Matplotlib :

In [1]:

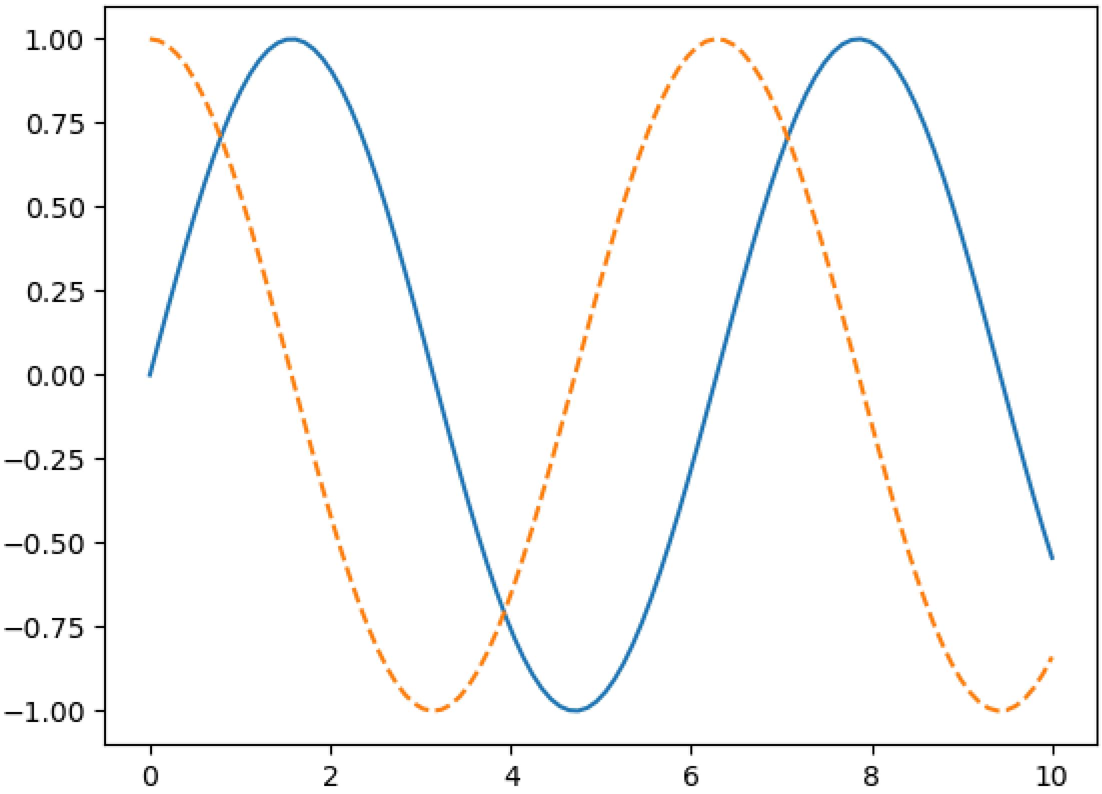
```
import numpy as np
import pandas as pd
```

In [3]:

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

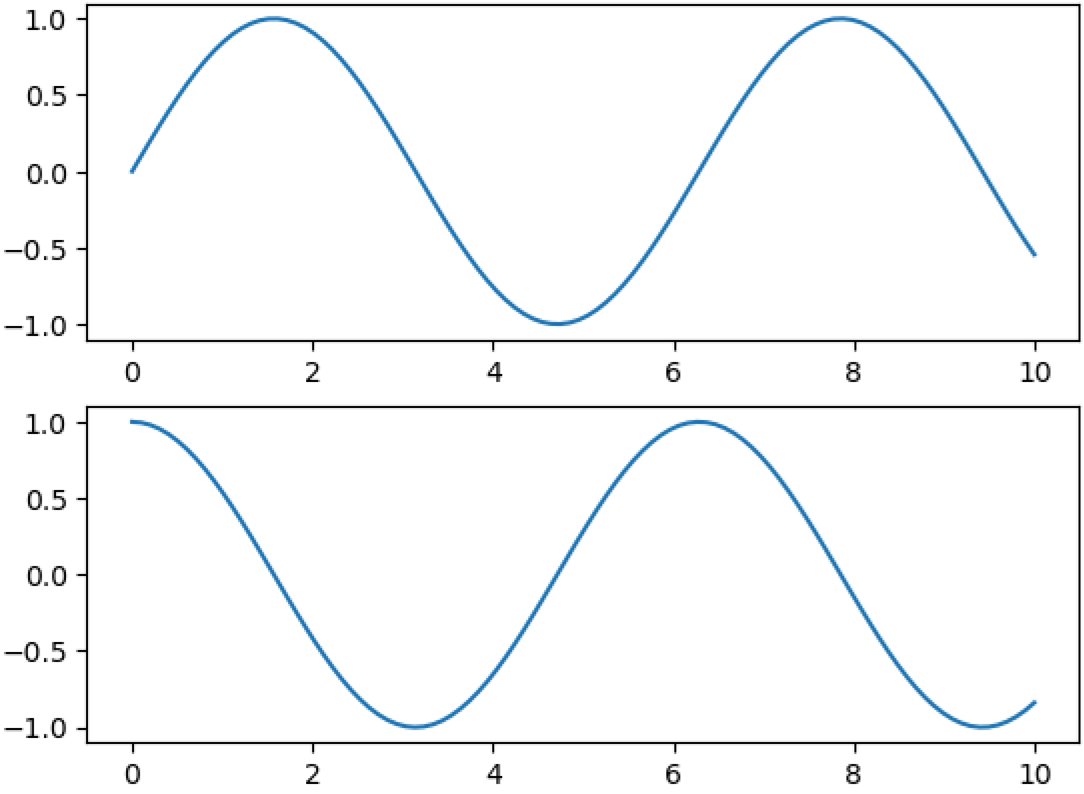
In [4]:

```
%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), '--')
plt.show()
```



In [5]:

```
# create a plot figure
plt.figure()

plt.subplot(2, 1, 1)   # (rows, columns, panel number)
plt.plot(x1, np.sin(x1))

# create the second of two panels and set current axis
plt.subplot(2, 1, 2)   # (rows, columns, panel number)
plt.plot(x1, np.cos(x1));
plt.show()
```



In [6]:

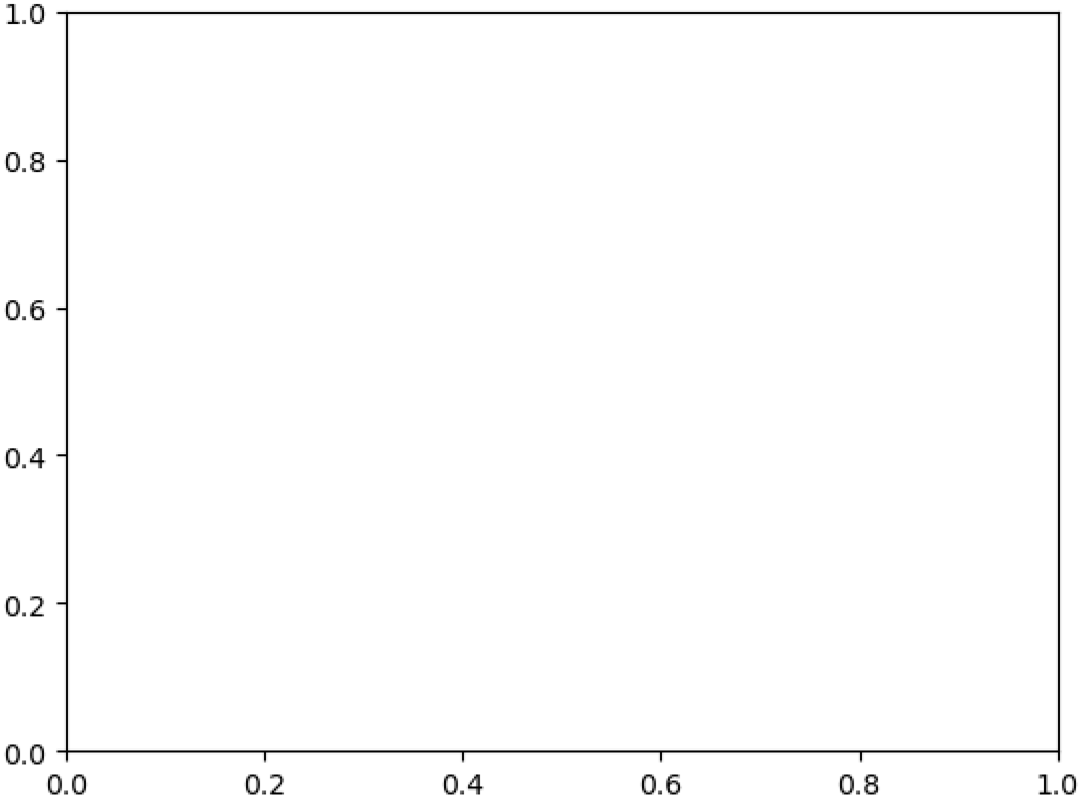
```
# get current figure information

print(plt.gcf())
```

```
Figure(640x480)
```

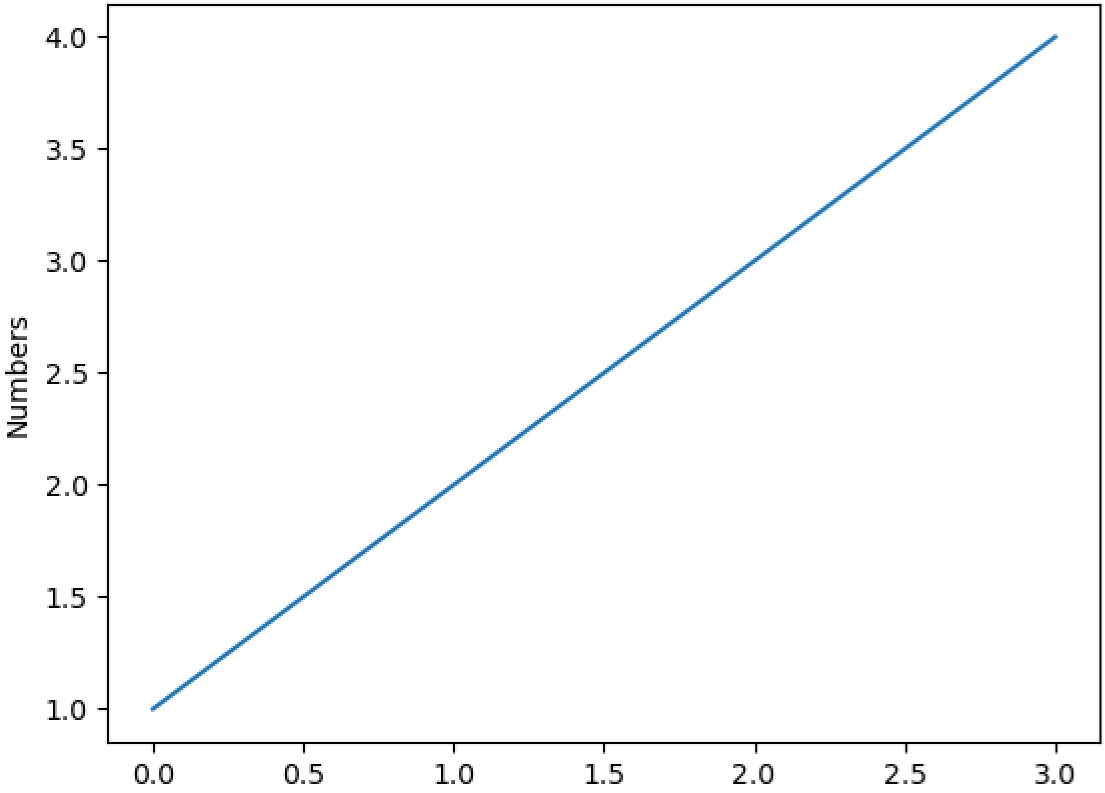
In [7]:

```
# get current axis information

print(plt.gca())
plt.show()
```

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

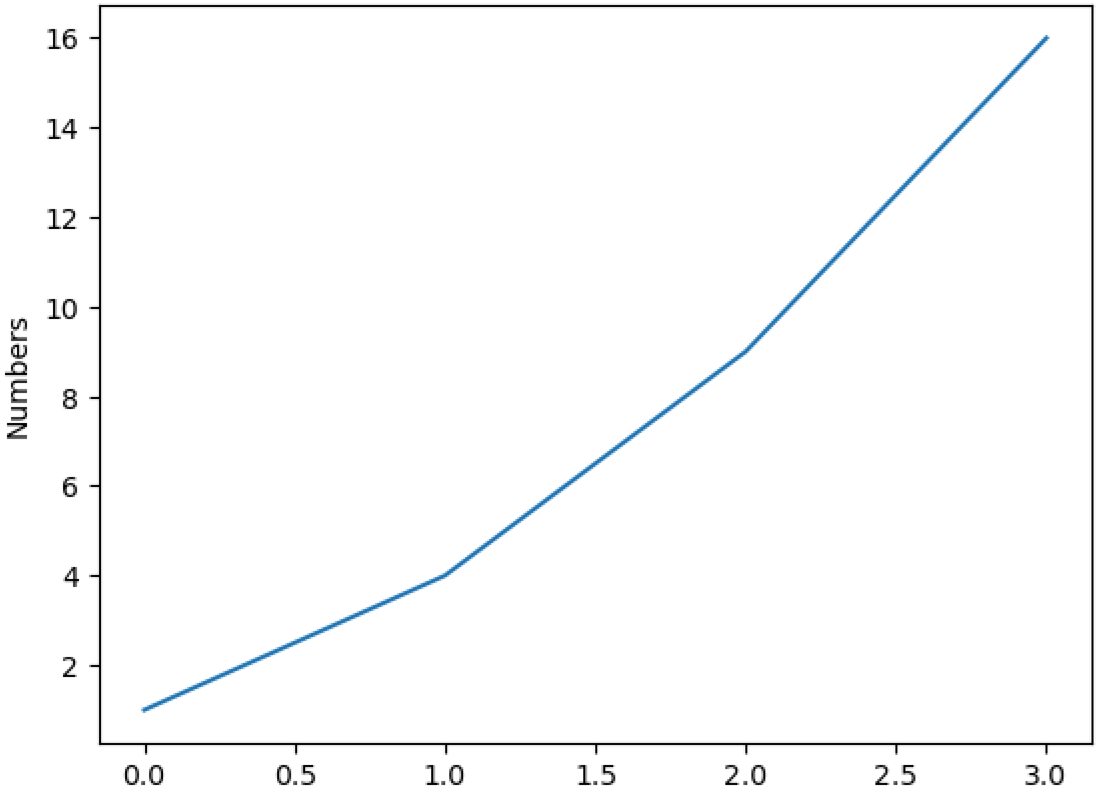


```
In [8]: plt.plot([1, 2, 3, 4])
        plt.ylabel('Numbers')
        plt.show()
```



