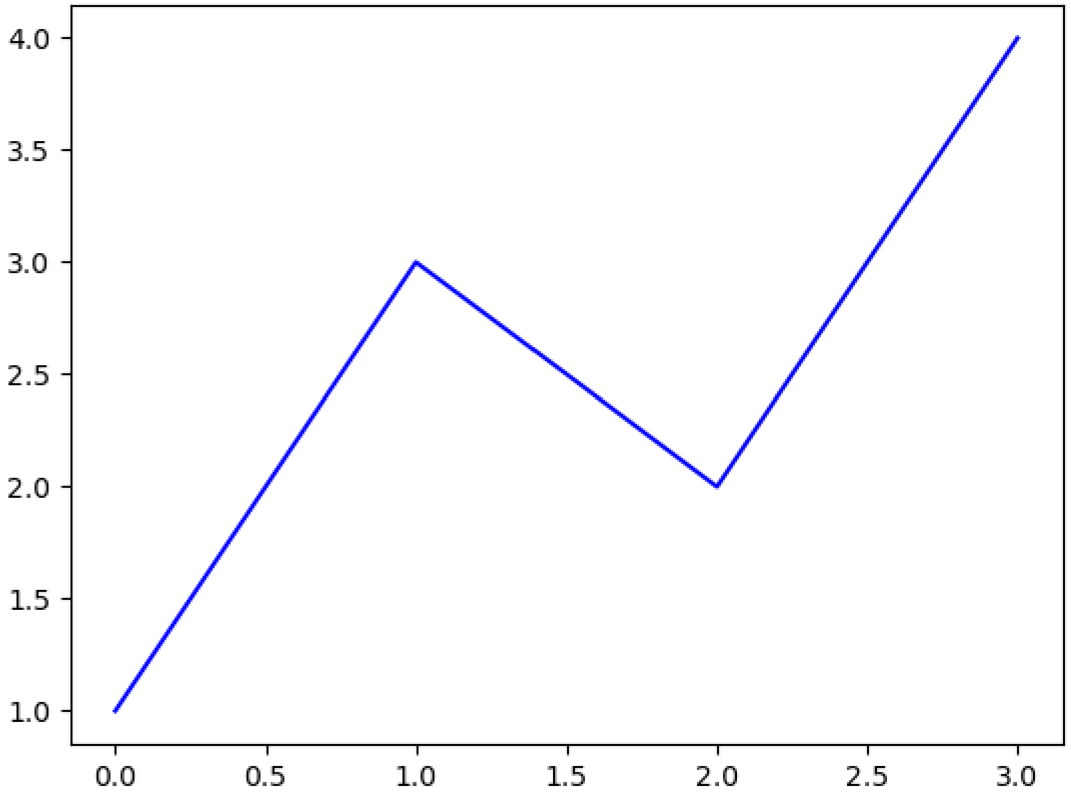


In [25]:

```
#First plot with Matplotlib
```

In [26]:

```
plt.plot([1, 3, 2, 4], 'b-')
plt.show( )
```

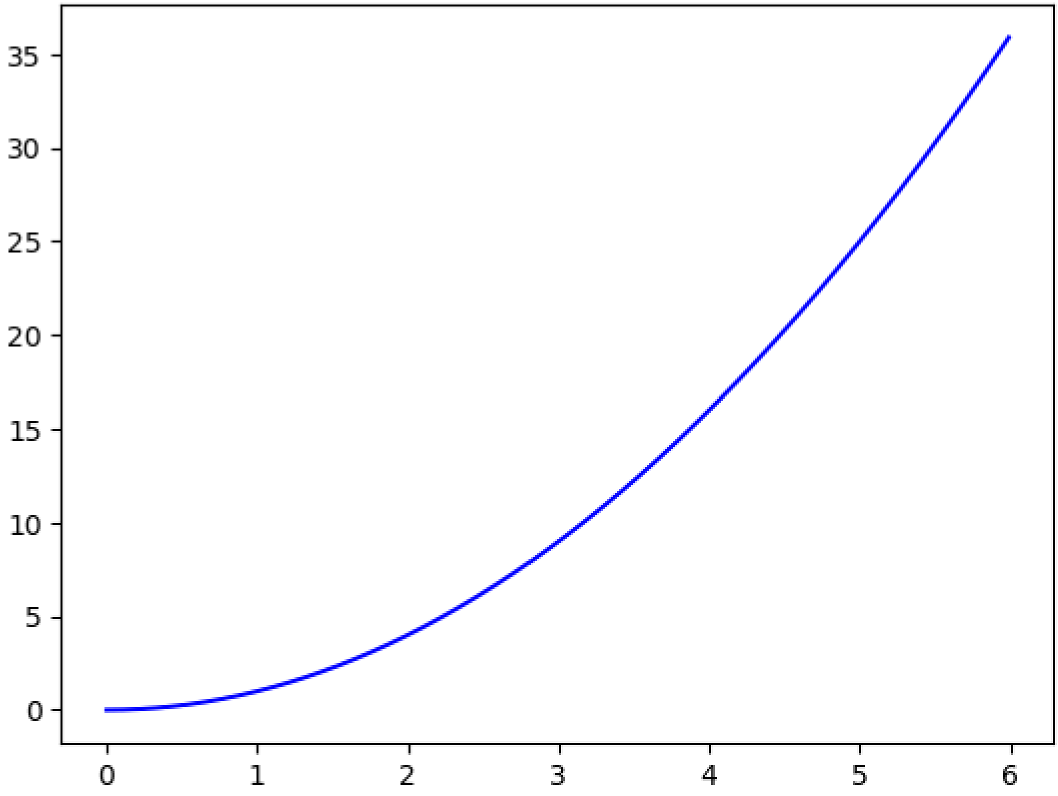


In [27]:

```
#Specify both Lists
```

In [28]:

```
x3 = np.arange(0.0, 6.0, 0.01)
plt.plot(x3, [xi**2 for xi in x3], 'b-')
plt.show()
```

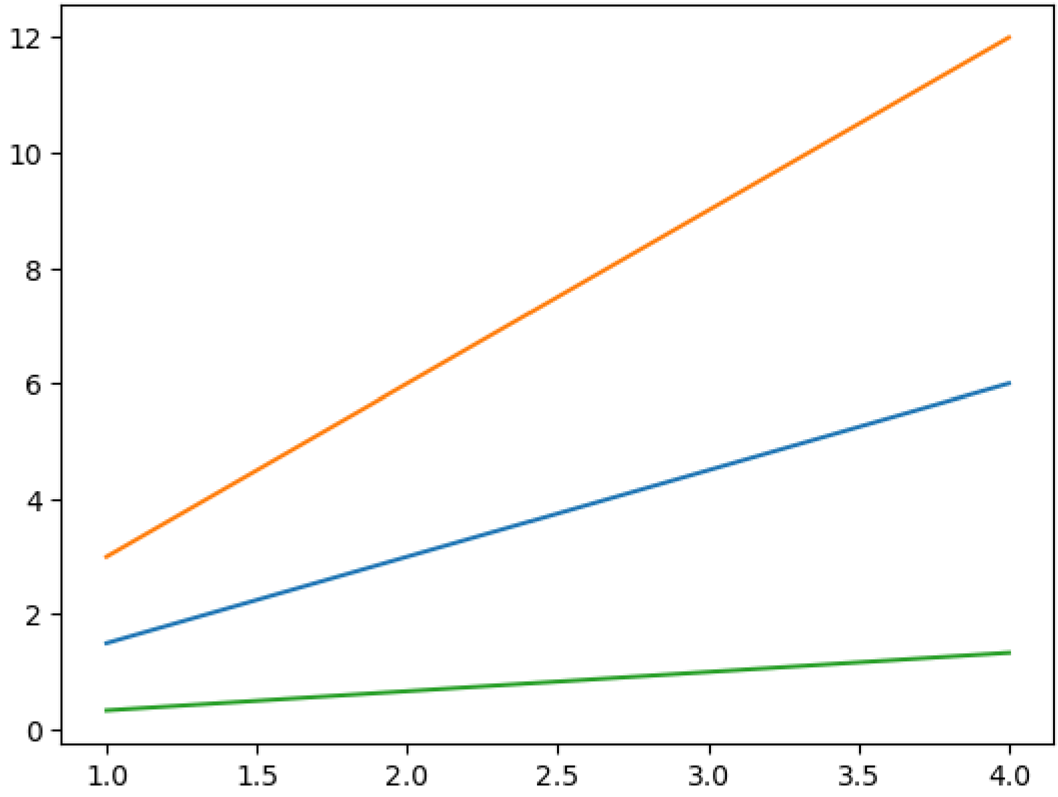


In [29]:

```
#Multiline Plots
```

In [30]:

```
x4 = range(1, 5)
plt.plot(x4, [xi*1.5 for xi in x4])
plt.plot(x4, [xi*3 for xi in x4])
plt.plot(x4, [xi/3.0 for xi in x4])
plt.show()
```