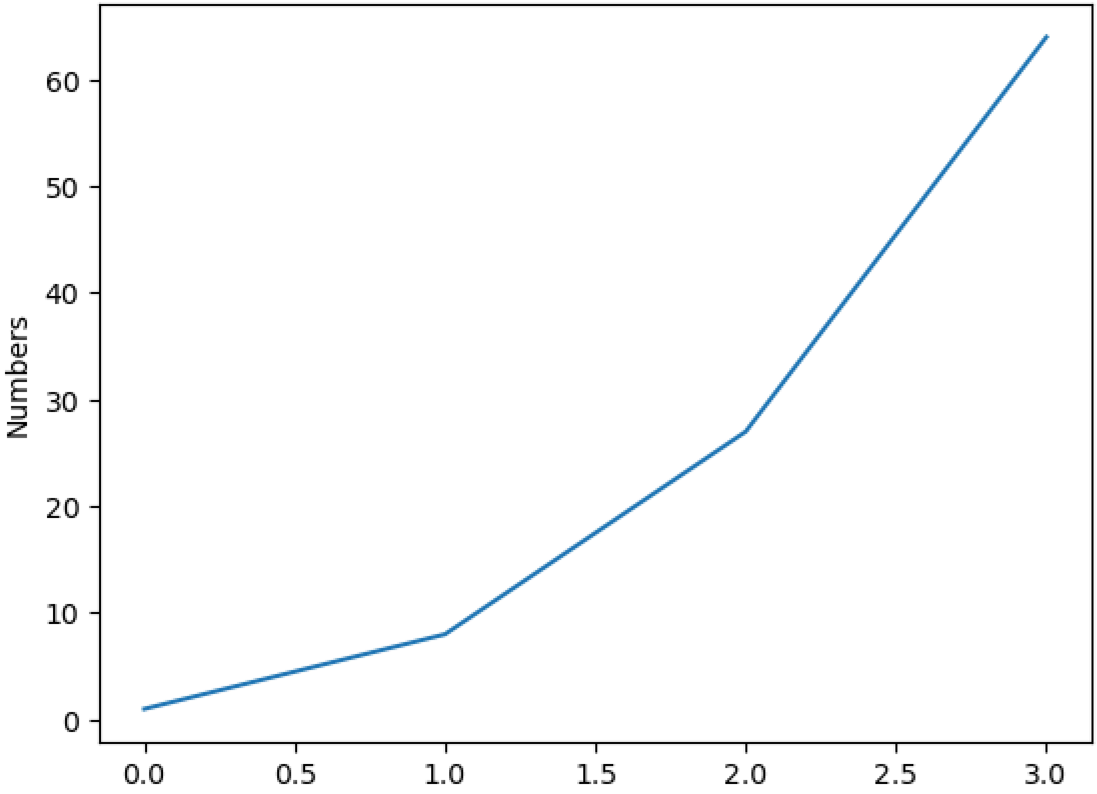
In [9]:

```
plt.plot([1, 4, 9, 16])
plt.ylabel('Numbers')
plt.show()
```

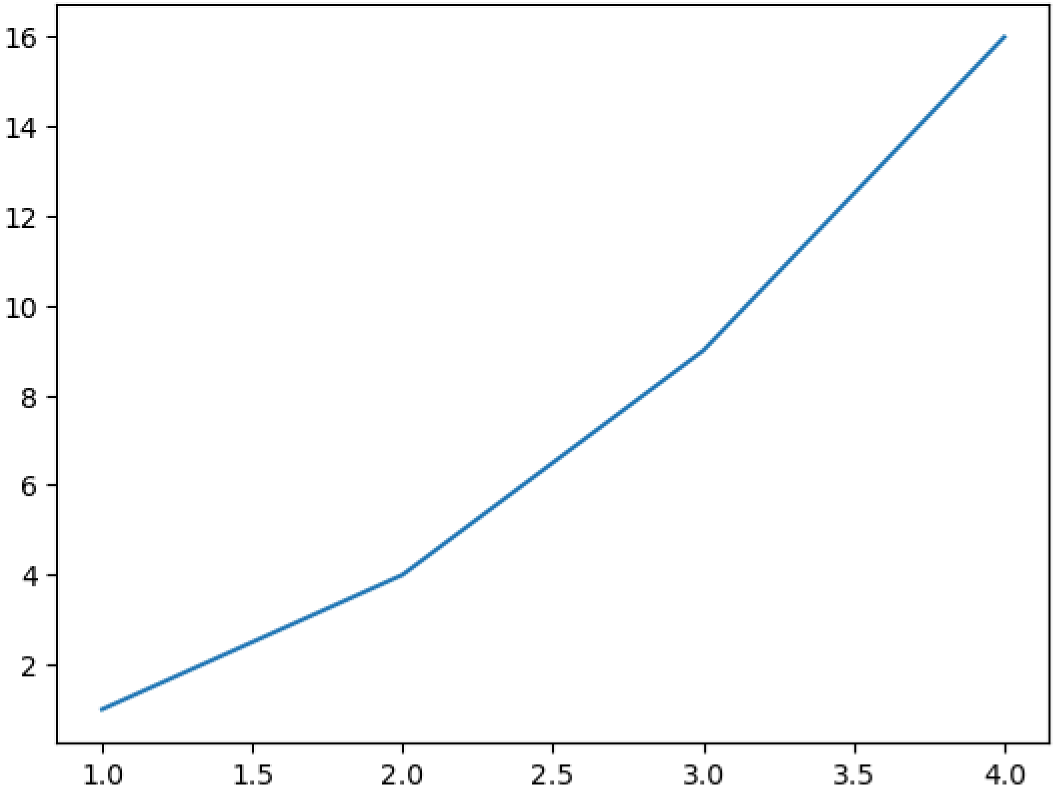


In [10]:

```
plt.plot([1, 8, 27, 64])
plt.ylabel('Numbers')
plt.show()
```

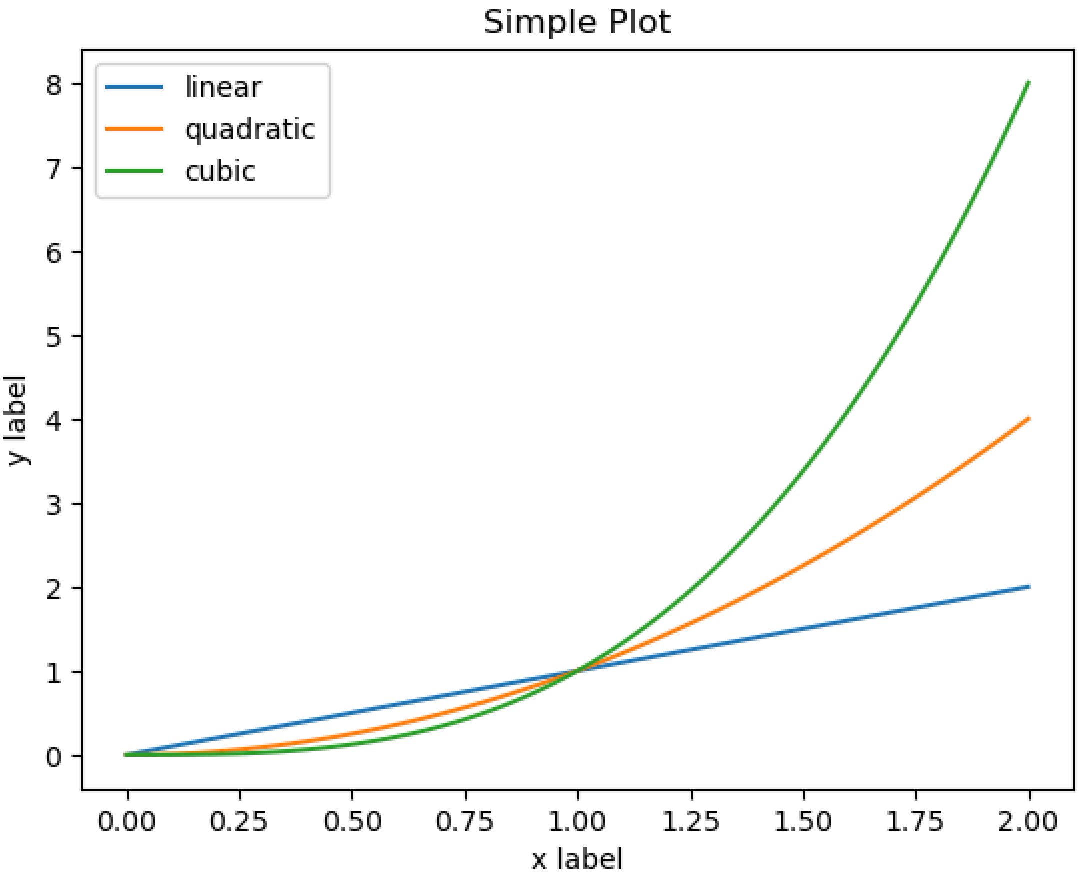


In [11]:
```python
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
plt.show()
```



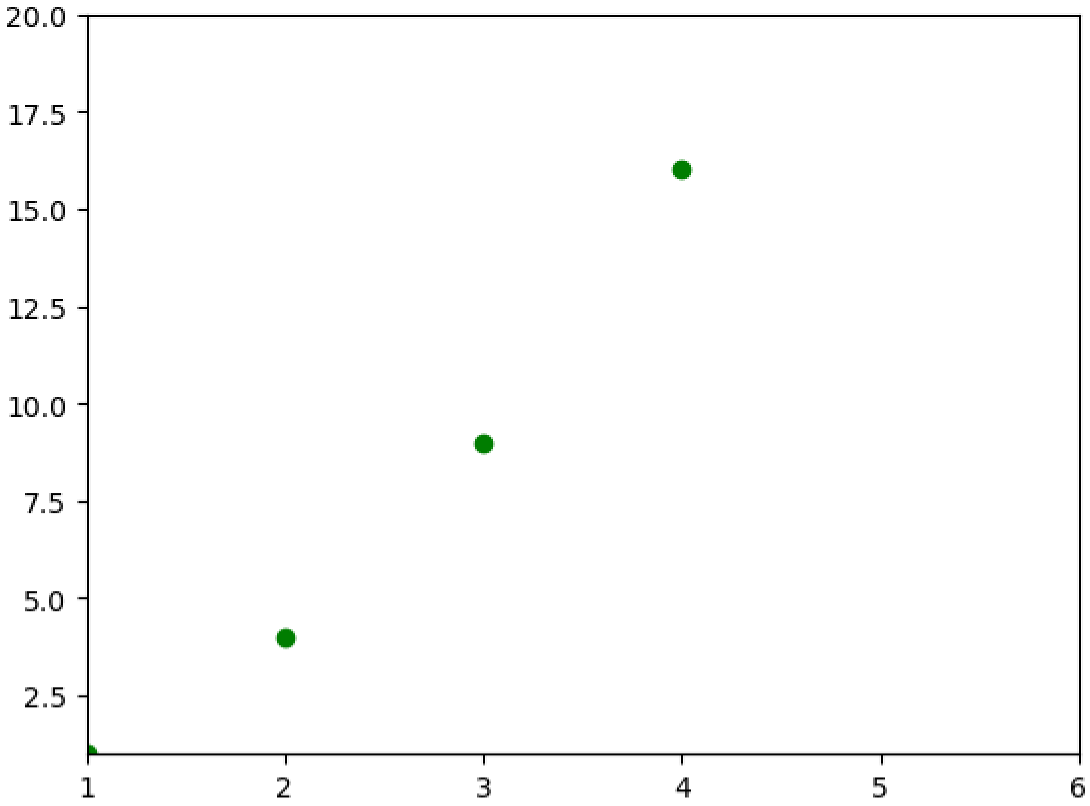
In [12]:

```
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('y label')

plt.title("Simple Plot")
plt.legend()
plt.show()
```



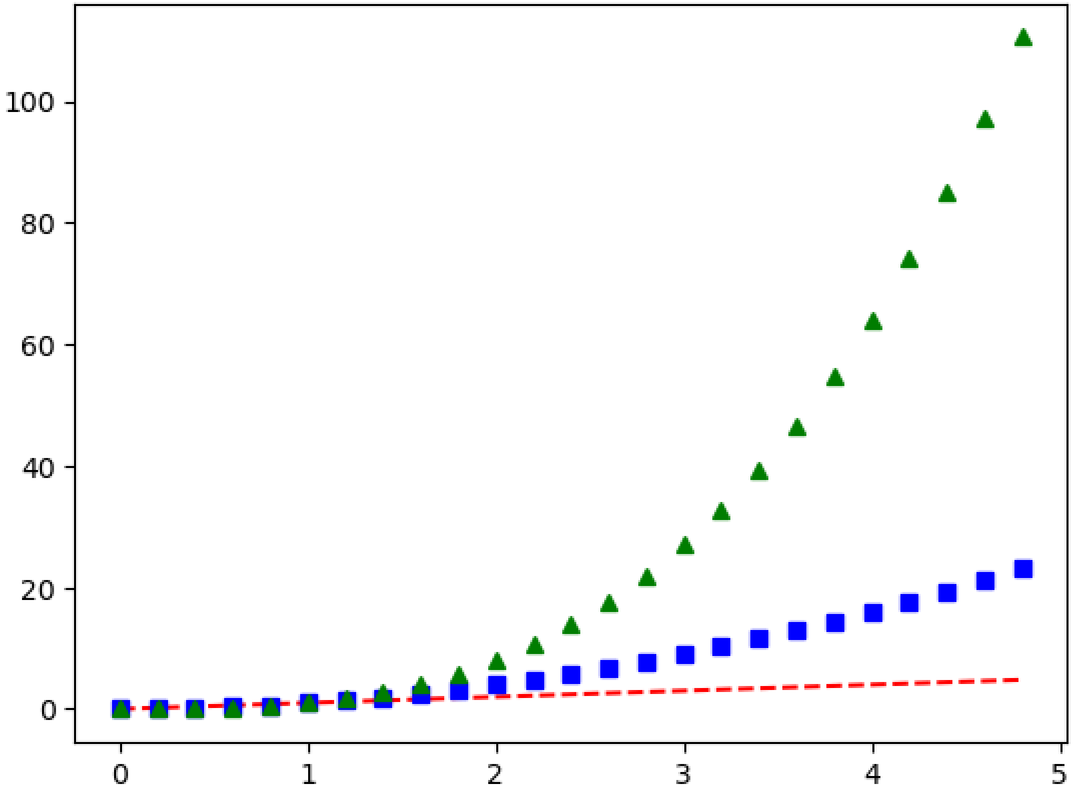
In [13]:

```
# Formatting the style of plot

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



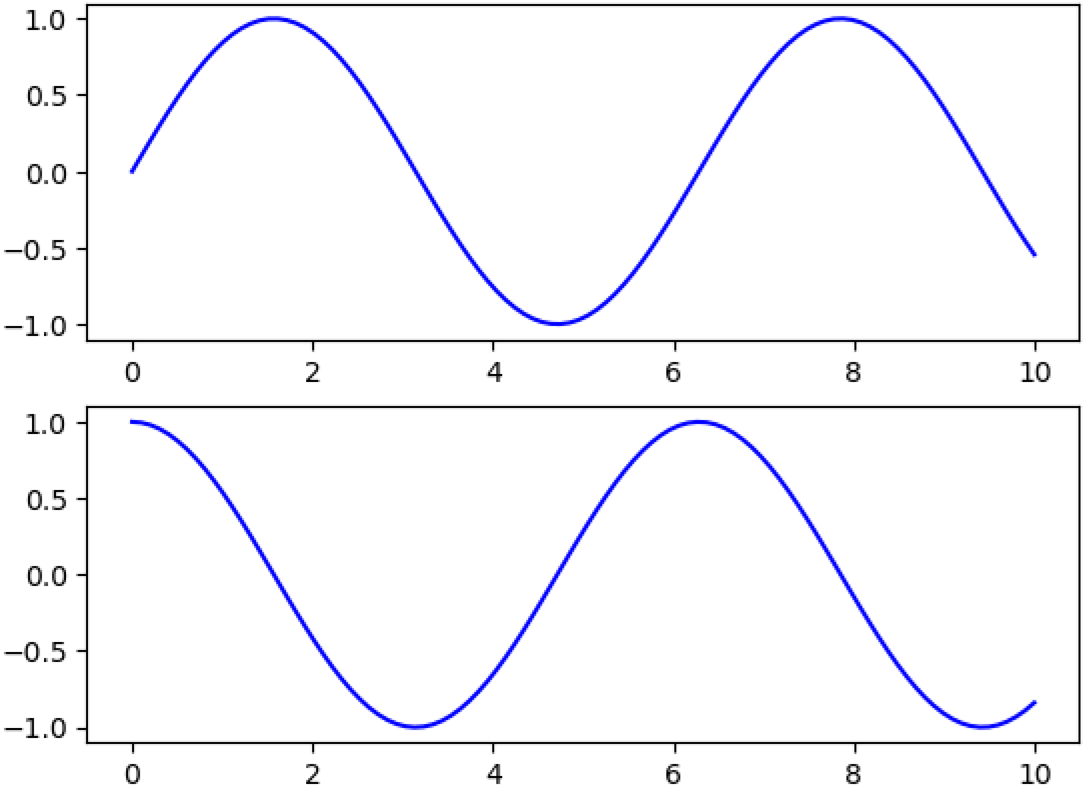
In [14]:

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

plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```



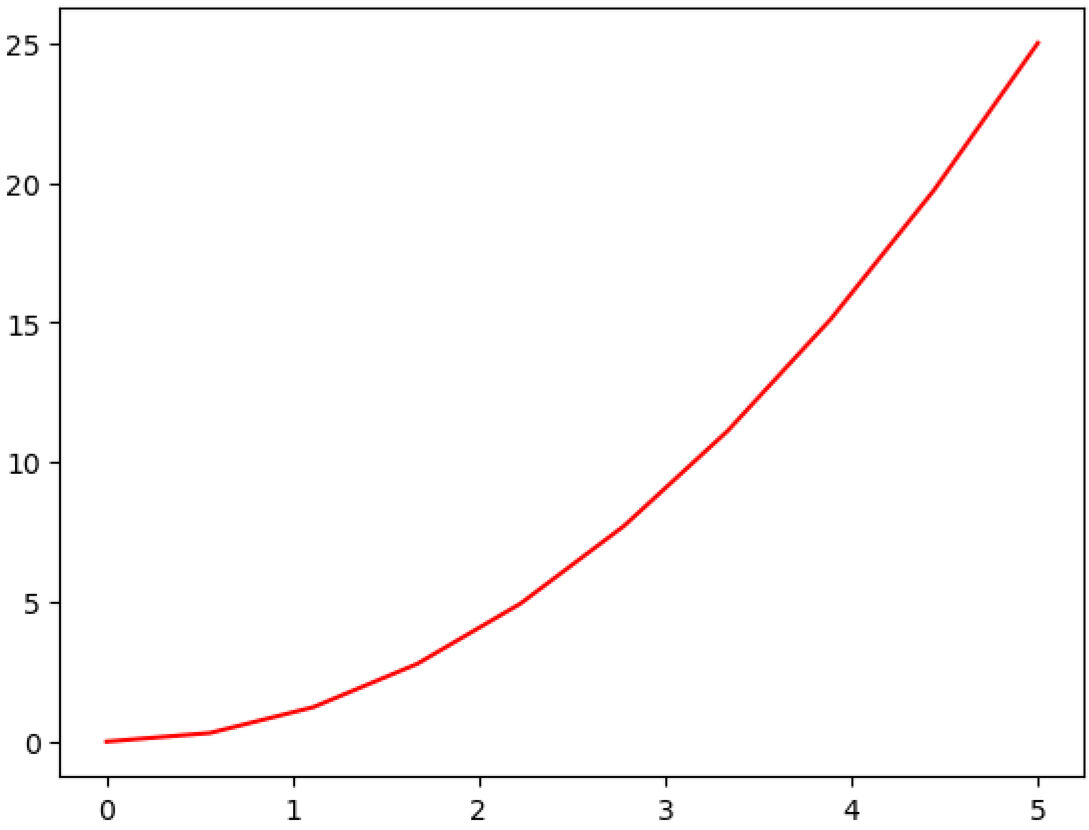
In [15]:

```
fig, ax = plt.subplots(2)

ax[0].plot(x1, np.sin(x1), 'b-')
ax[1].plot(x1, np.cos(x1), 'b-')
plt.show()
```

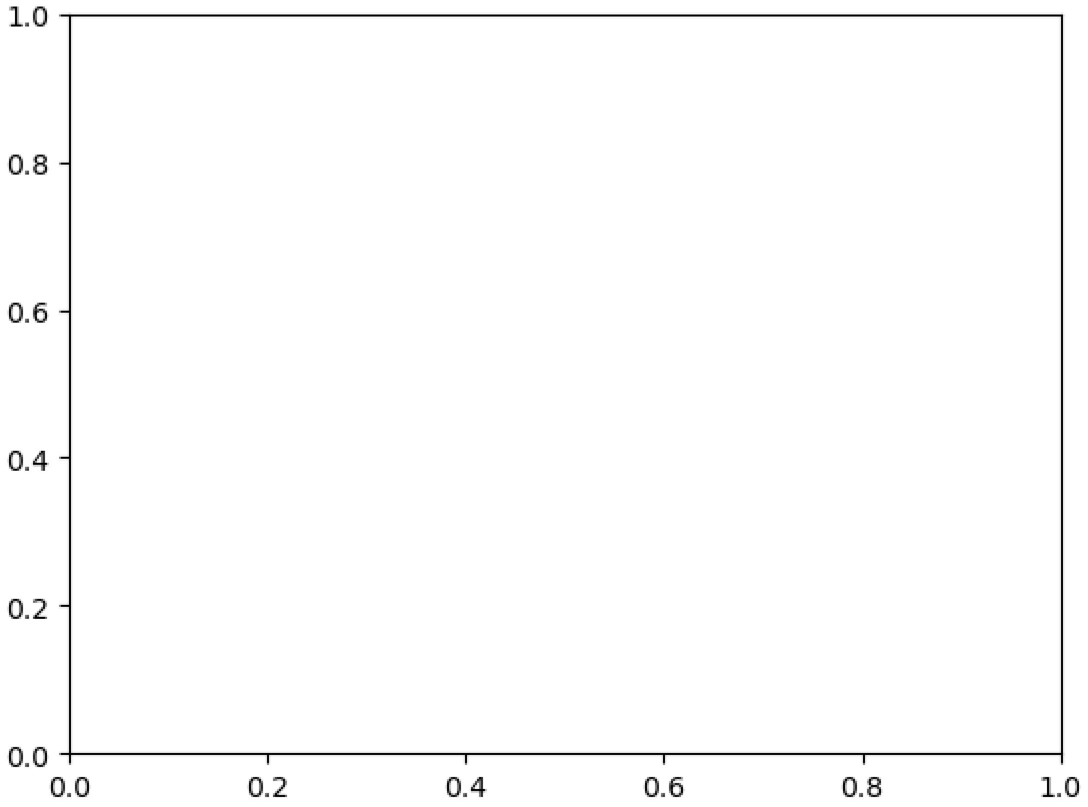


In [16]:

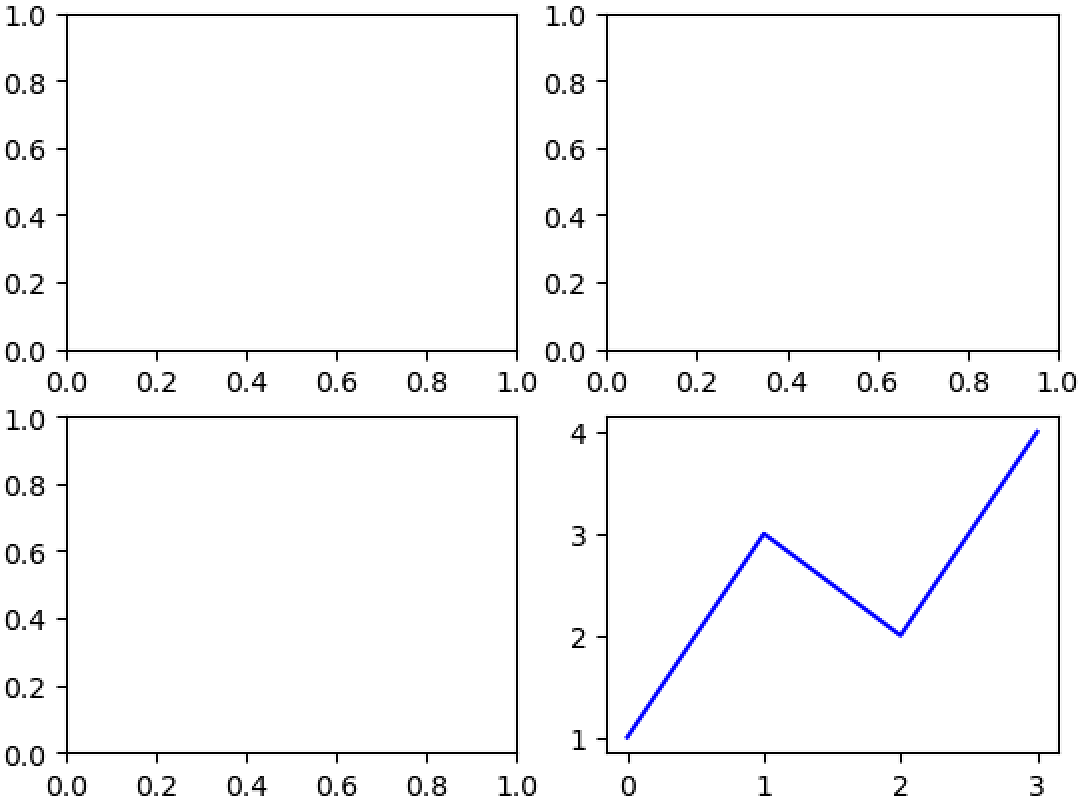
```
fig = plt.figure()