In [31]:

```
#Parts of a Plot
```

In [32]:

```
fig.savefig('plot1.png')
```

In [33]:

```
fig.canvas.get_supported_filetypes()
```

Out[33]:

```
{'eps': 'Encapsulated Postscript',
 'jpg': 'Joint Photographic Experts Group',
 'jpeg': 'Joint Photographic Experts Group',
 'pdf': 'Portable Document Format',
 'pgf': 'PGF code for LaTeX',
 'png': 'Portable Network Graphics',
 'ps': 'Postscript',
 'raw': 'Raw RGBA bitmap',
 'rgba': 'Raw RGBA bitmap',
 'svg': 'Scalable Vector Graphics',
 'svgz': 'Scalable Vector Graphics',
 'tif': 'Tagged Image File Format',
 'tiff': 'Tagged Image File Format',
 'webp': 'WebP Image Format'}
```

In [34]:

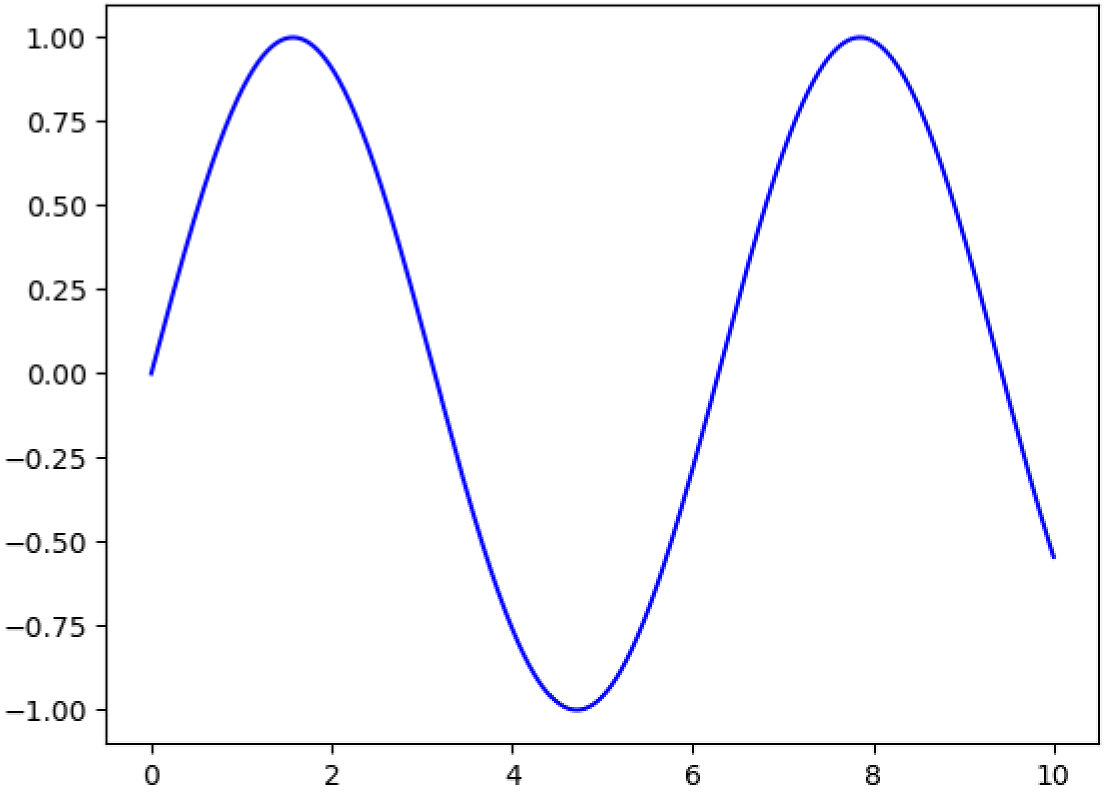
```
#Line plot
```

In [35]:

```
# Create figure and axes first
fig = plt.figure()
ax = plt.axes()

# Declare a variable x5
x5 = np.linspace(0, 10, 1000)

# Plot the sinusoid function
ax.plot(x5, np.sin(x5), 'b-')
```

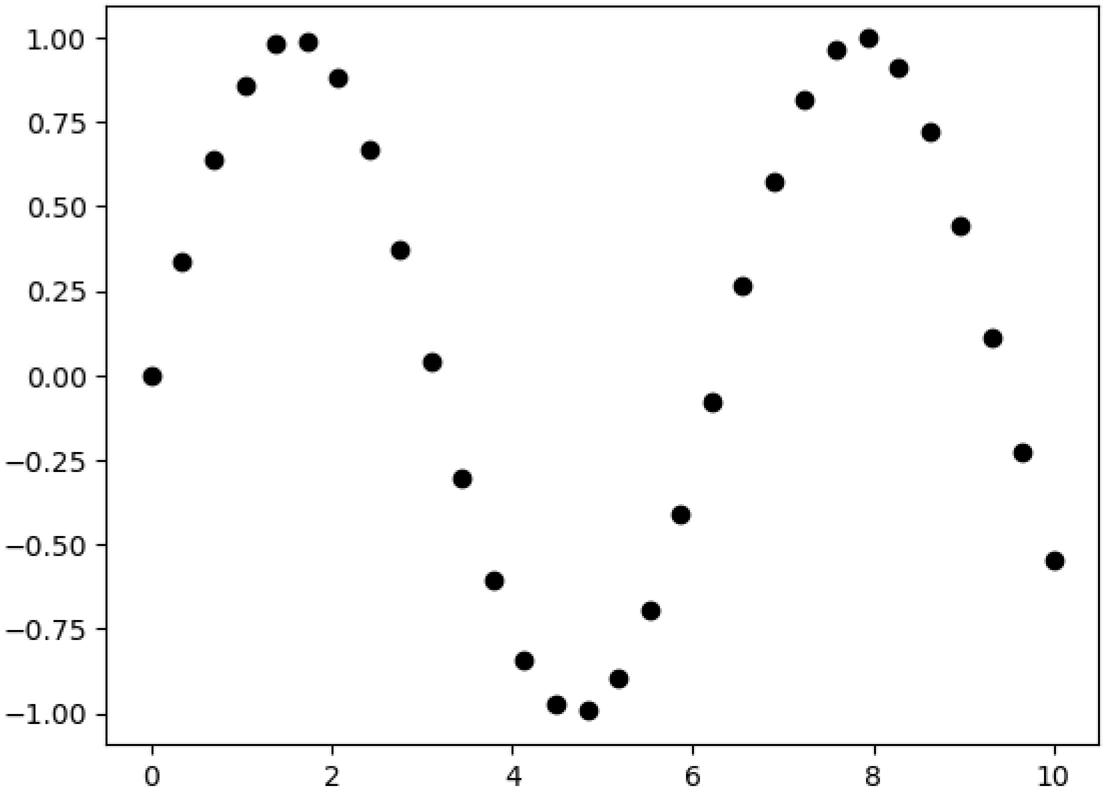
Out[35]:

```
[<matplotlib.lines.Line2D at 0x26bb7f7ae10>]
```



In [36]:
```
#Scatter Plot
```

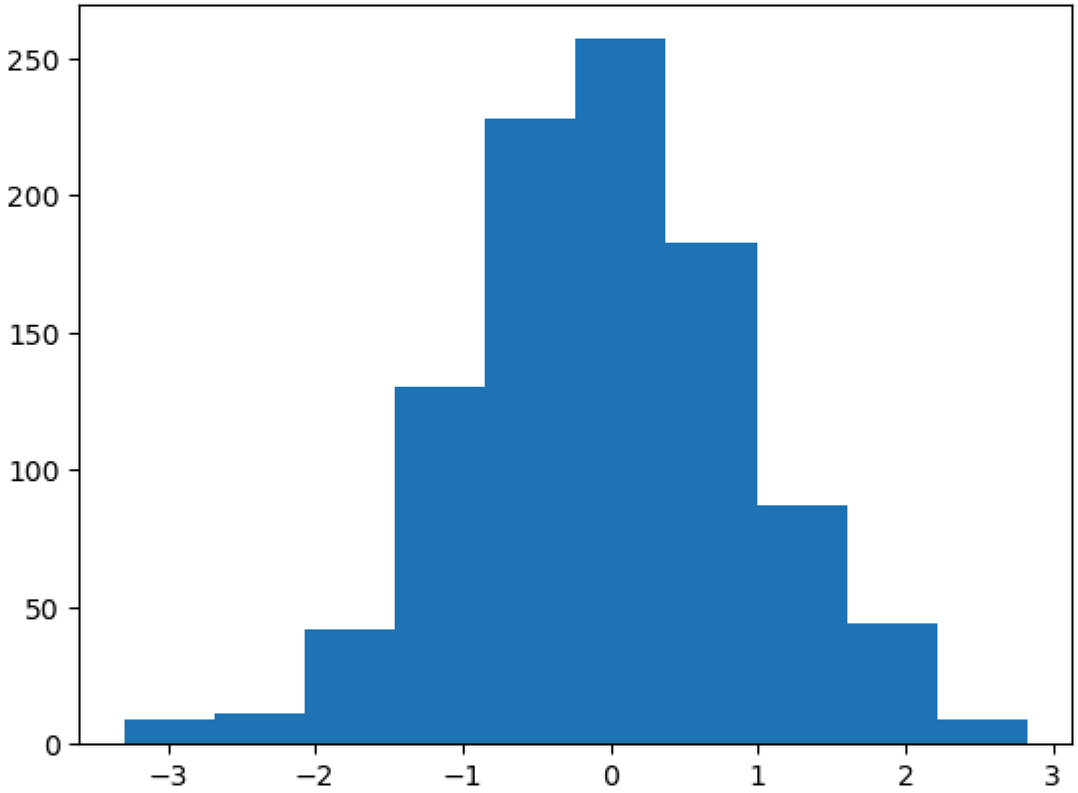
In [37]:
```
x7 = np.linspace(0, 10, 30)
y7 = np.sin(x7)
plt.plot(x7, y7, 'o', color = 'black');
```