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')
plt.show()
```



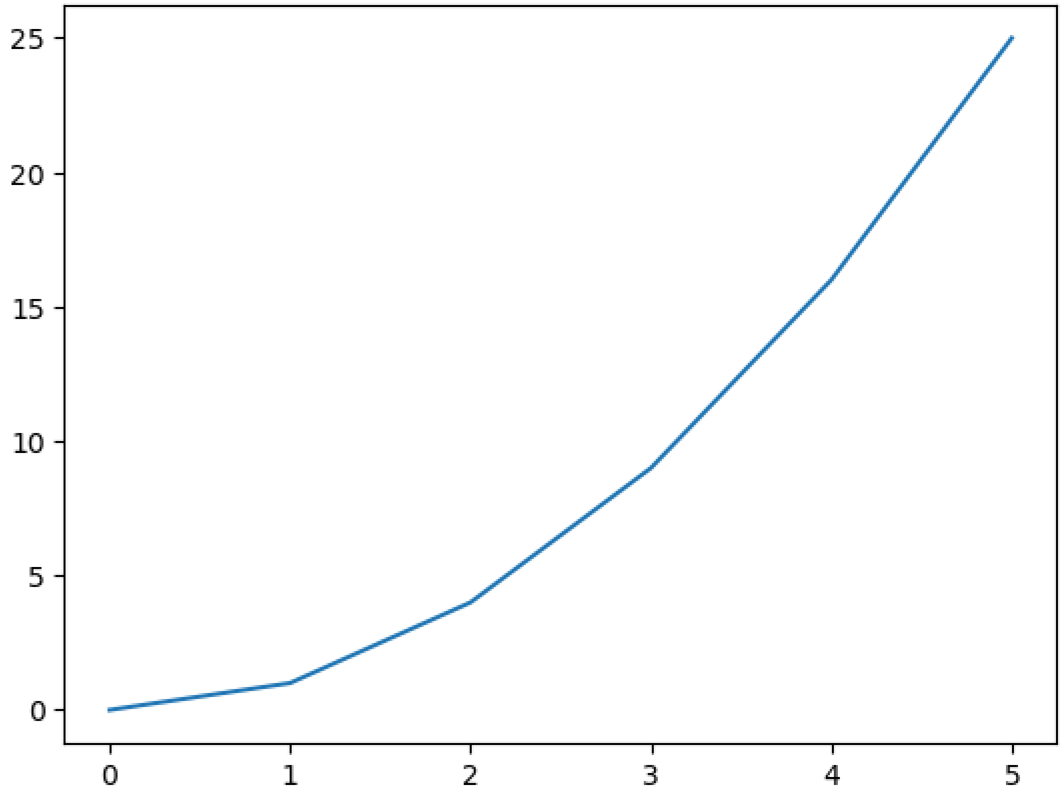
In [17]:
```
fig = plt.figure()
ax = plt.axes()
plt.show()
```



In [18]:
```
# Figure and Subplots

fig = plt.figure()
ax1 = fig.add_subplot(2, 2, 1)
ax2 = fig.add_subplot(2, 2, 2)
ax3 = fig.add_subplot(2, 2, 3)
ax4 = fig.add_subplot(2, 2, 4)
plt.plot([1, 3, 2, 4], 'b-')
plt.show()
```

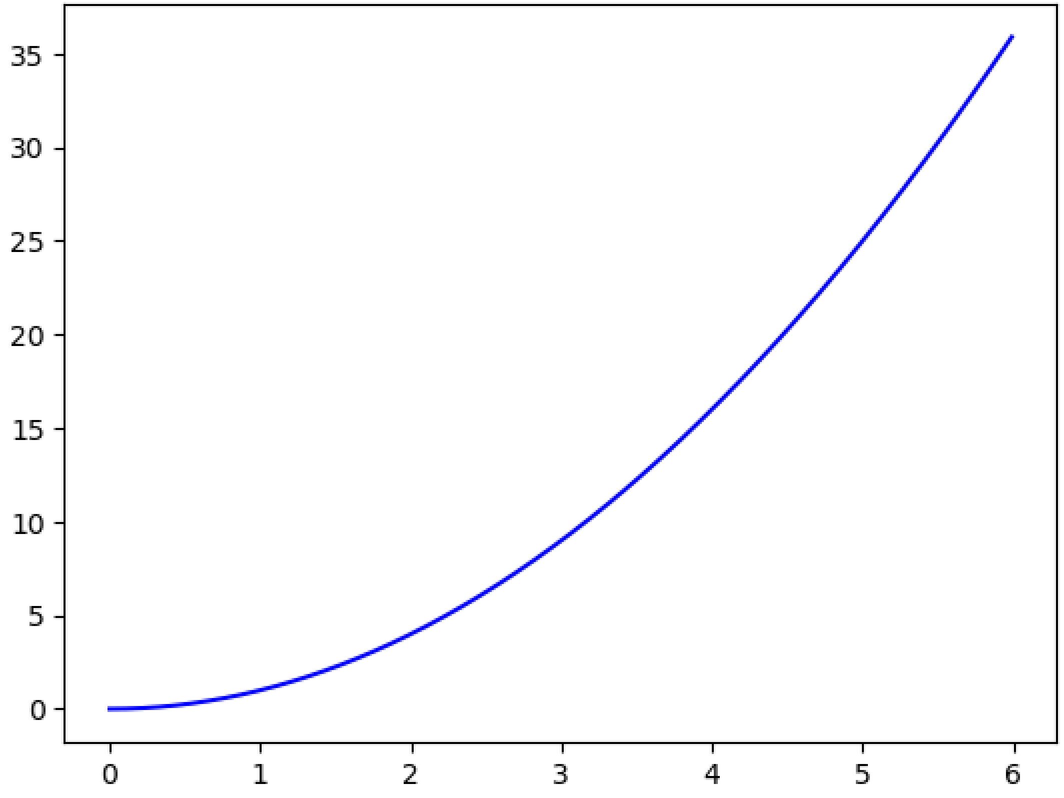


In [19]:
```
## 11. First plot with Matplotlib

x3 = np.arange(6)
plt.plot(x3, [xi**2 for xi in x3])
plt.show()
```



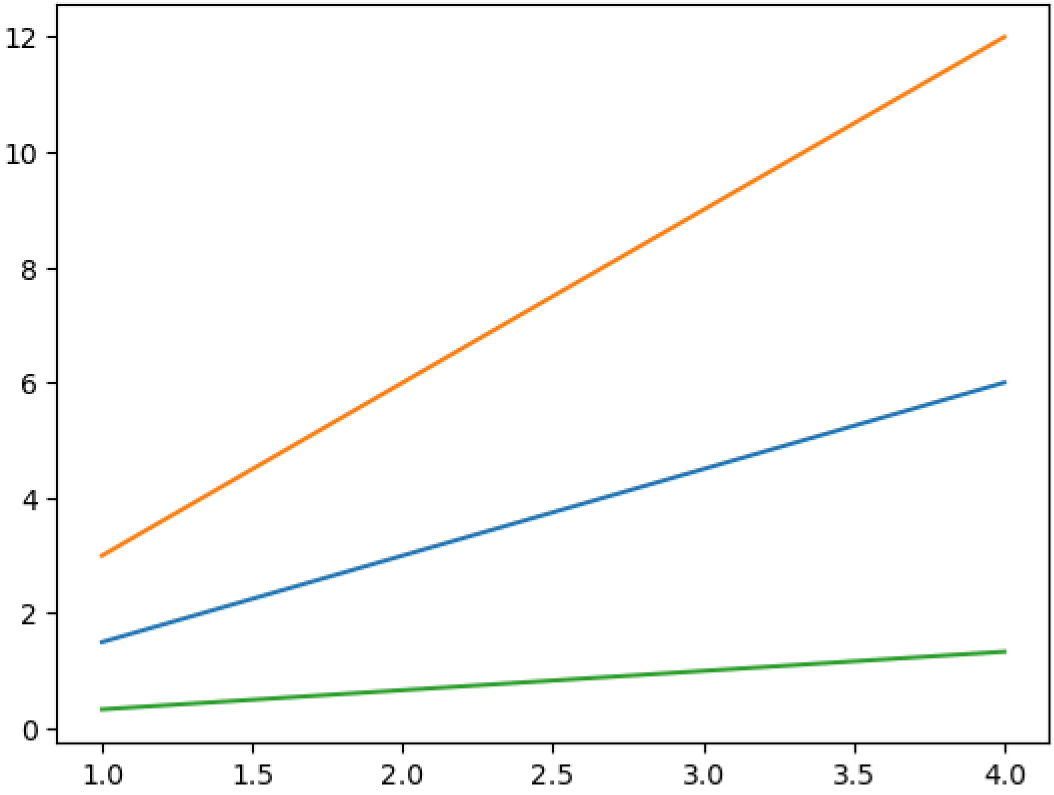
In [20]:

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



In [21]:

```
# Multiline Plots

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()
```



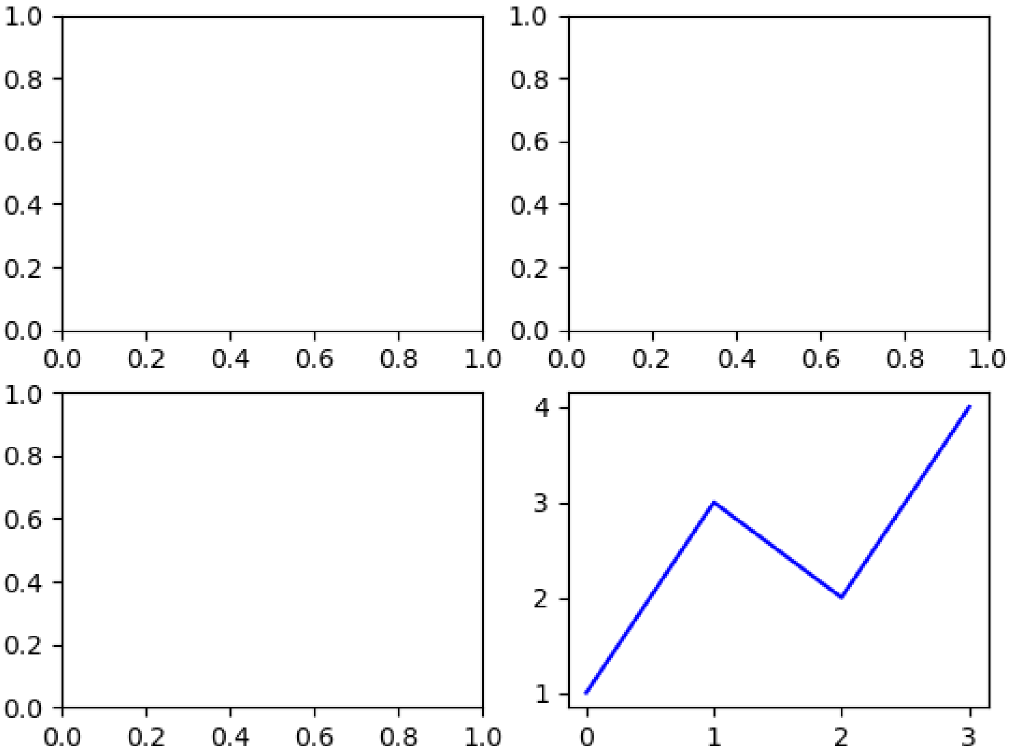
In [22]:

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

In [23]:

```
from IPython.display import Image
Image('plot1.png')
```

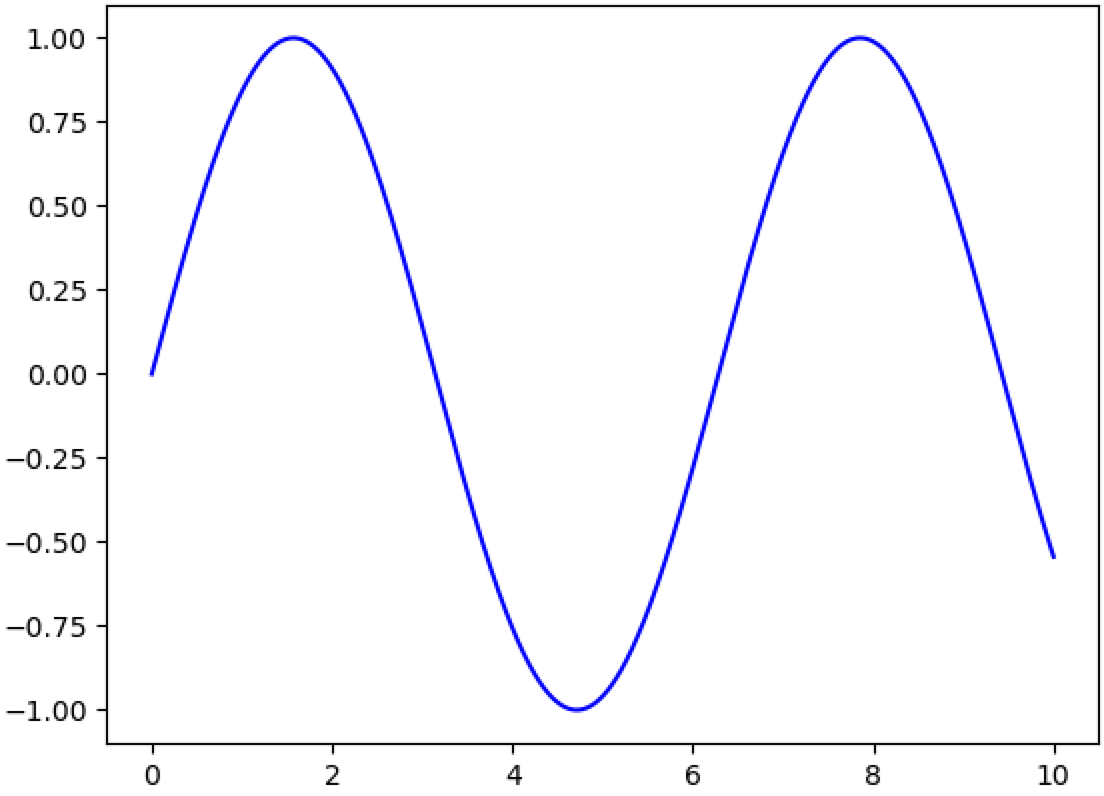
Out[23]:



In [24]:
```
fig.canvas.get_supported_filetypes()
```

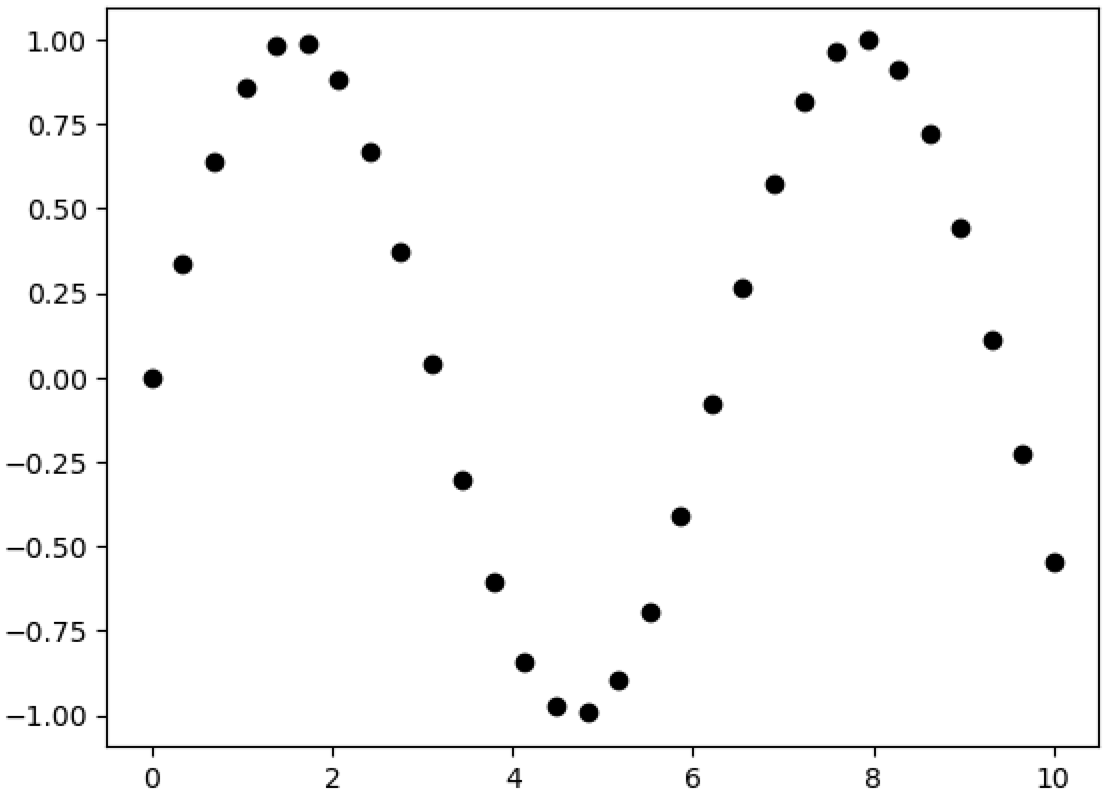
Out[24]:
```
{'eps': 'Encapsulated Postscript',
 'jpg': 'Joint Photographic Experts Group',
 'jpeg': 'Joint Photographic Experts Group',
 'pdf': 'Portable Document Format',
 'pgf': 'PGF code for LaTeX',
 'png': 'Portable Network Graphics',
 'ps': 'Postscript',
 'raw': 'Raw RGBA bitmap',
 'rgba': 'Raw 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 [25]:
```
# Line Plot

fig = plt.figure()
ax = plt.axes()
x5 = np.linspace(0, 10, 1000)
ax.plot(x5, np.sin(x5), 'b-')
plt.show()
```