In [38]:
```
#Histogram
```

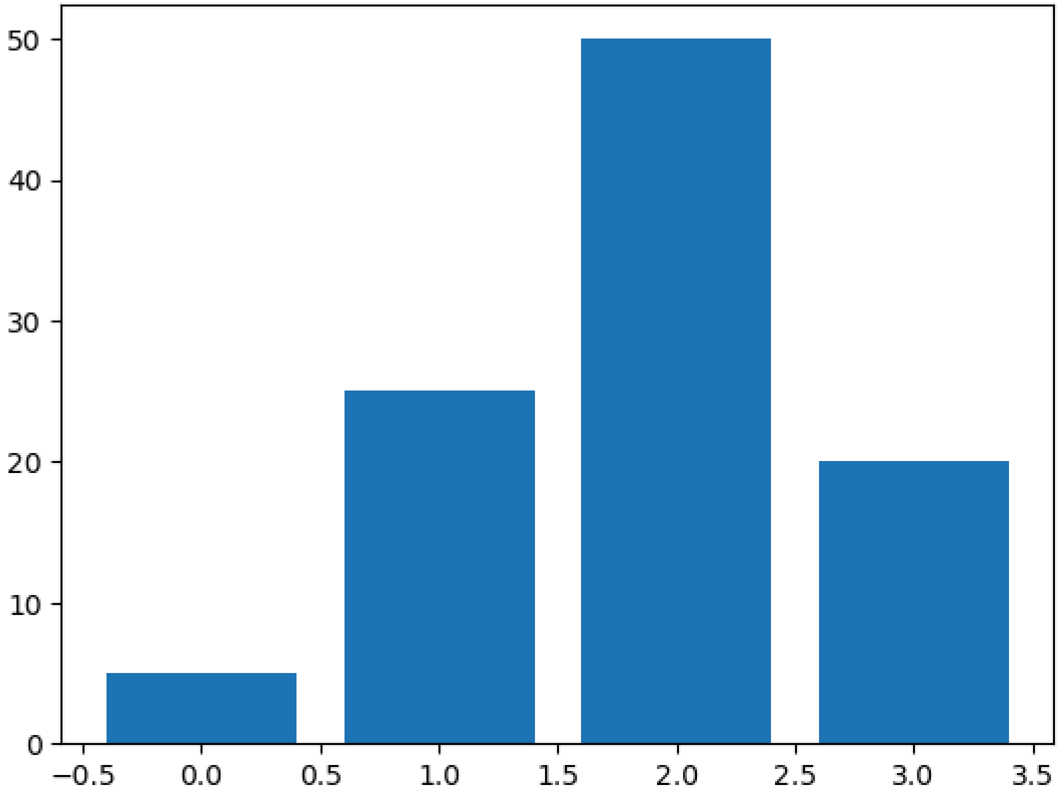
```
In [39]: data1 = np.random.randn(1000) #randn- random numbers
         plt.hist(data1)
```

```
Out[39]: (array([  9.,  11.,  42., 130., 228., 257., 183.,  87.,  44.,   9.]),
          array([-3.29367376, -2.68187339, -2.07007301, -1.45827263, -0.84647225,
                 -0.23467187,  0.3771285 ,  0.98892888,  1.60072926,  2.21252964,
                  2.82433002]),
          <BarContainer object of 10 artists>)
```



```
In [40]: #Bar Chart
```

```
In [41]: data2 = [5. , 25. , 50. , 20.]
         plt.bar(range(len(data2)), data2)
         plt.show()
```

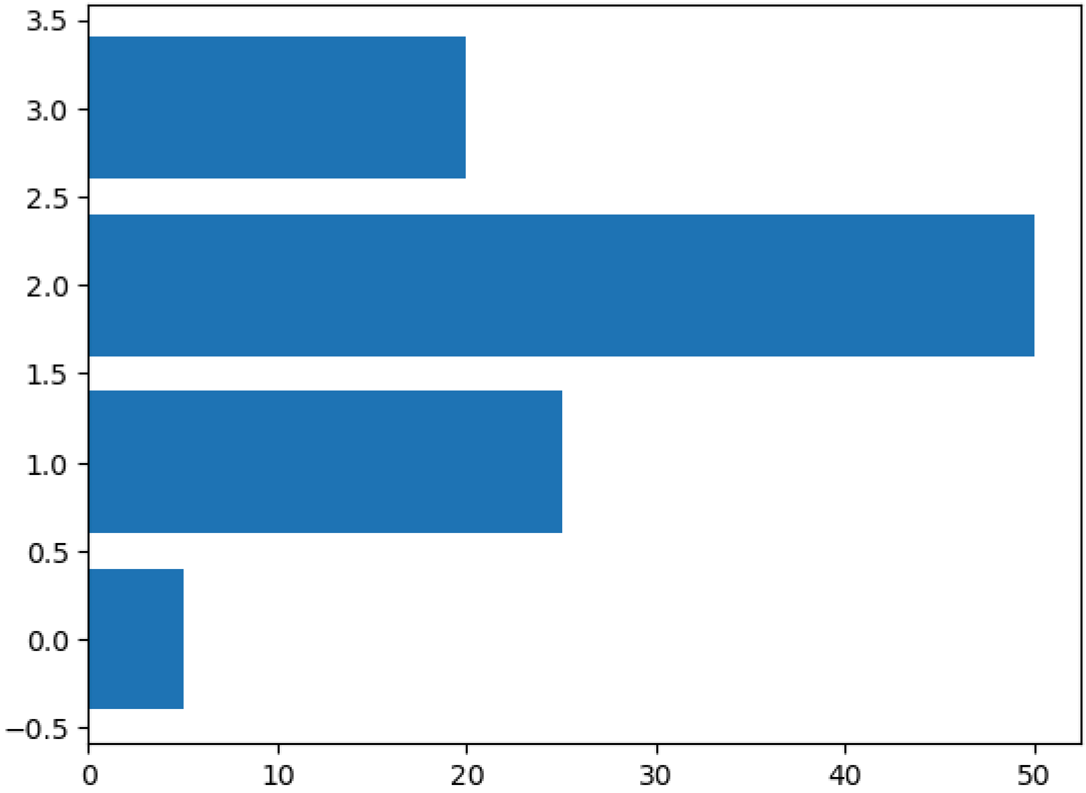


In [42]:

```
#Horizontal Bar Chart
```

In [43]:

```
data2 = [5. , 25. , 50. , 20.]
plt.barh(range(len(data2)), data2)
plt.show()
```