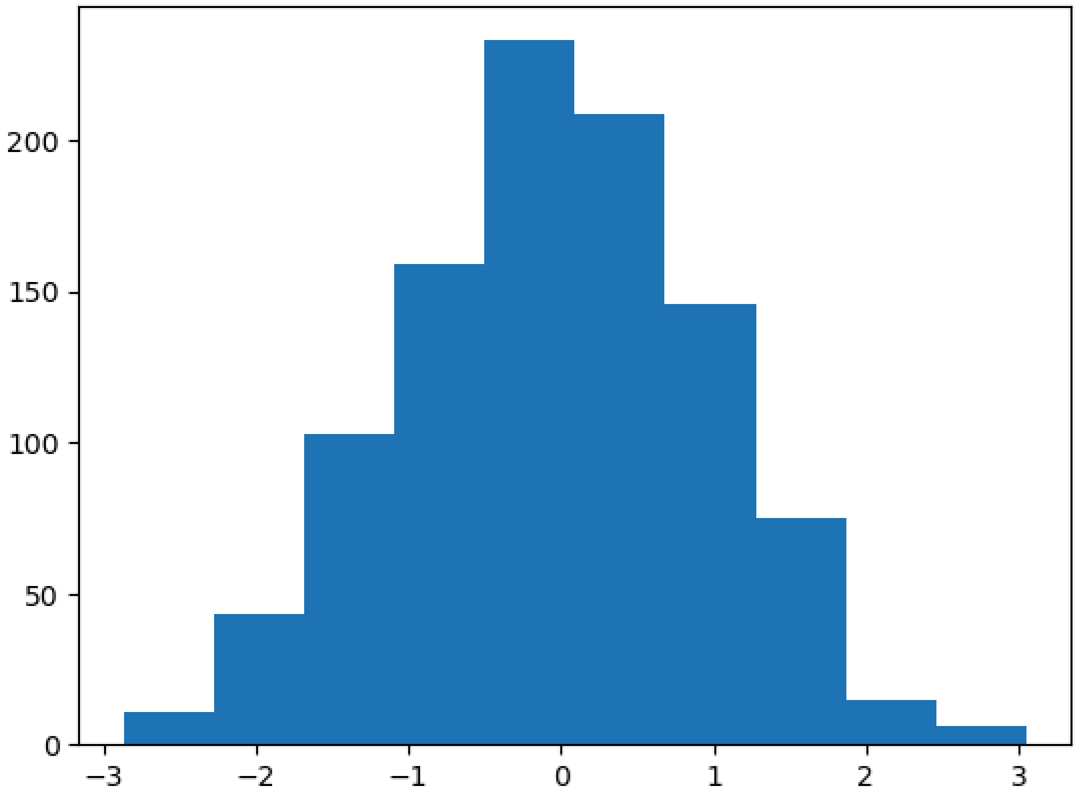


In [26]:

```
# Scatter Plot

x7 = np.linspace(0, 10, 30)
y7 = np.sin(x7)
plt.plot(x7, y7, 'o', color = 'black');
plt.show()
```

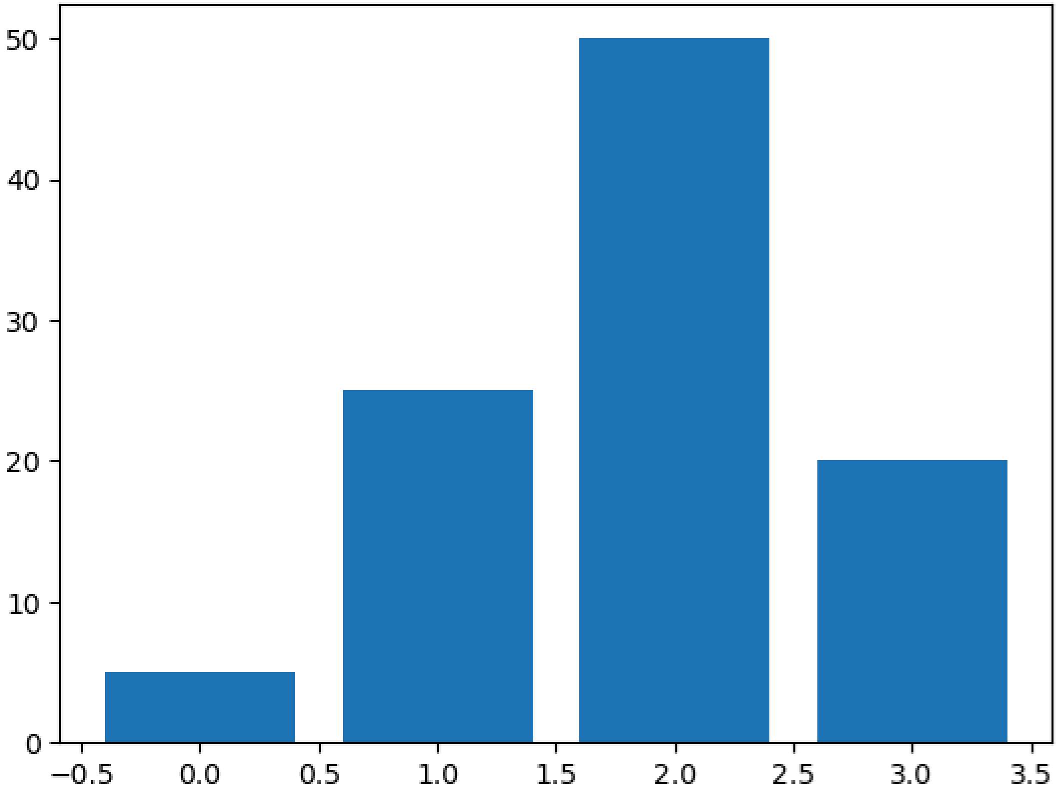


In [27]:
```
# Histogram

data1 = np.random.randn(1000)
plt.hist(data1)
plt.show()
```



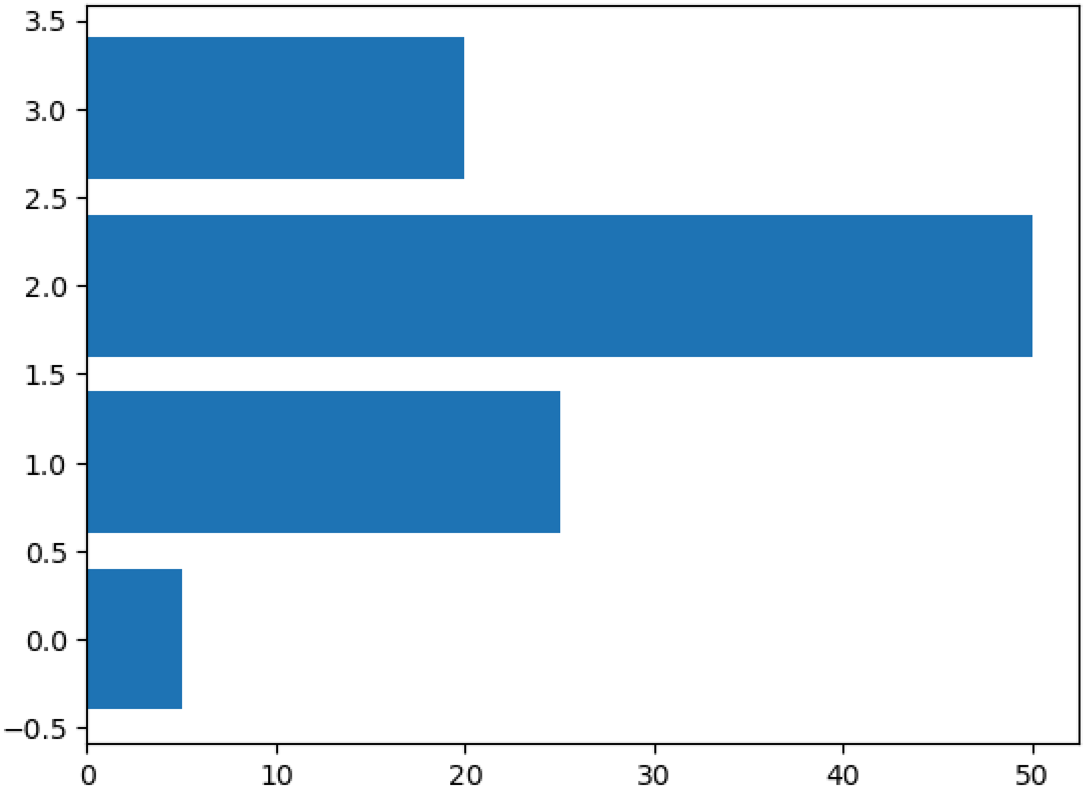
In [28]:

```
# Bar Chart

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



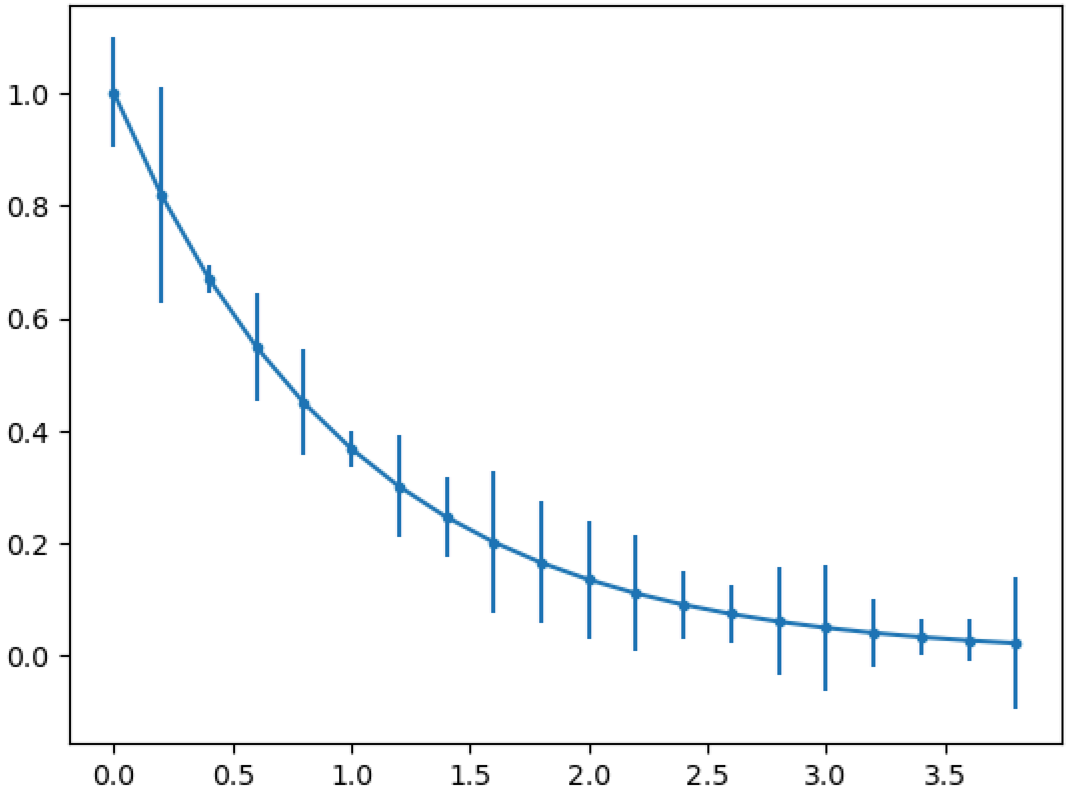
In [29]:

```
# Histogram Bar Chart

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



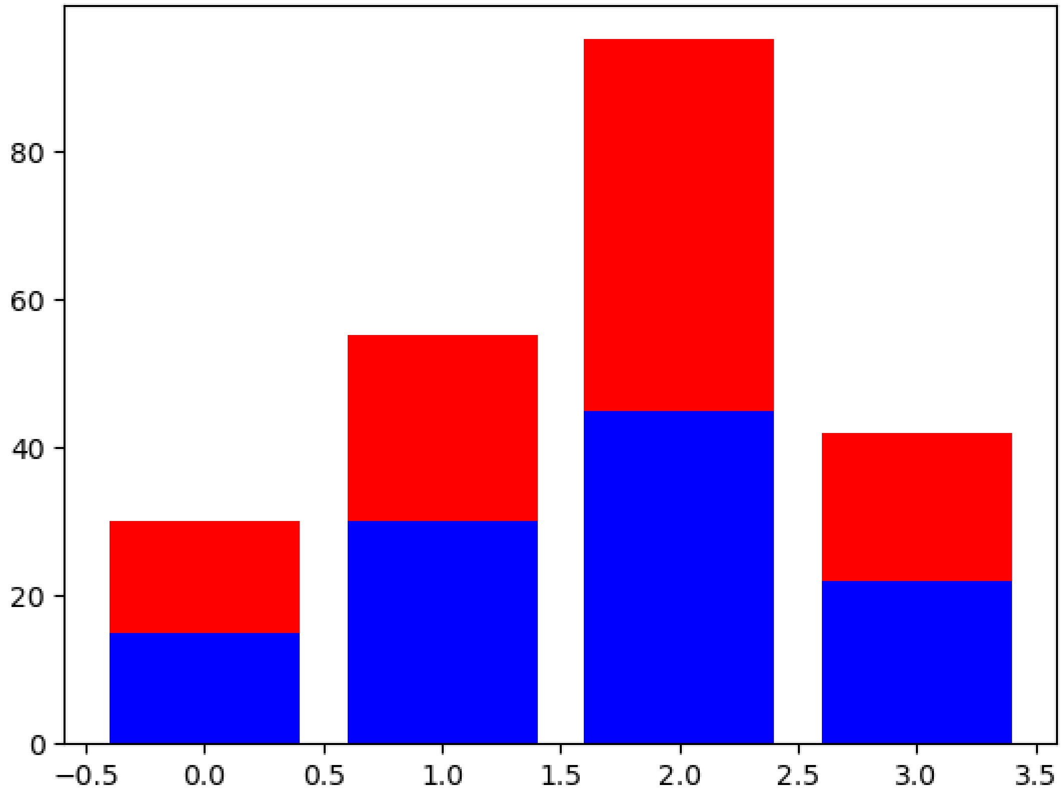
In [30]:

```
# Error Bar Chart

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()
```



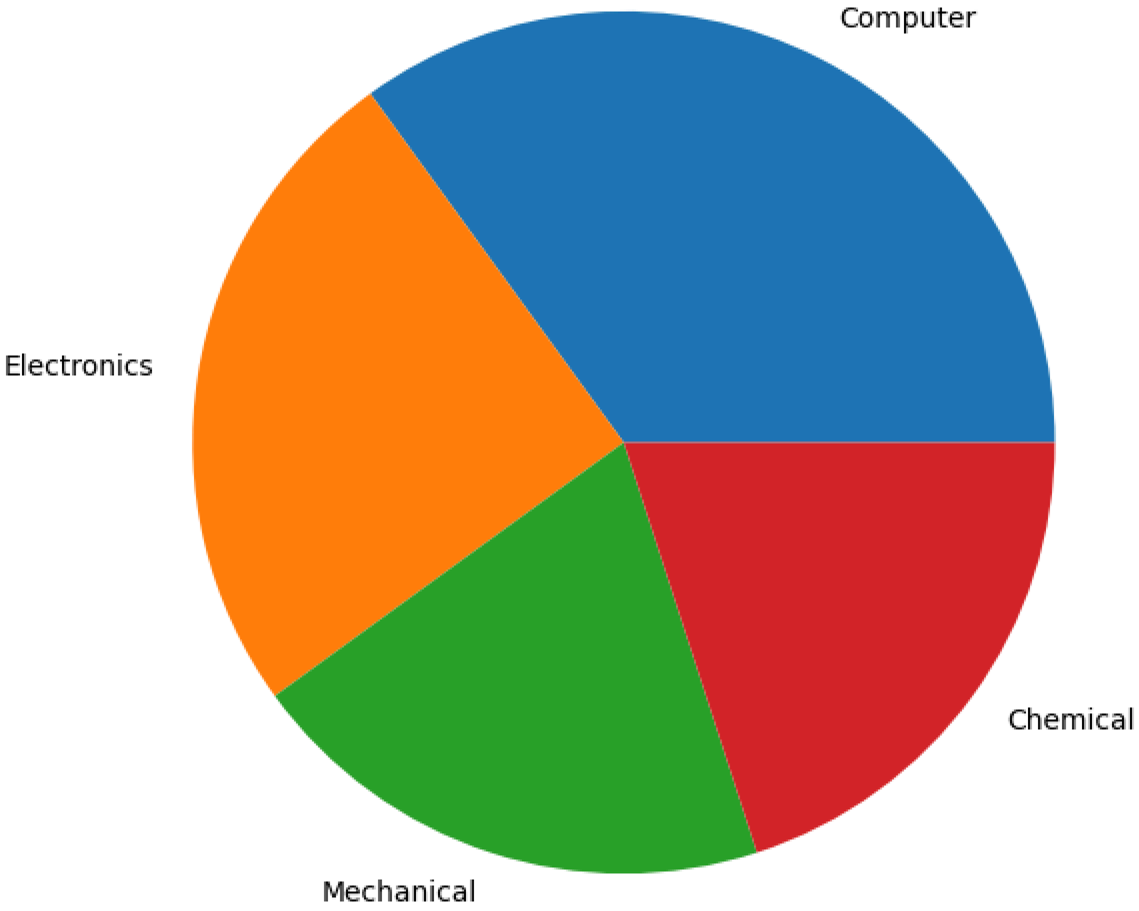
In [31]:

```
# Stacked Bar Chart

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()
```



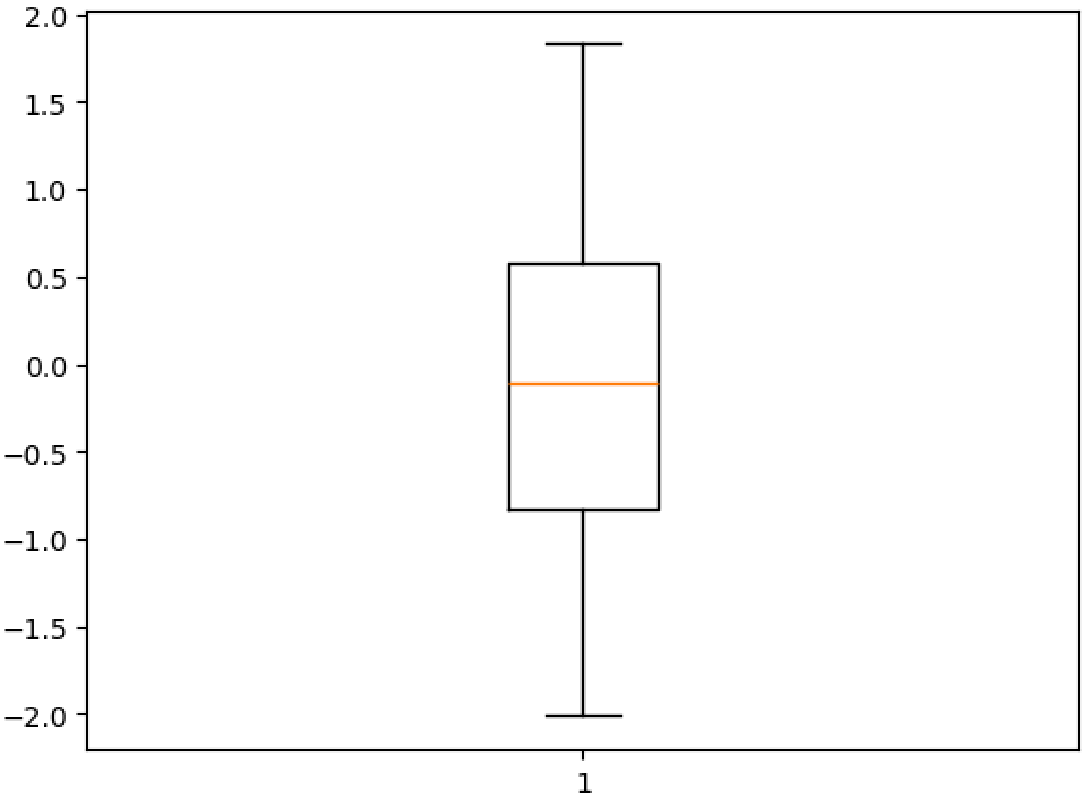
In [32]:

```
# Pie Chart

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



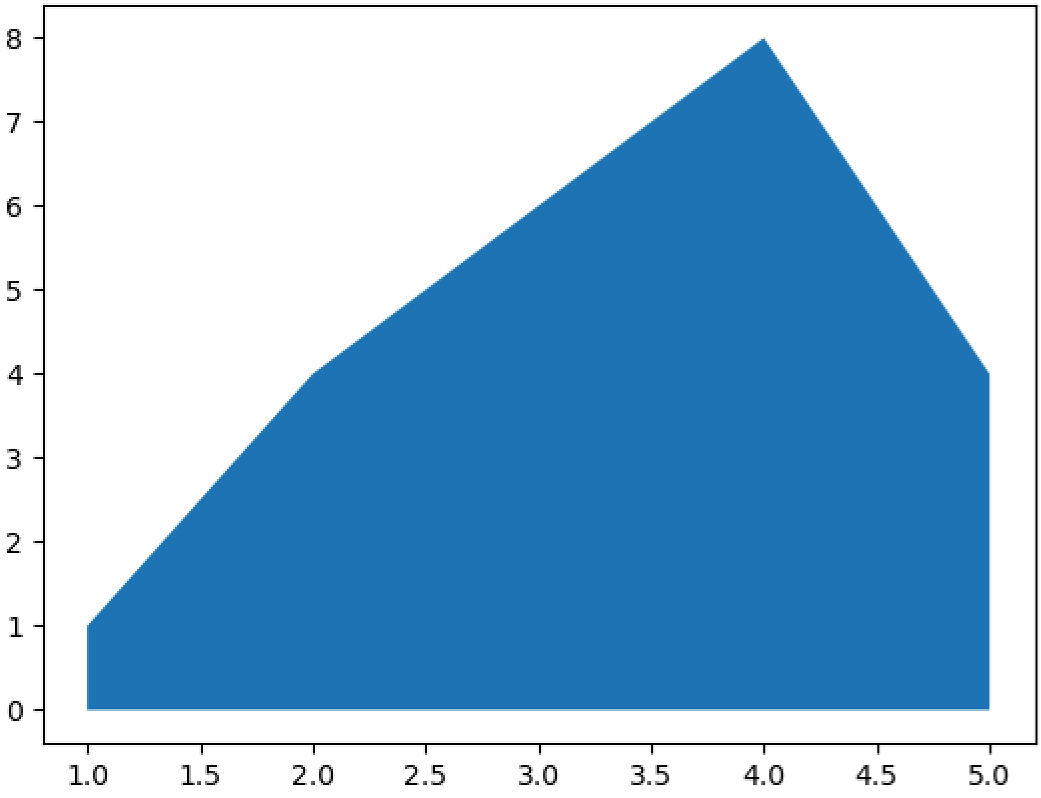
In [33]:

```
# Boxplot

data3 = np.random.randn(100)
plt.boxplot(data3)
plt.show();
```



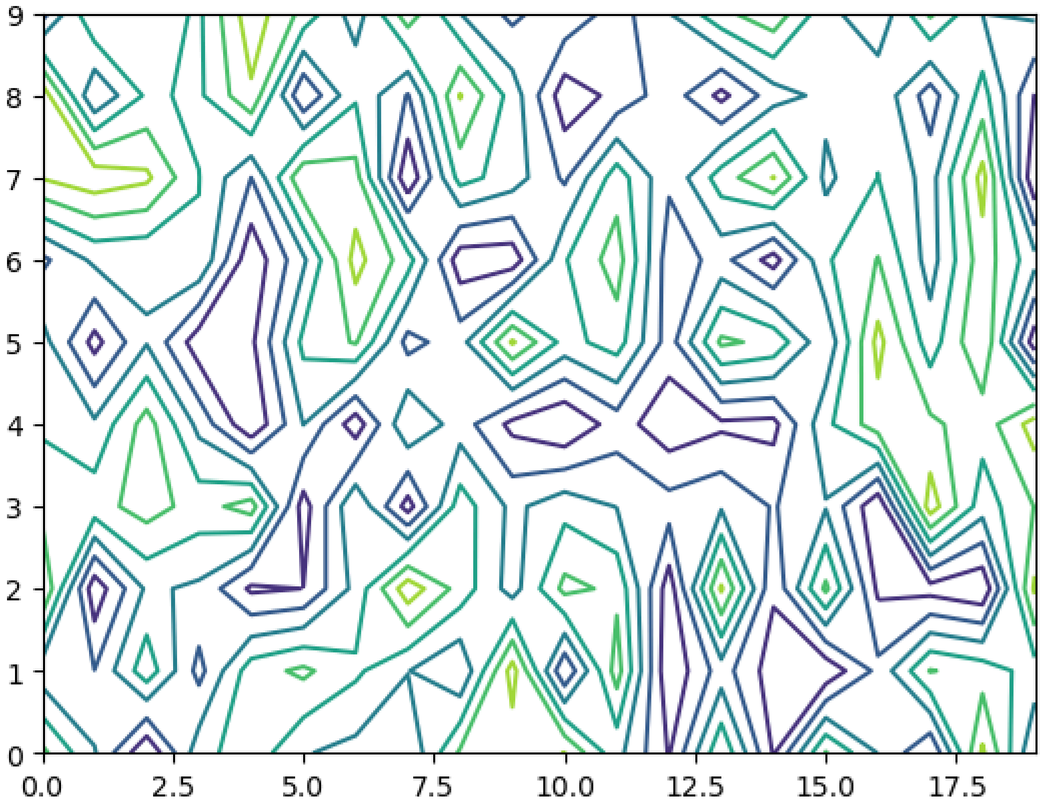
In [34]:

```
# Aera Chart

x12 = range(1, 6)
y12 = [1, 4, 6, 8, 4]

plt.fill_between(x12, y12)
plt.show()
```



In [35]:

```
# Contour Plot

matrix1 = np.random.rand(10, 20)
cp = plt.contour(matrix1)
plt.show()
```



In [36]:

```
# Styles with Matplotlib Plots

print(plt.style.available)
```

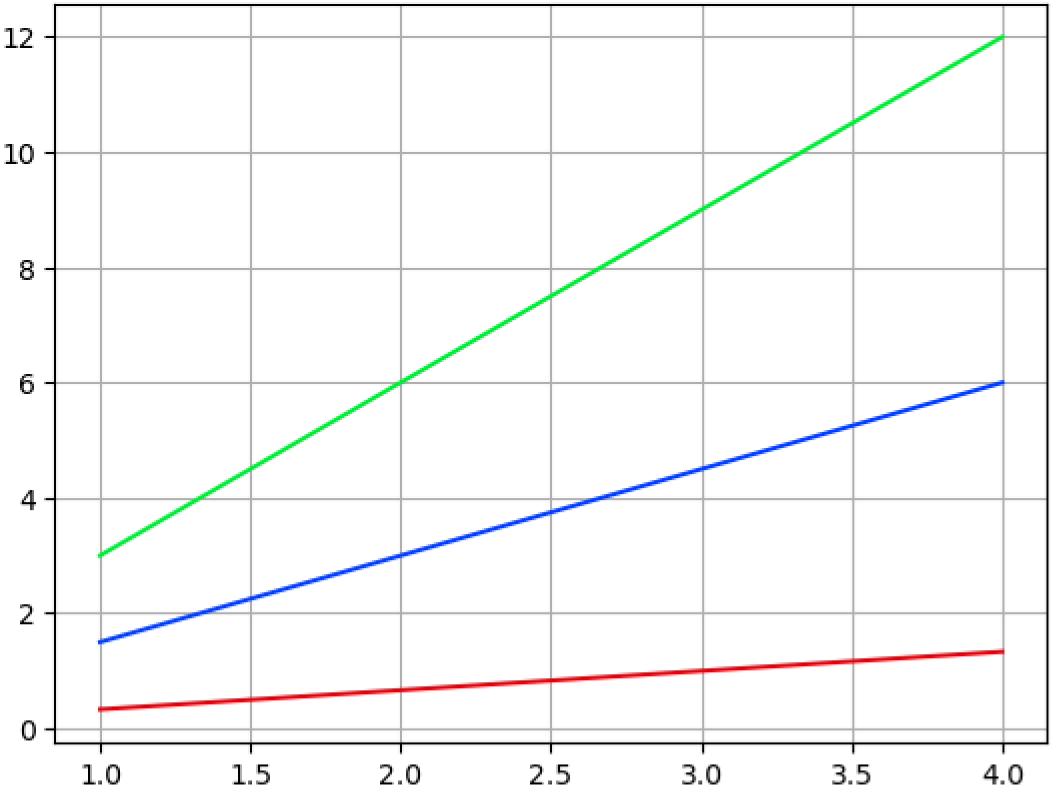
```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'bm
h', 'classic', 'dark_background', 'fast', 'fivethirtyeight', 'ggplot', 'grayscale', 's
eaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblind', 'seaborn-v0_8-dark',
'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-deep', 'seaborn-v0
_8-muted', 'seaborn-v0_8-notebook', 'seaborn-v0_8-paper', 'seaborn-v0_8-pastel', 'seab
orn-v0_8-poster', 'seaborn-v0_8-talk', 'seaborn-v0_8-ticks', 'seaborn-v0_8-white', 'se
aborn-v0_8-whitegrid', 'tableau-colorblind10']
```

In [37]:

```
plt.style.use('seaborn-v0_8-bright')
```

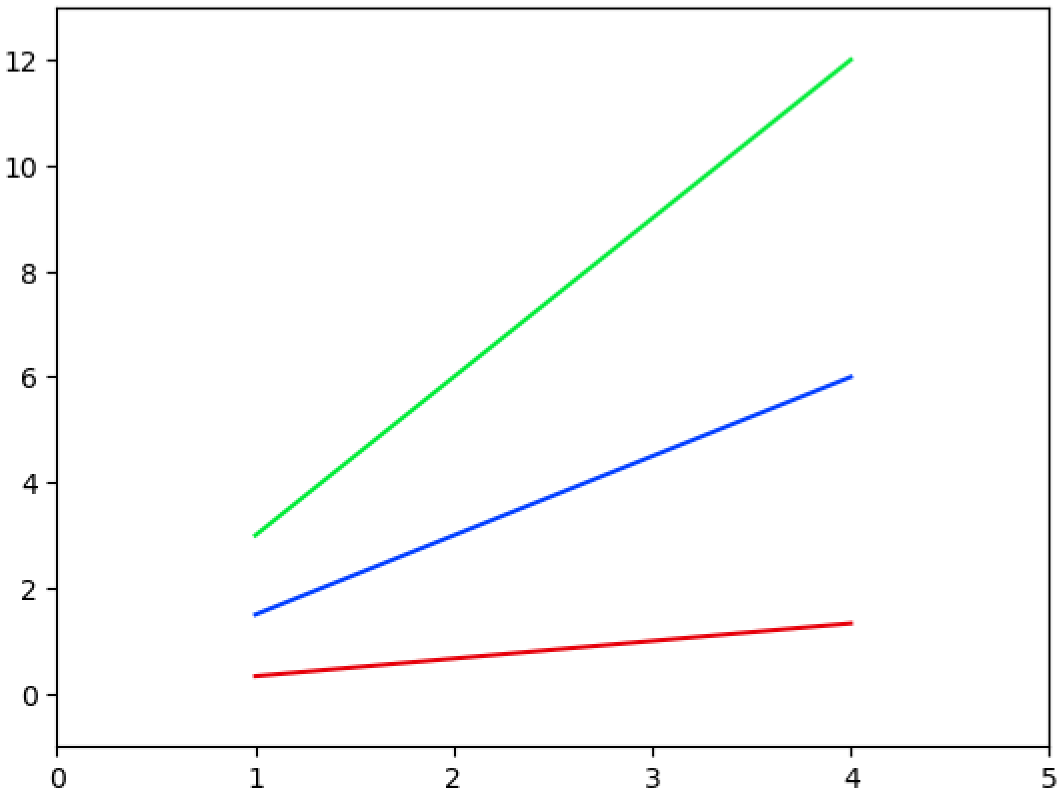
In [38]:

```
# Adding a Grid

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 [39]:

```
# Handling axes

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 [40]:

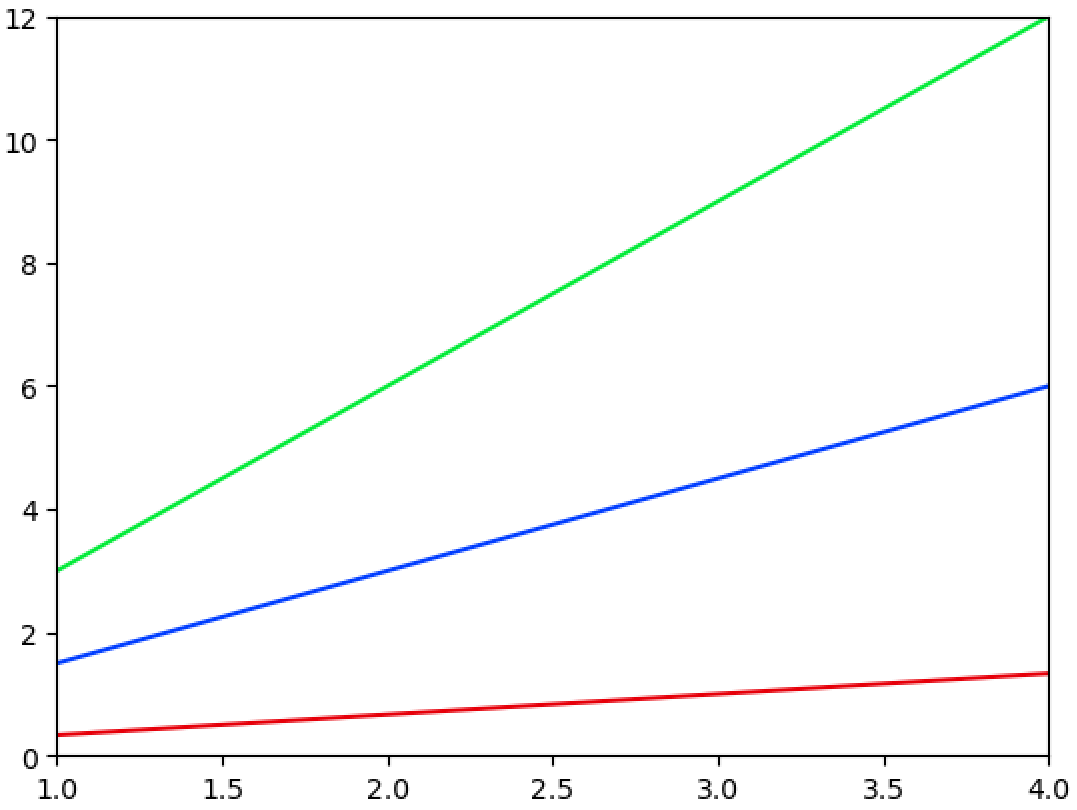
```
x15 = np.arange(1, 5)
plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
plt.xlim([1.0, 4.0])
plt.ylim([0.0, 12.0])
```

Out[40]:

```
(0.0, 12.0)
```

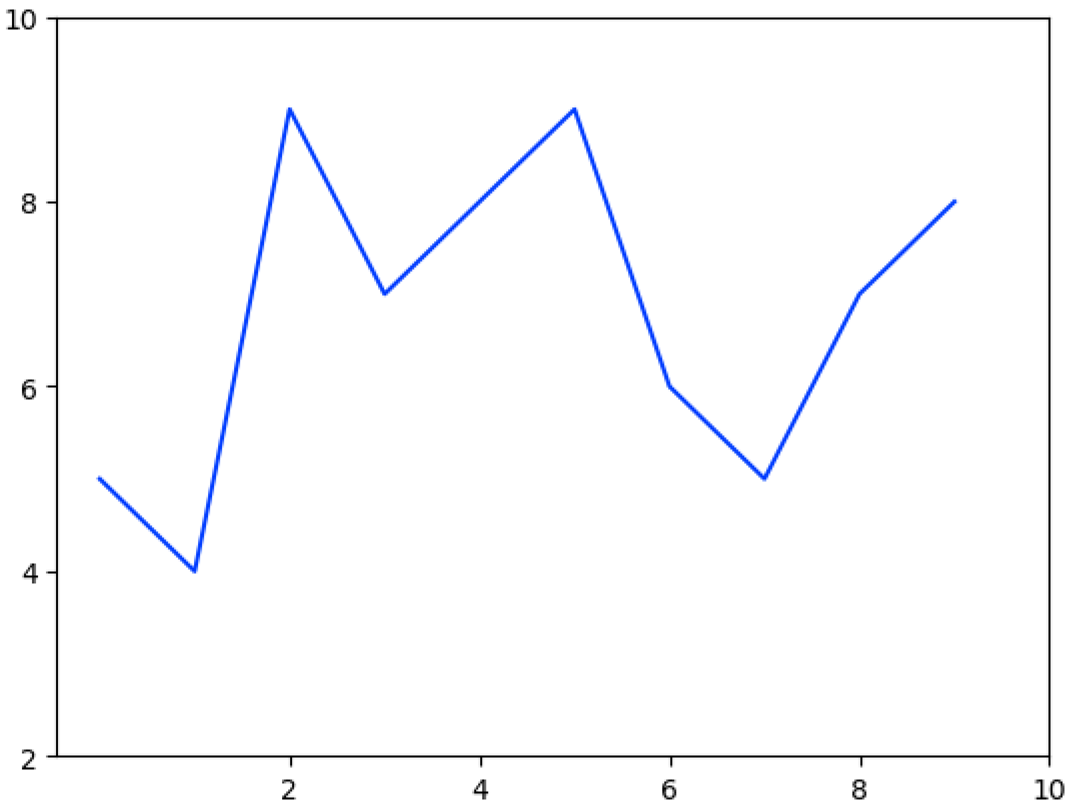
In [41]:

```
plt.show()
```



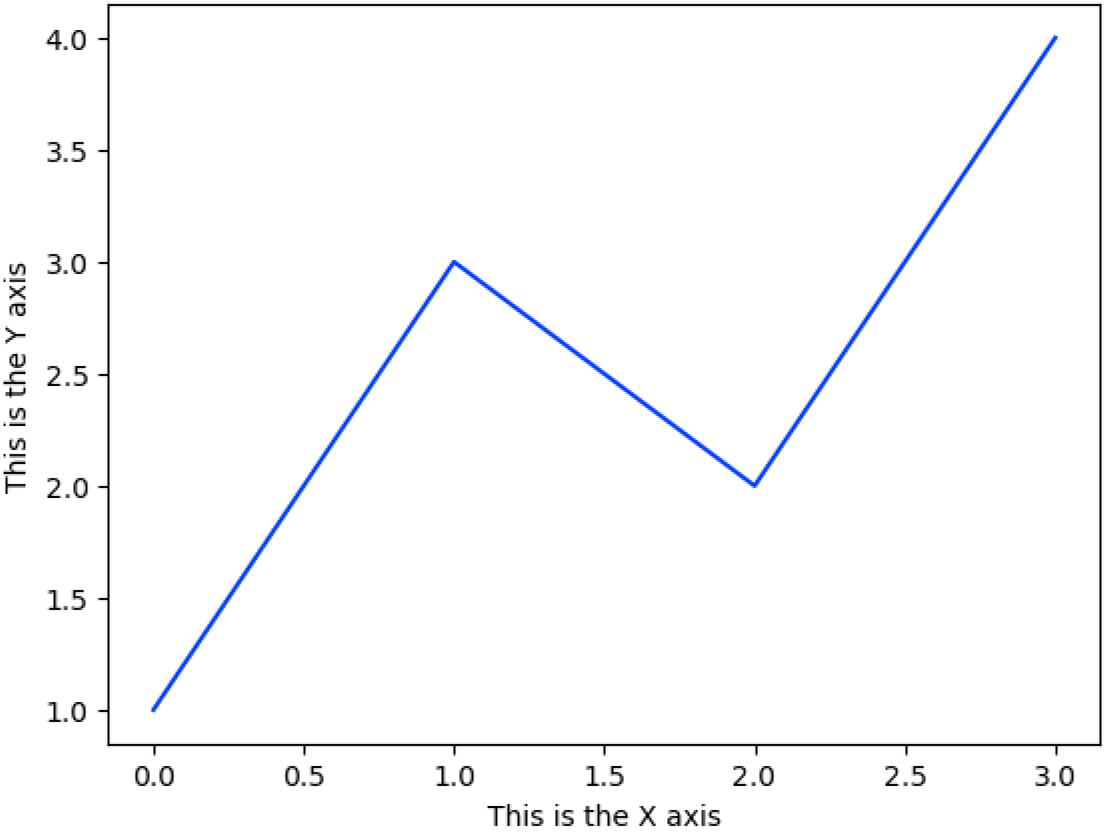
In [42]:

```
# Handling X and Y ticks

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()
```



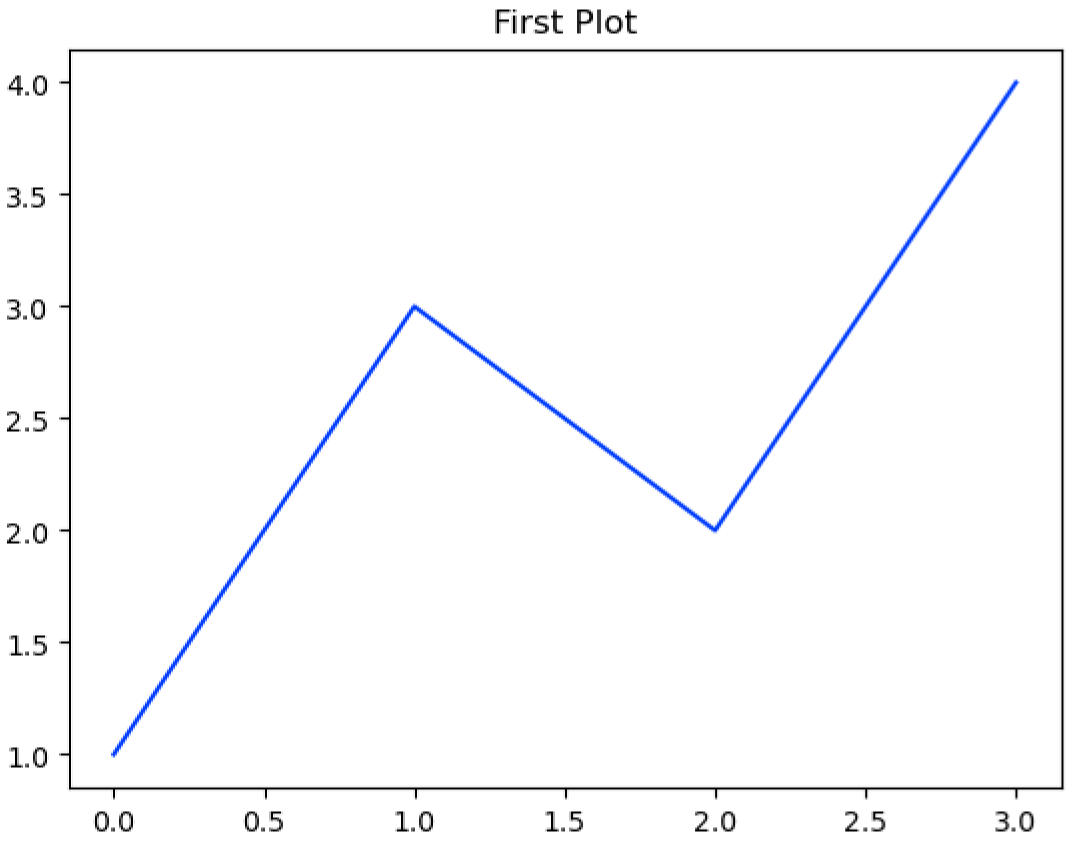
In [43]:

```
# Adding Lables

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

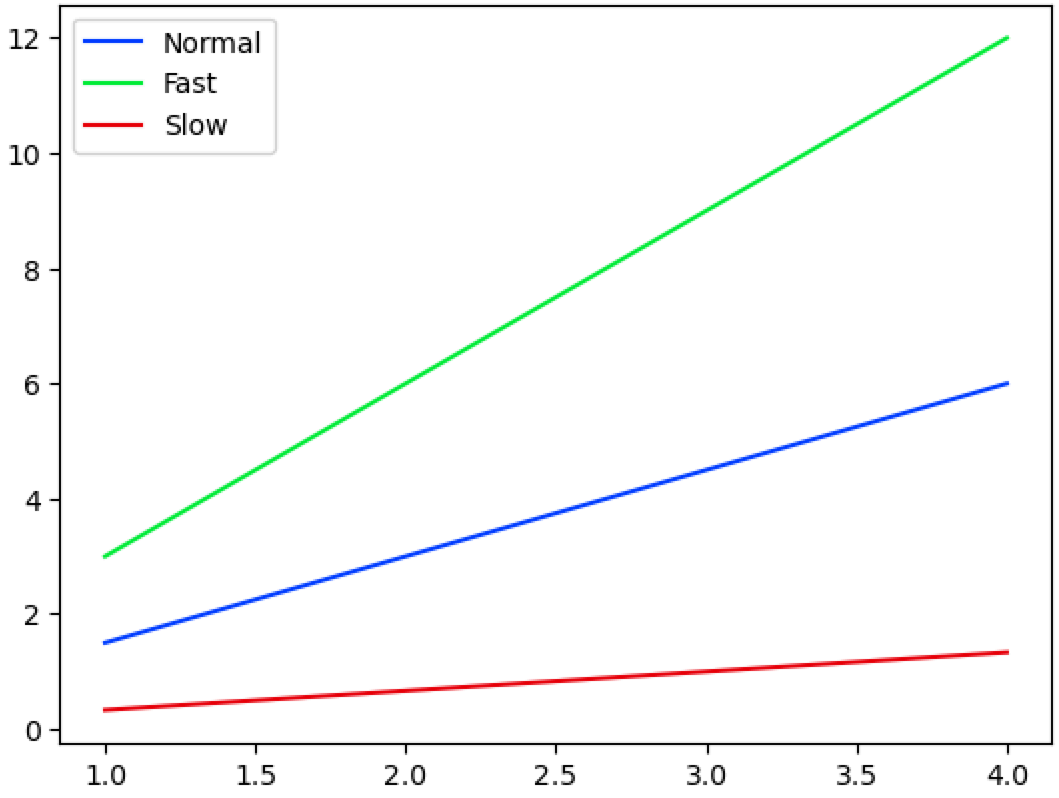


In [44]:

```
# Adding a Title

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

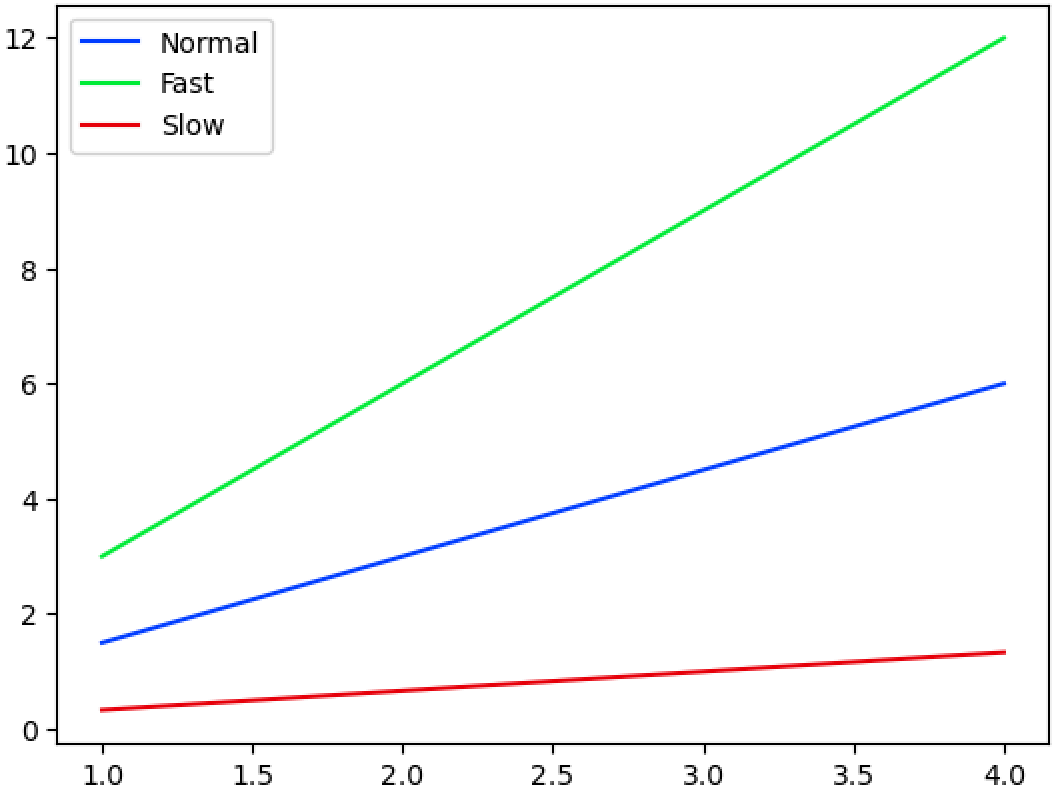


```
In [45]: # Adding a Legend

         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'])
         plt.show()
```



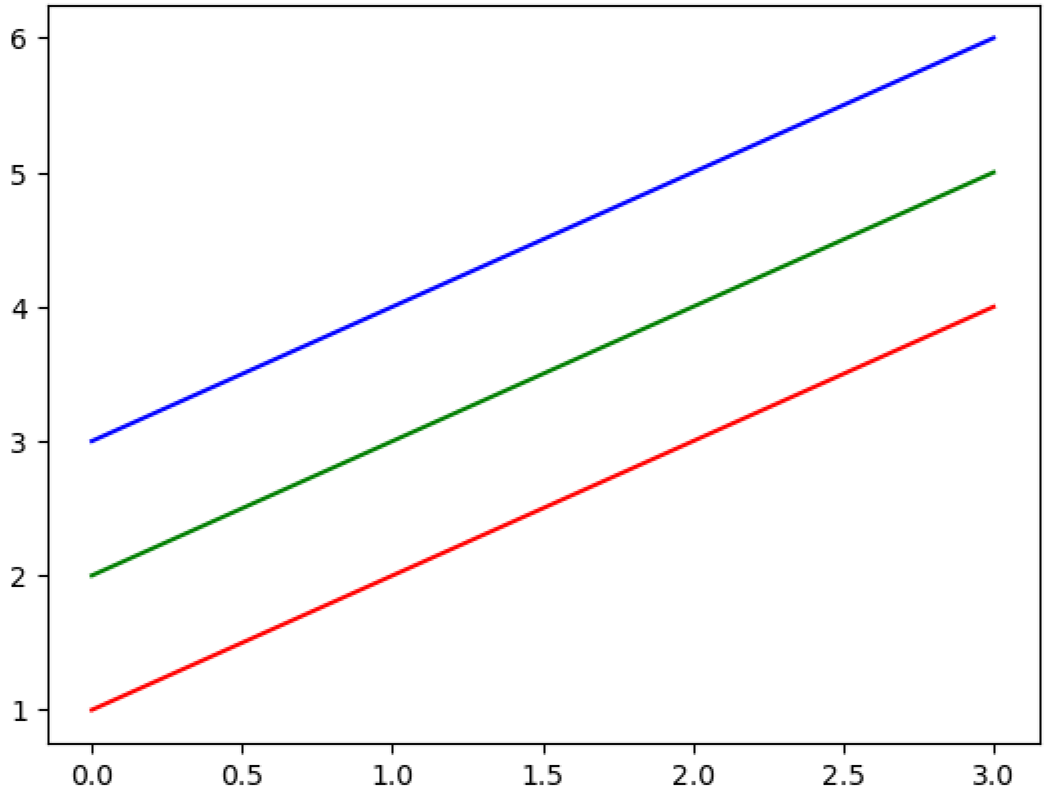
In [46]:

```
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()
plt.show()
```



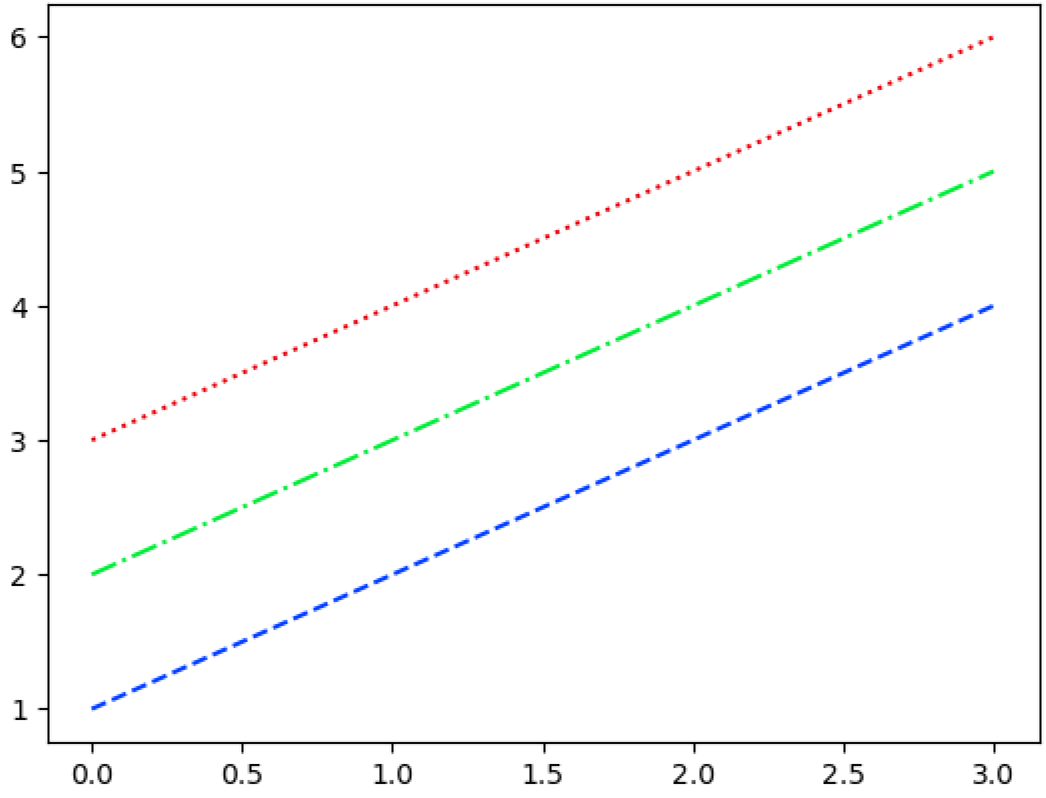
In [47]:

```
# Control colours

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



In [48]:

```
# Controls Line Styles

x16 = np.arange(1, 5)
plt.plot(x16, '--', x16+1, '-.', x16+2, ':')
plt.show()
```


6
5
4
3
2
1
0.0
0.5
1.0
1.5
2.0
2.5
3.0