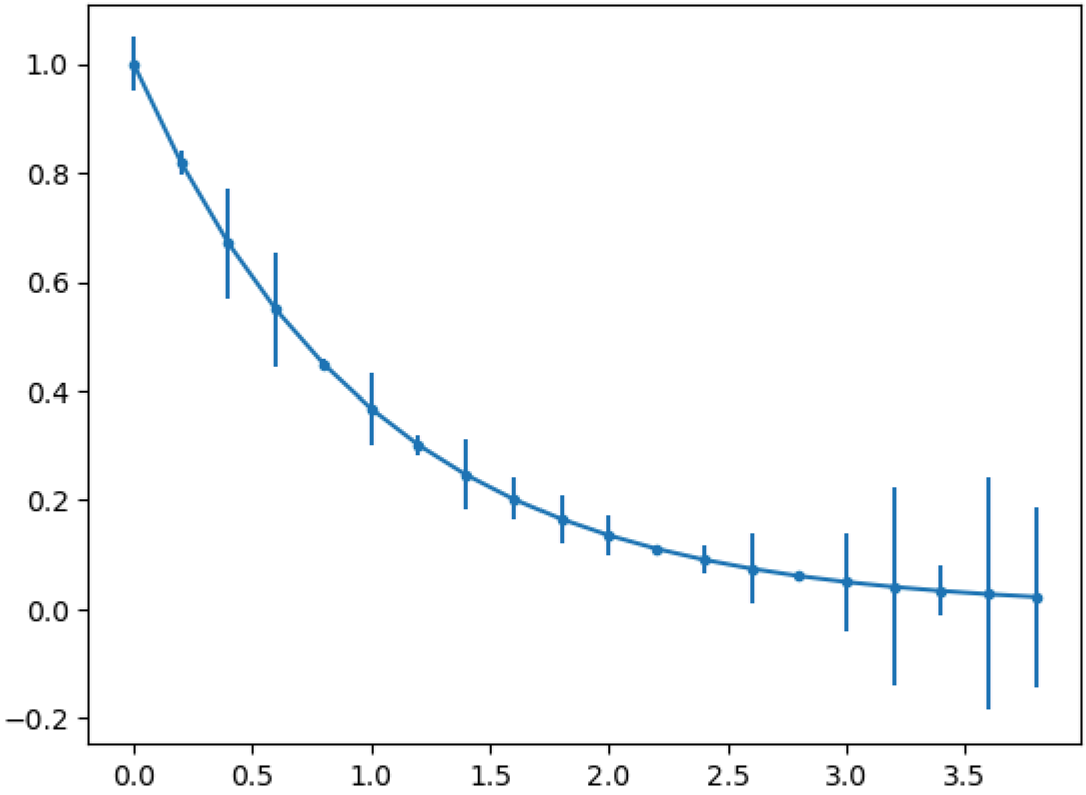


In [44]:

```
#Error Bar Chart
```

In [45]:

```
x9 = np.arange(0, 4, 0.2)
y9 = np.exp(-x9)
e1 = 0.1 * np.abs(np.random.randn(len(y9)))
plt.errorbar(x9, y9, yerr = e1, fmt = '.-')
plt.show()
```

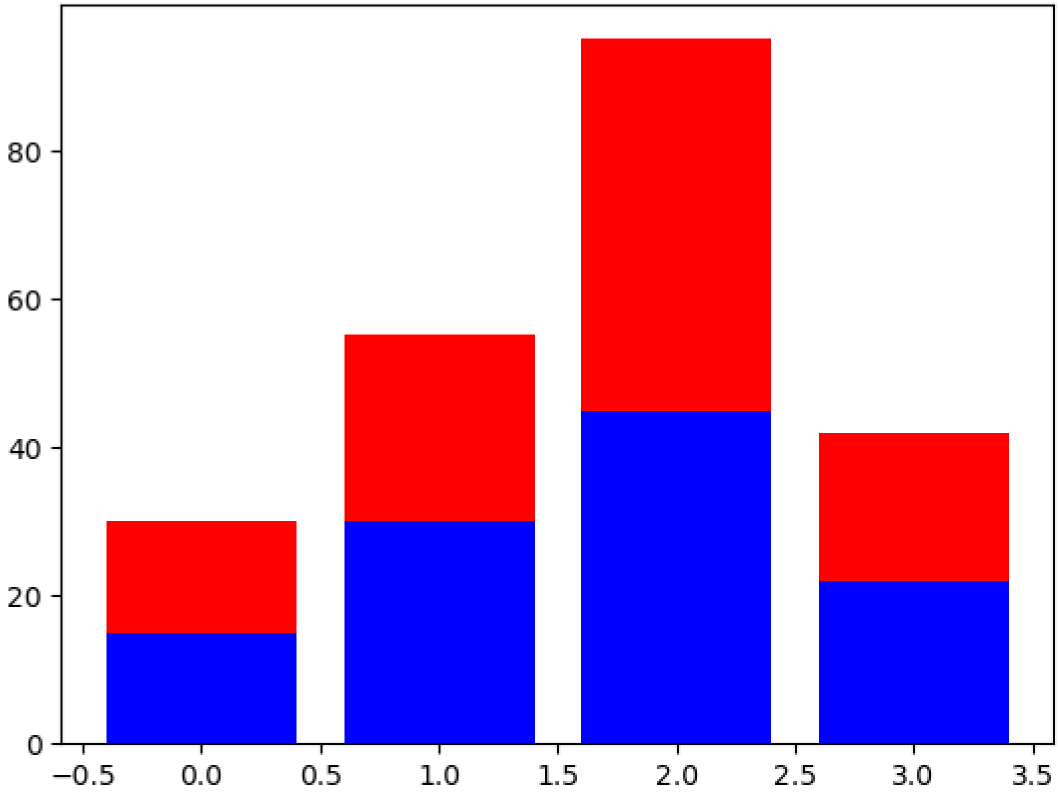


In [46]:

```
#Stacked Bar Chart
```

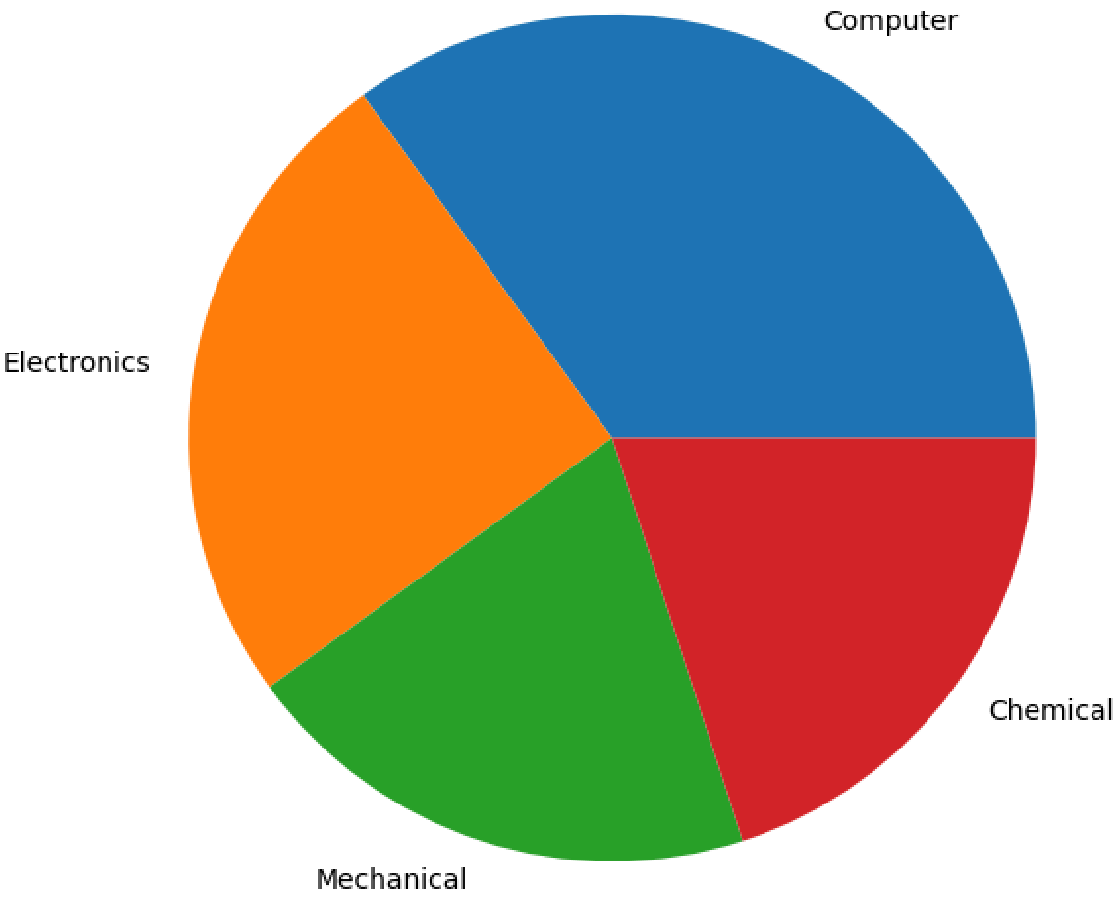
In [47]:

```
A = [15., 30., 45., 22.]
B = [15., 25., 50., 20.]
z2 = range(4)
plt.bar(z2, A, color = 'b')
plt.bar(z2, B, color = 'r', bottom = A)
plt.show()
```



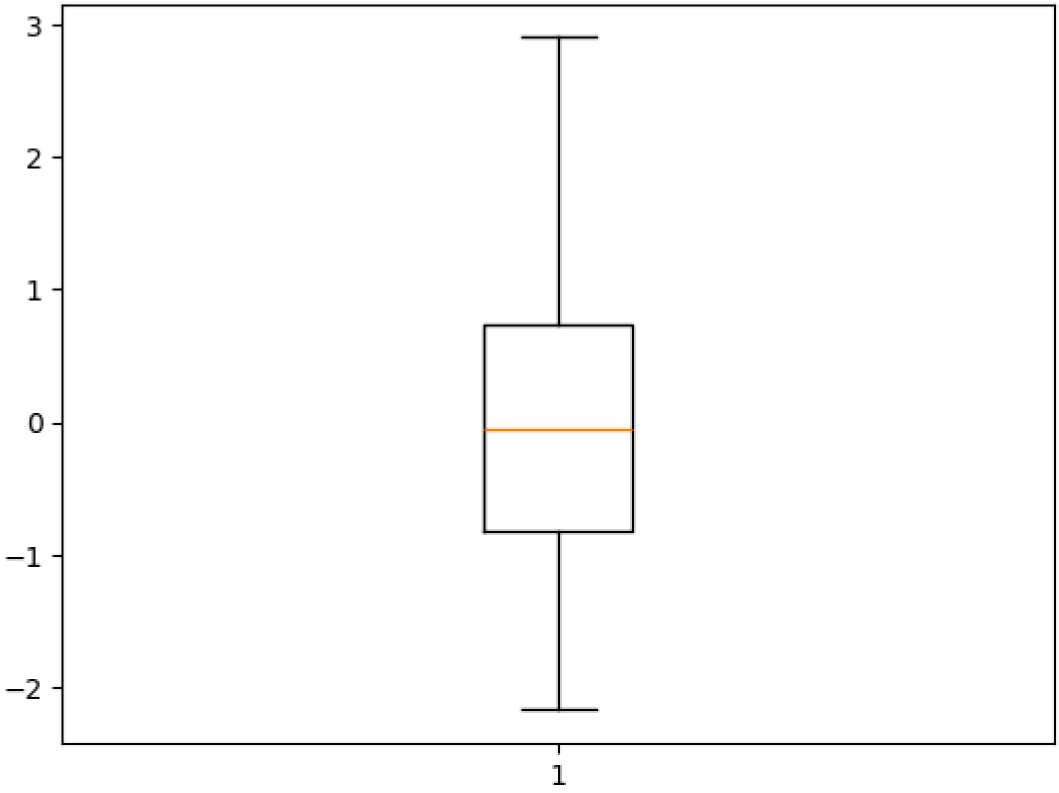
In [48]:
```
#Pie Chart
```

In [49]:
```
plt.figure(figsize=(7,7))
x10 = [35, 25, 20, 20]
labels = ['Computer', 'Electronics', 'Mechanical', 'Chemical']
plt.pie(x10, labels=labels)
plt.show()
```



In [50]:
```
#Boxplot
```

In [51]:
```
data3 = np.random.randn(100)
plt.boxplot(data3)
plt.show()
```

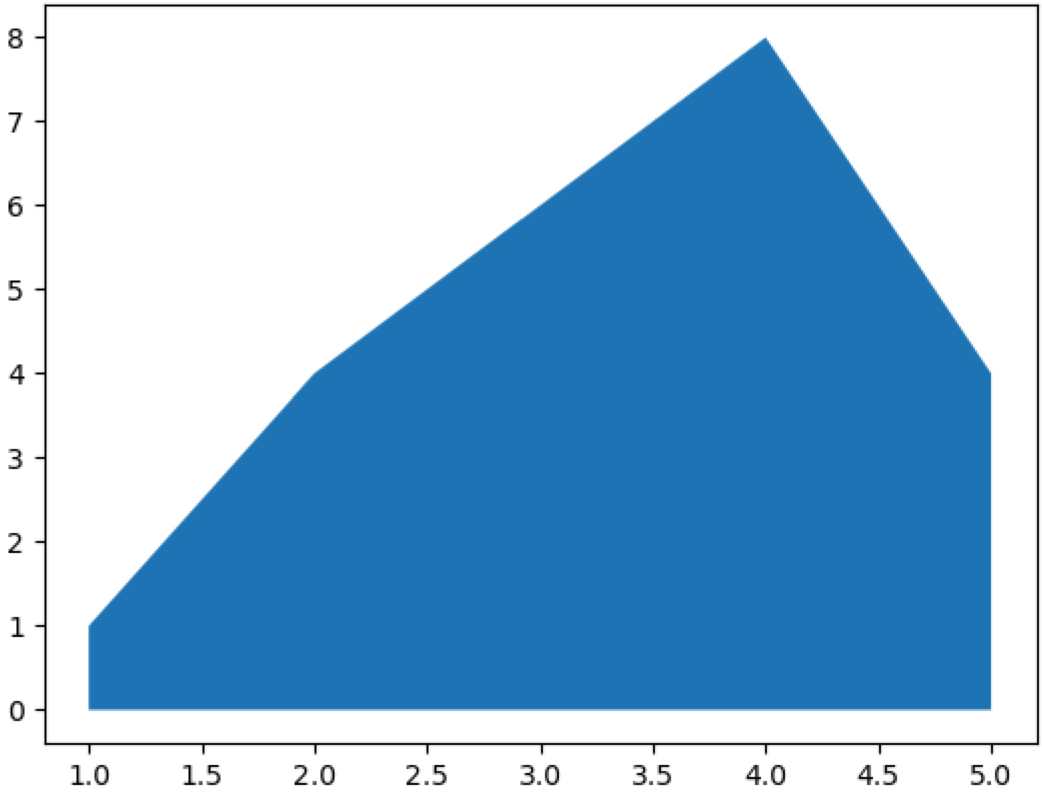


In [52]:

```
#Area Chart
```

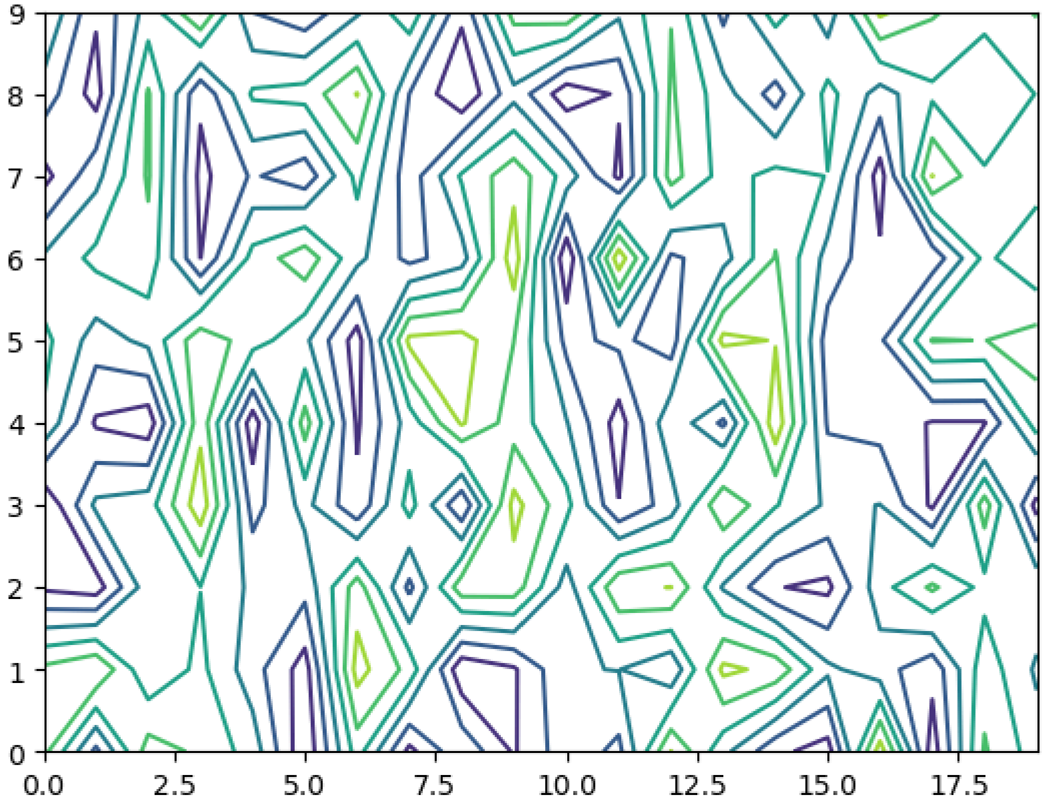
In [53]:

```
# Create some data
x12 = range(1, 6)
y12 = [1, 4, 6, 8, 4]
# Area plot
plt.fill_between(x12, y12)
plt.show()
```



In [54]:
```
#Contour Plot
```

In [55]:
```
# Create a matrix
matrix1 = np.random.rand(10, 20)
cp = plt.contour(matrix1)
plt.show()
```



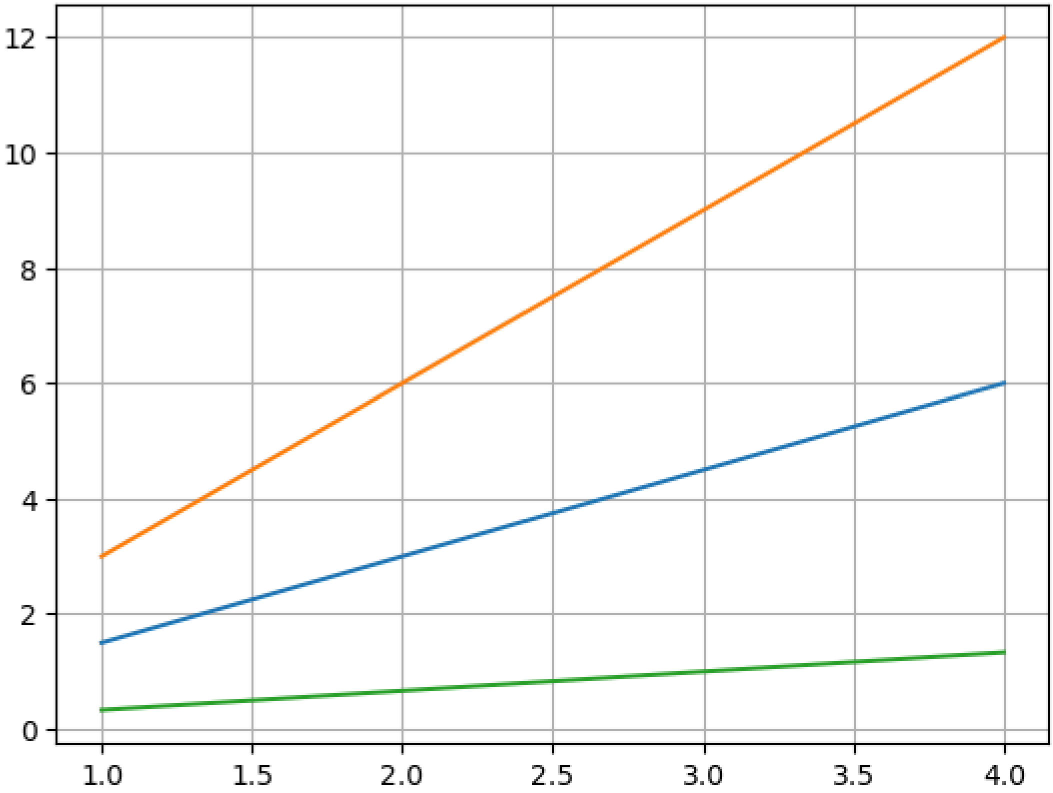
In [56]:
```
#Styles with Matplotlib Plots
```

In [57]:
```
print(plt.style.available)
```

```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid',
'bmh', 'classic', 'dark_background', 'fast', 'fivethirtyeight', 'ggplot', 'graysc
ale', 'seaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblind', 'seaborn-
v0_8-dark', 'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-d
eep', 'seaborn-v0_8-muted', 'seaborn-v0_8-notebook', 'seaborn-v0_8-paper', 'seabo
rn-v0_8-pastel', 'seaborn-v0_8-poster', 'seaborn-v0_8-talk', 'seaborn-v0_8-tick
s', 'seaborn-v0_8-white', 'seaborn-v0_8-whitegrid', 'tableau-colorblind10']
```

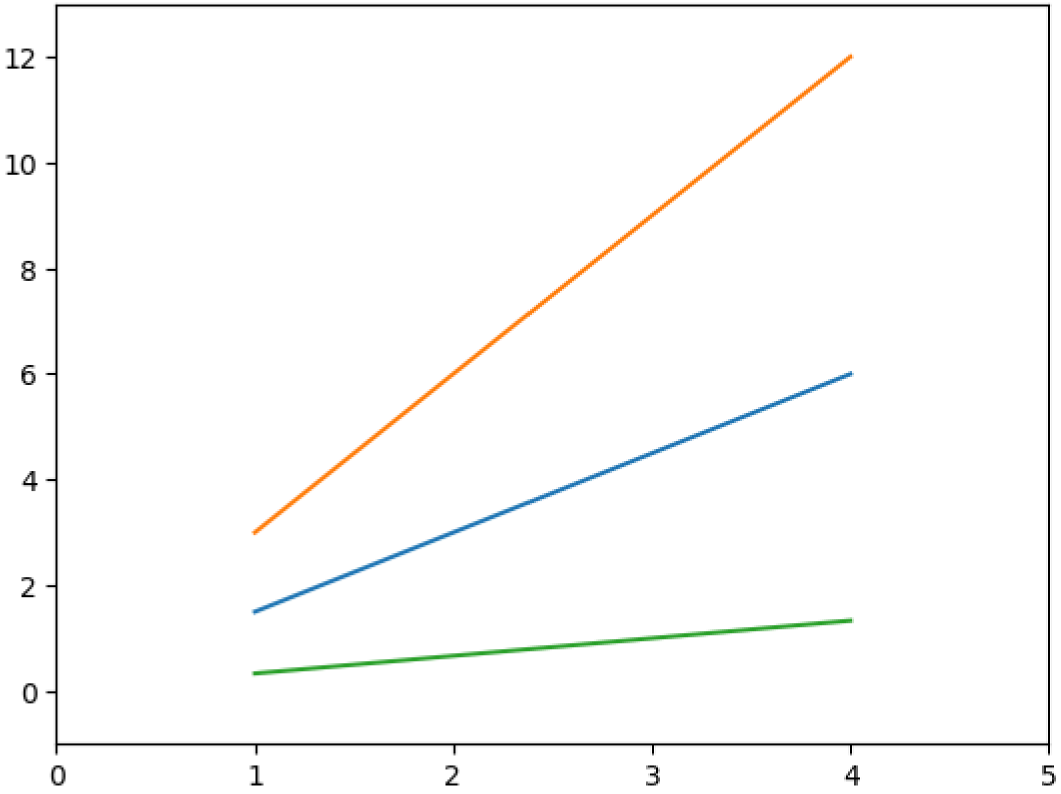
In [58]:
```
#Adding a grid
```

In [59]:
```
x15 = np.arange(1, 5)
plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
plt.grid(True)
plt.show()
```



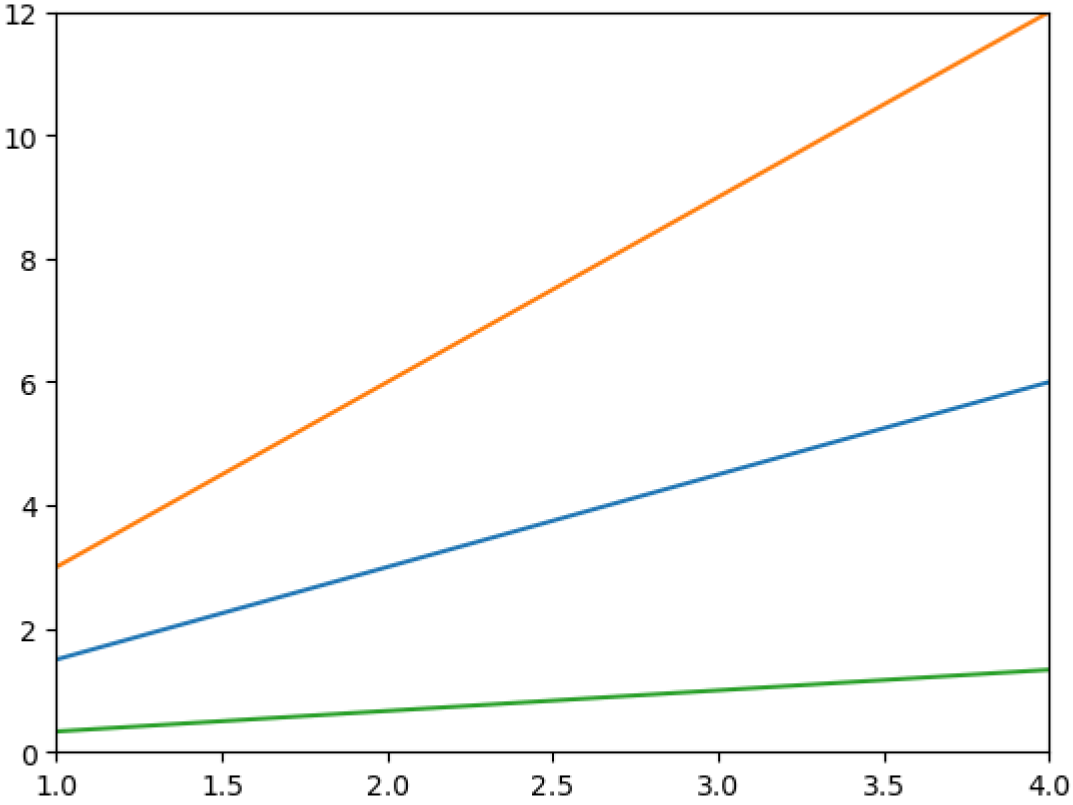
```
In [60]: #Handling axes
```

```
In [61]: x15 = np.arange(1, 5)
         plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
         plt.axis() # shows the current axis limits values
         plt.axis([0, 5, -1, 13])
         plt.show()
```



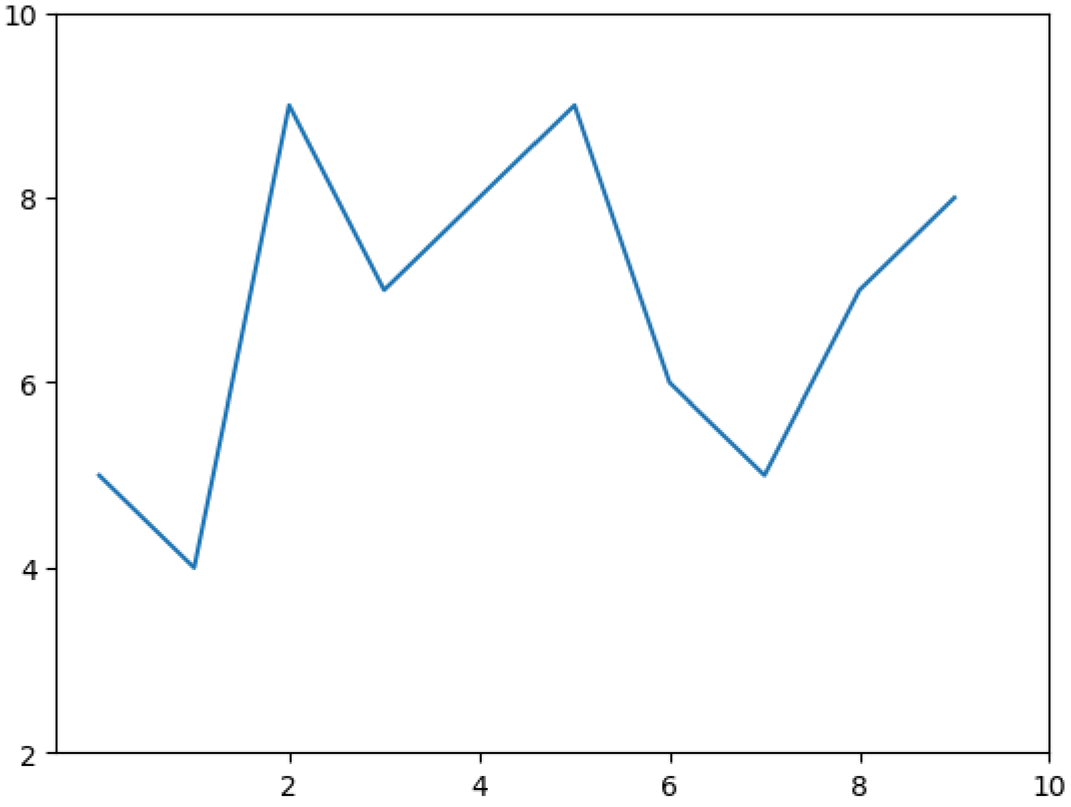
In [62]:
```
x15 = np.arange(1, 5)
plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
plt.xlim([1.0, 4.0])
plt.ylim([0.0, 12.0])
```

Out[62]:
```
(0.0, 12.0)
```



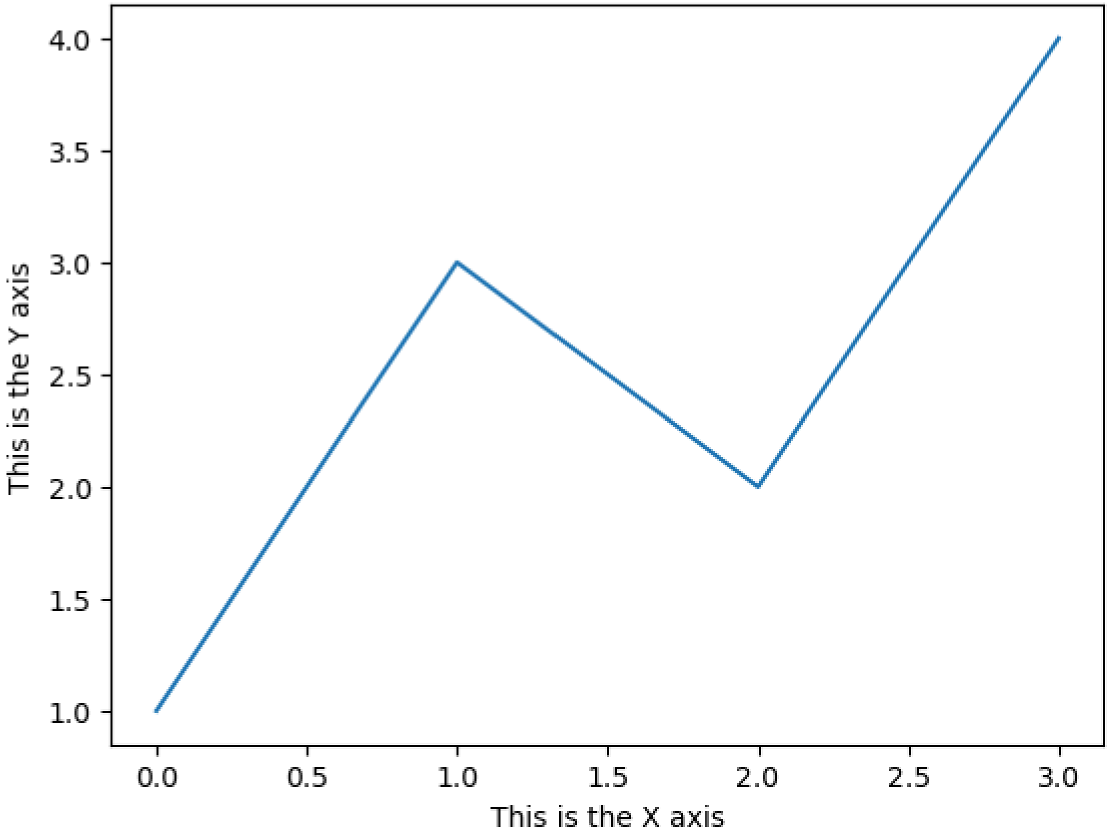
In [63]:
```
#Handling X and Y tickss
```

In [64]:
```
u = [5, 4, 9, 7, 8, 9, 6, 5, 7, 8]
plt.plot(u)
plt.xticks([2, 4, 6, 8, 10])
plt.yticks([2, 4, 6, 8, 10])
plt.show()
```



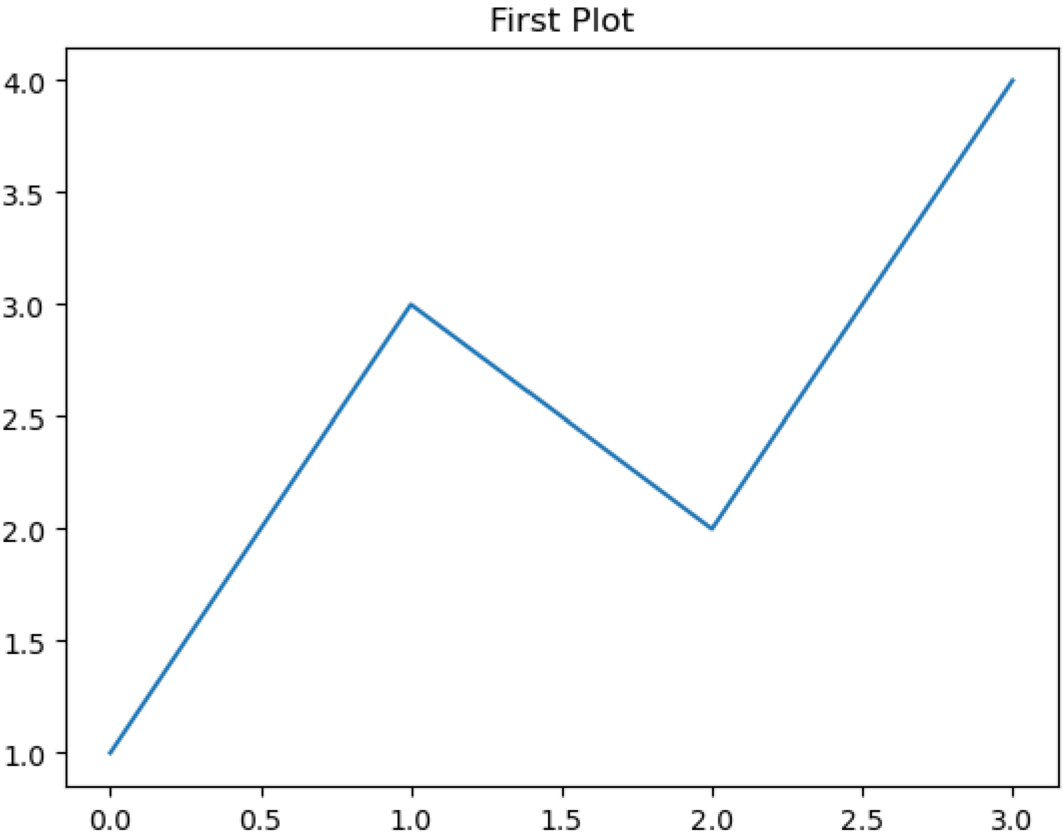
```
In [65]: #Adding labels
```

```
In [66]: plt.plot([1, 3, 2, 4])
         plt.xlabel('This is the X axis')
         plt.ylabel('This is the Y axis')
         plt.show()
```



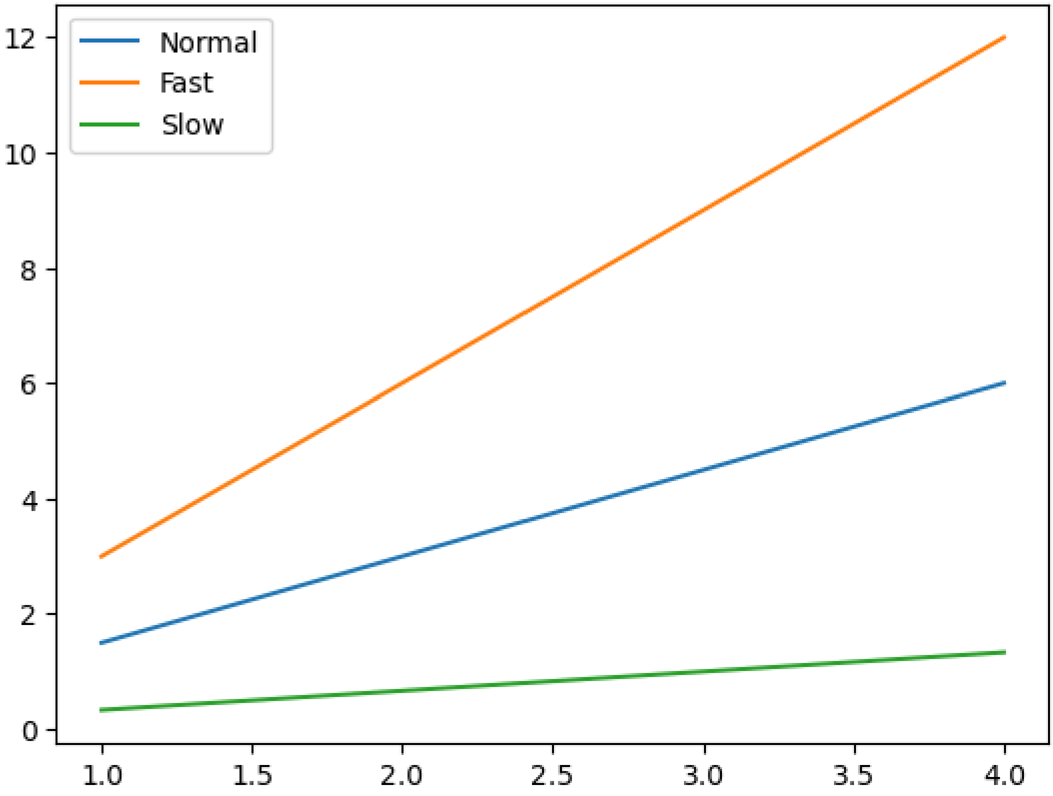
```
In [67]: #Adding a title
```

```
In [68]: plt.plot([1, 3, 2, 4])
         plt.title('First Plot')
         plt.show()
```



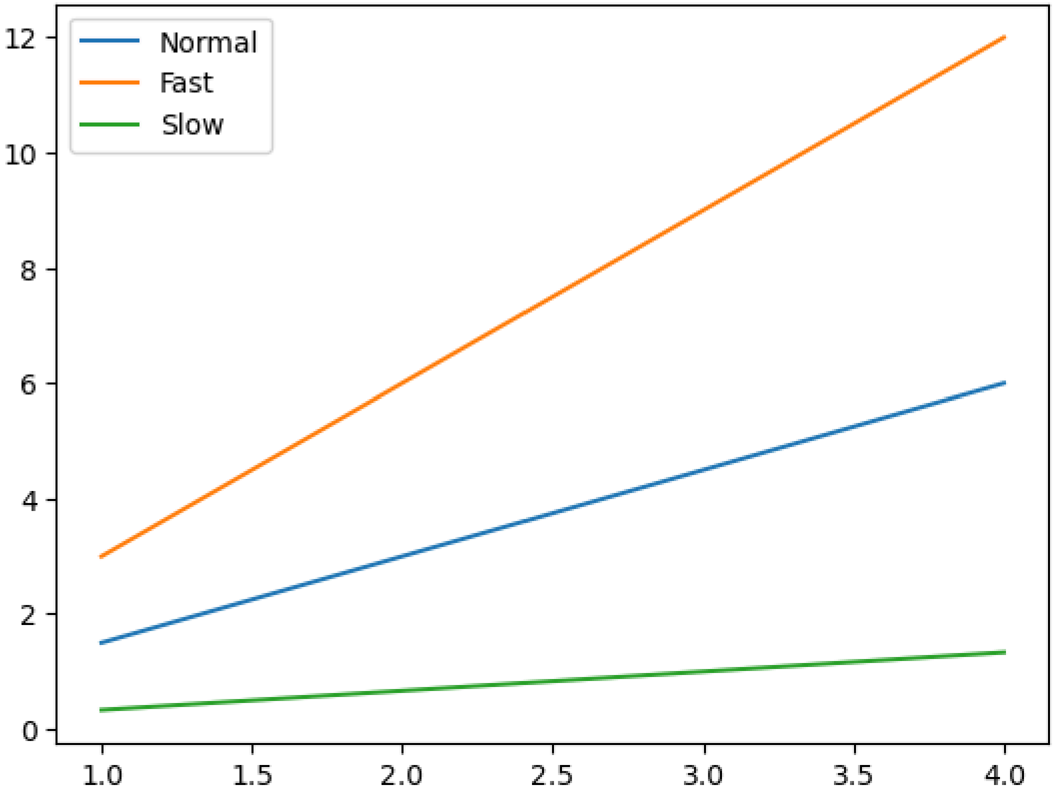
```
In [69]: #Adding a legend
```

```
In [70]: x15 = np.arange(1, 5)
         fig, ax = plt.subplots()
         ax.plot(x15, x15*1.5)
         ax.plot(x15, x15*3.0)
         ax.plot(x15, x15/3.0)
         ax.legend(['Normal','Fast','Slow'])

Out[70]: <matplotlib.legend.Legend at 0x26bb9cd9490>
```



```
In [71]: x15 = np.arange(1, 5)
         fig, ax = plt.subplots()
         ax.plot(x15, x15*1.5, label='Normal')
         ax.plot(x15, x15*3.0, label='Fast')
         ax.plot(x15, x15/3.0, label='Slow')
         ax.legend()
```

```
Out[71]: <matplotlib.legend.Legend at 0x26bb87bce90>
```

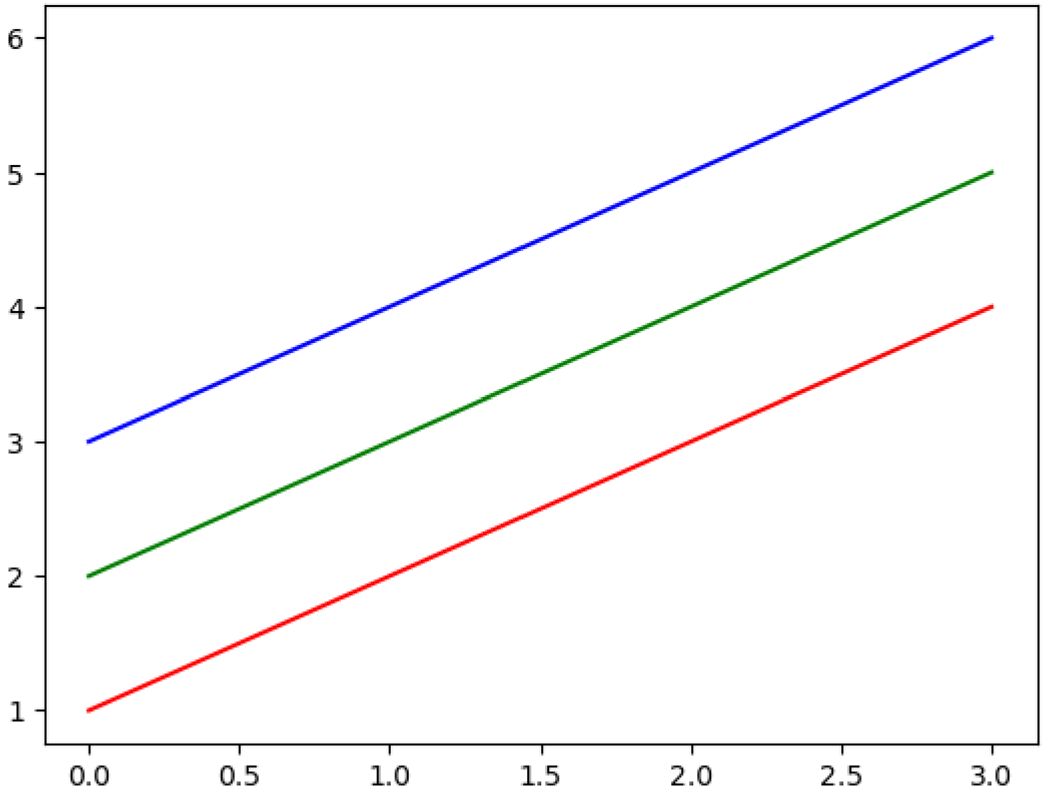


In [72]:

```
#Control colours
```

In [73]:

```
x16 = np.arange(1, 5)
plt.plot(x16, 'r')
plt.plot(x16+1, 'g')
plt.plot(x16+2, 'b')
plt.show()
```



In [ ]: