In [4]:
```
import numpy as np
```

In [5]:
```
import pandas as pd
```

In [6]:
```
import matplotlib.pyplot as plt
```

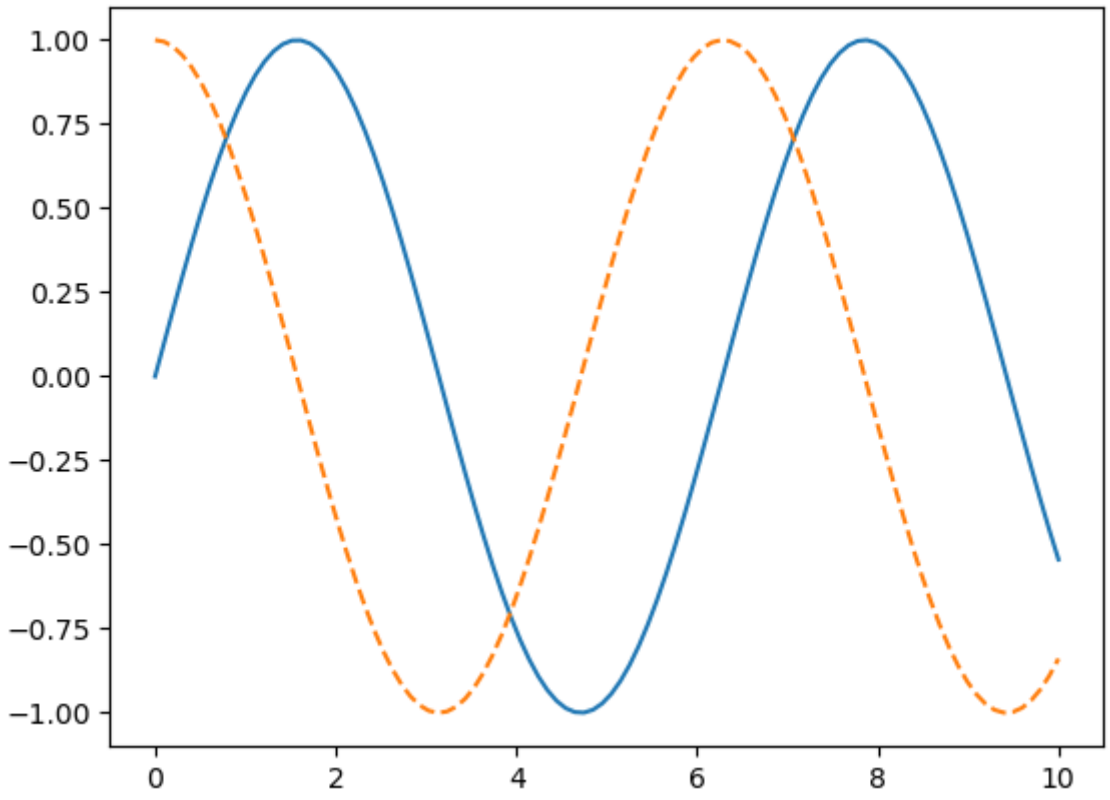
In [7]:
```
x1 = np.linspace(0, 10, 100)
```

In [8]:
```
fig = plt.figure()
```

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

In [9]:
```
plt.plot(x1, np.sin(x1), '-')
plt.plot(x1, np.cos(x1), '--')
```

Out[9]:
```
[<matplotlib.lines.Line2D at 0x1e45cccd880>]
```



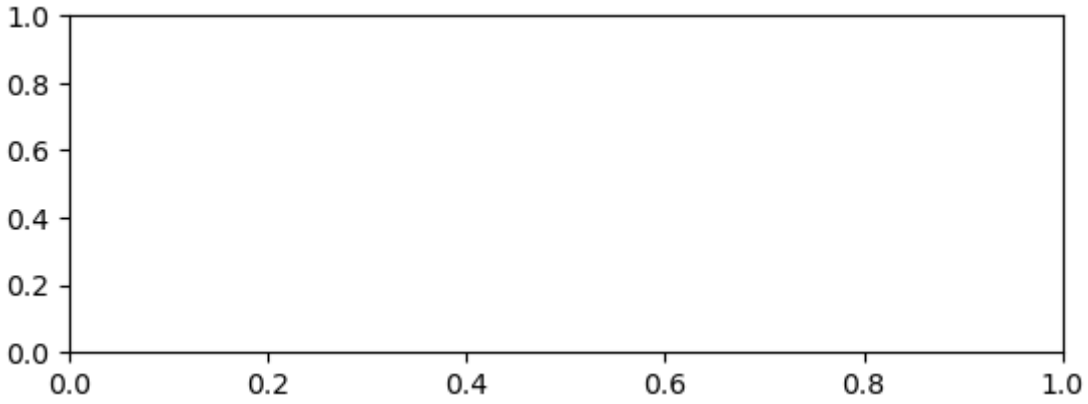
In [10]:
```
plt.figure()
```

Out[10]:
```
<Figure size 640x480 with 0 Axes>
```

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

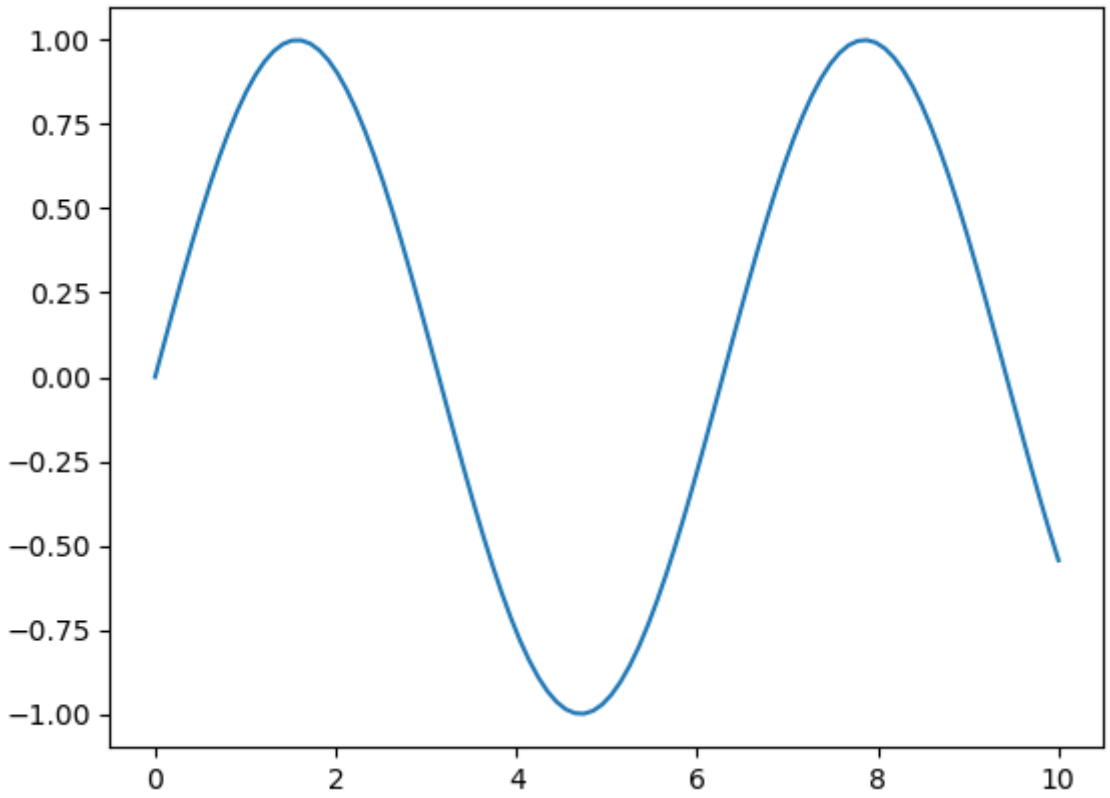
In [11]:
```
plt.subplot(2, 1, 1)
```

Out[11]:
```
<AxesSubplot:>
```



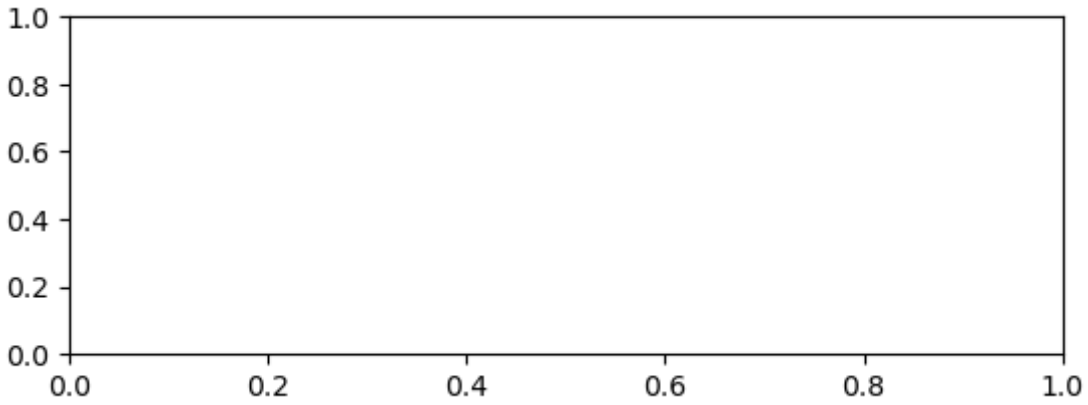
In [12]:
```
plt.plot(x1, np.sin(x1))
```

Out[12]:
```
[<matplotlib.lines.Line2D at 0x1e45cd7c7c0>]
```



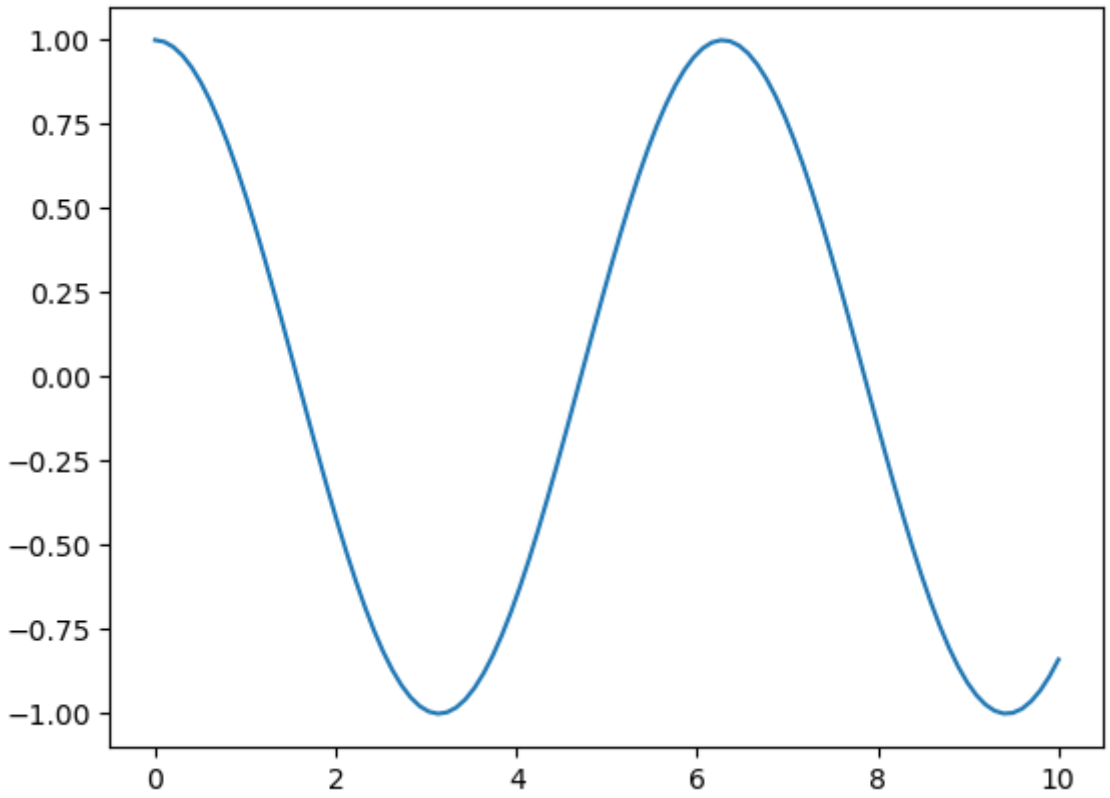
In [13]:
```
plt.subplot(2, 1, 2)
```

Out[13]:
```
<AxesSubplot:>
```



In [17]:
```
plt.plot(x1, np.cos(x1))
```

Out[17]:
```
[<matplotlib.lines.Line2D at 0x1e45e088610>]
```

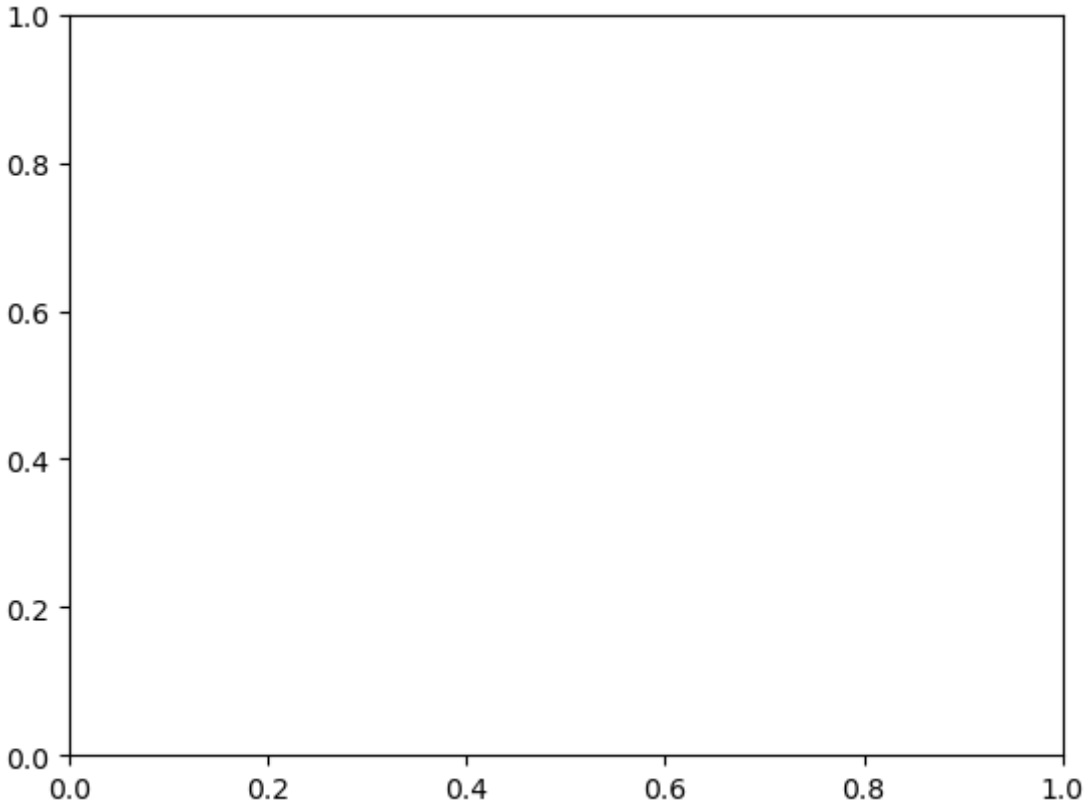


In [18]:
```
print(plt.gcf())
```

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

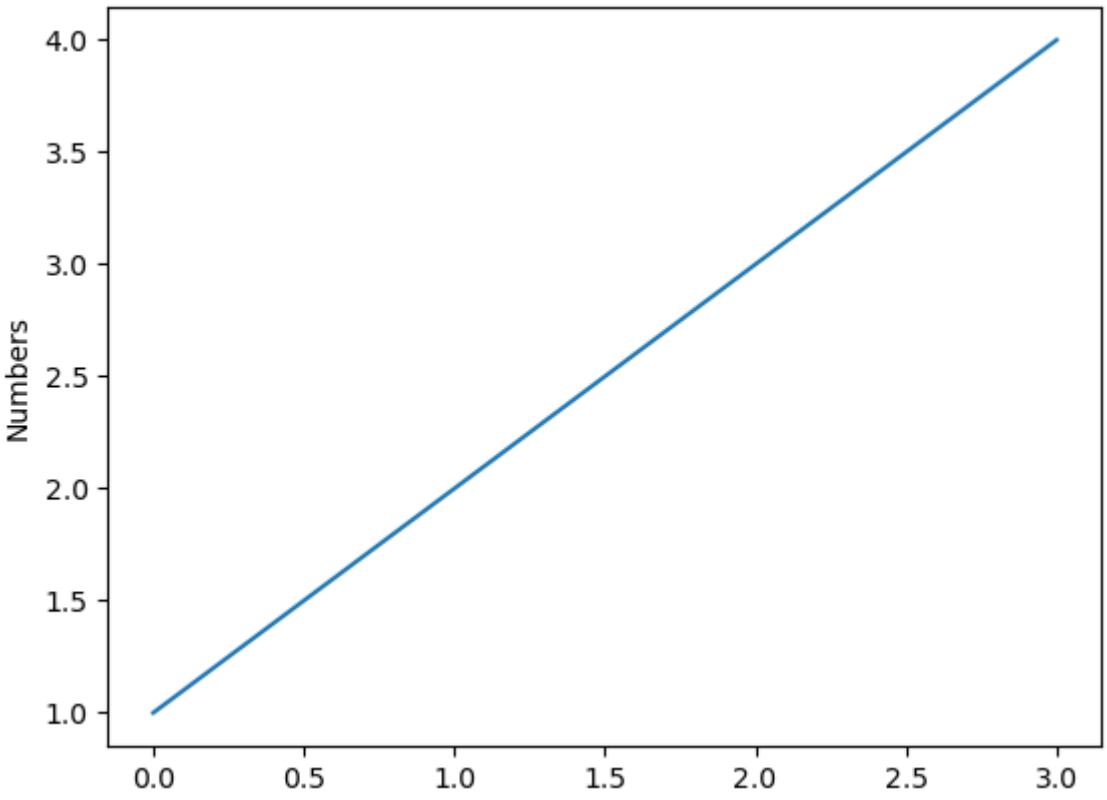
In [19]:
```
print(plt.gca())
```

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



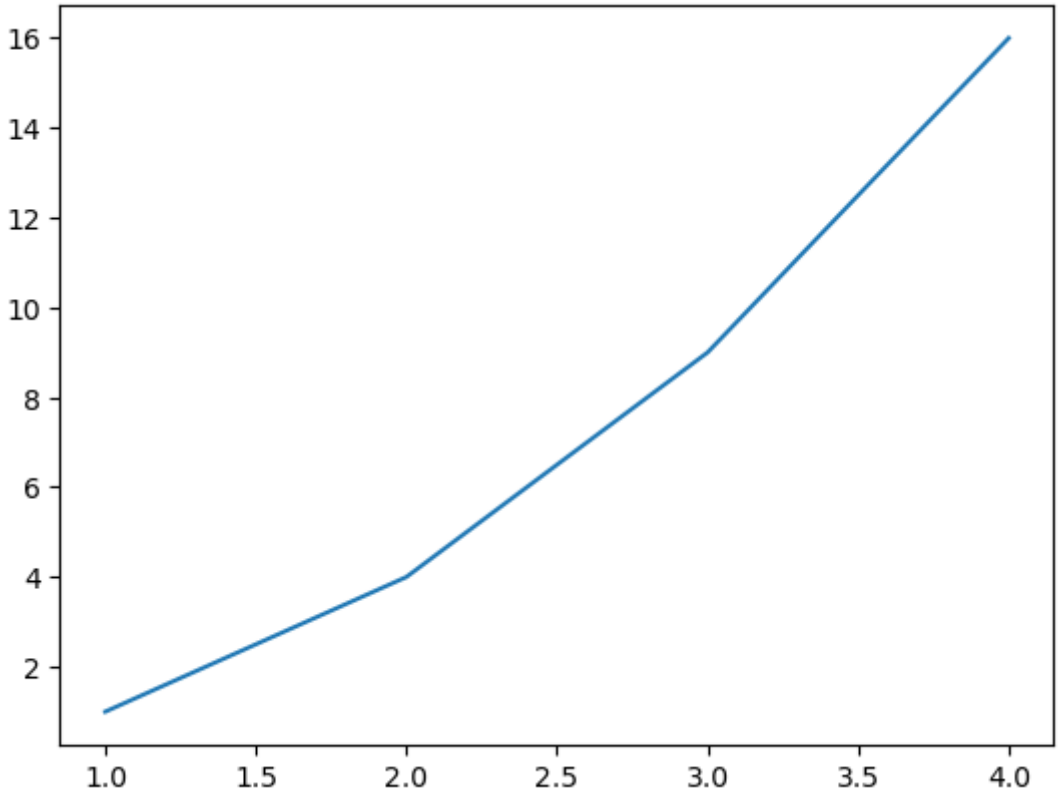
In [22]:

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



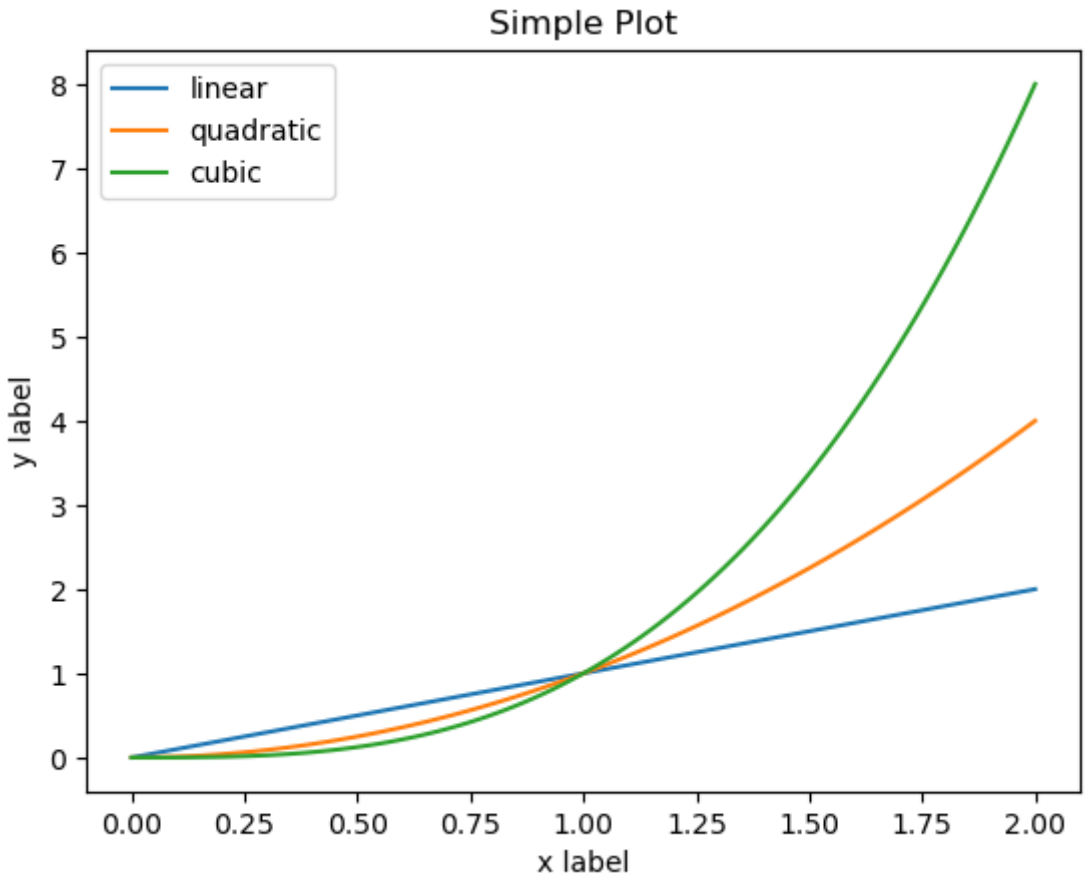
In [24]:

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



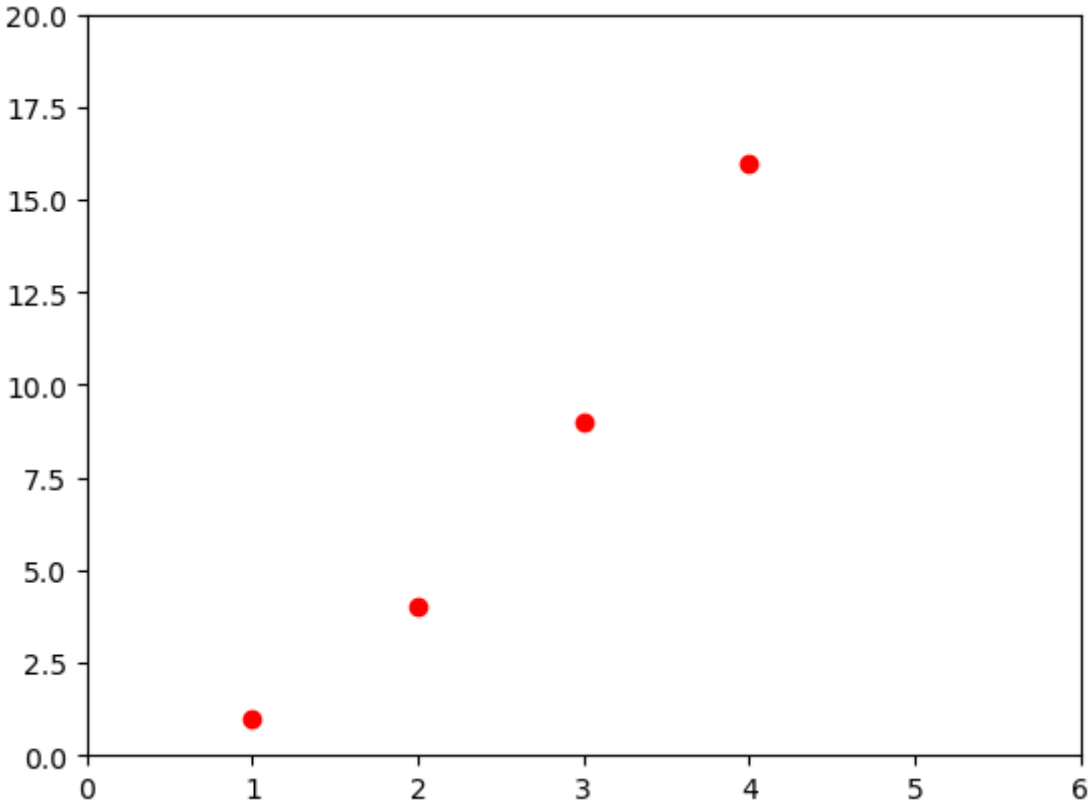
In [25]:

```
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 [26]:

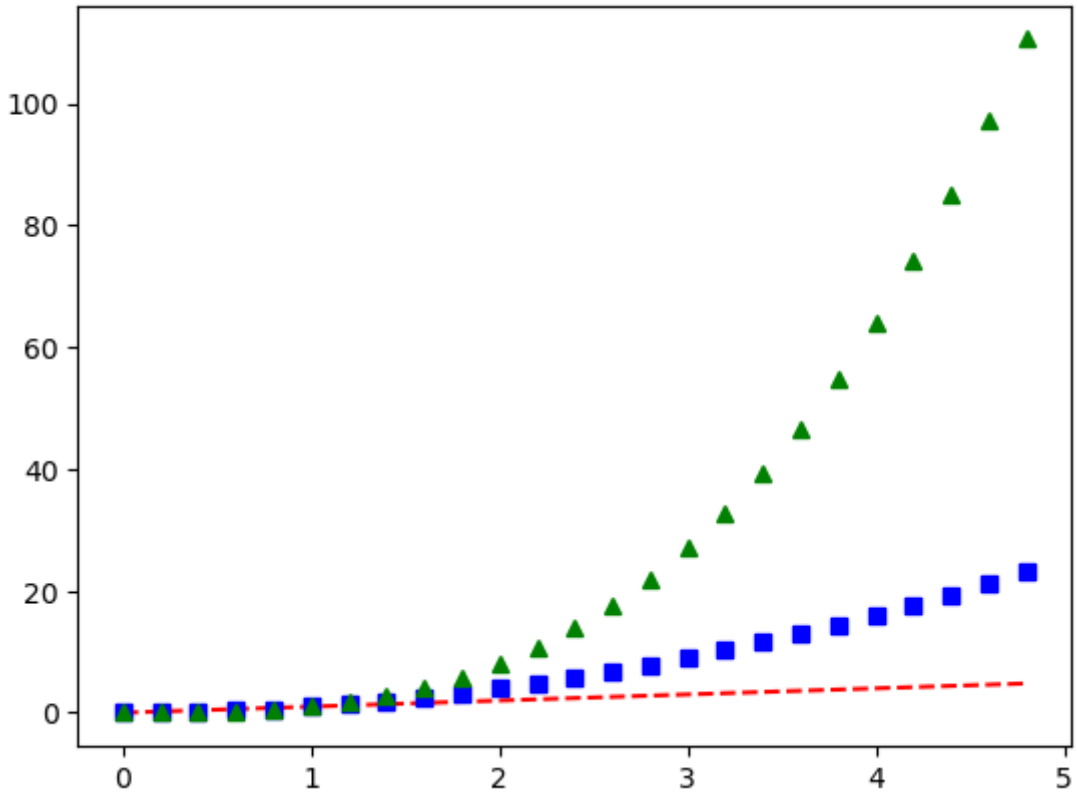
```
plt.plot([1,2,3,4],[1,4,9,16], 'ro')
plt.axis([0,6,0,20])
plt.show()
```



In [28]:

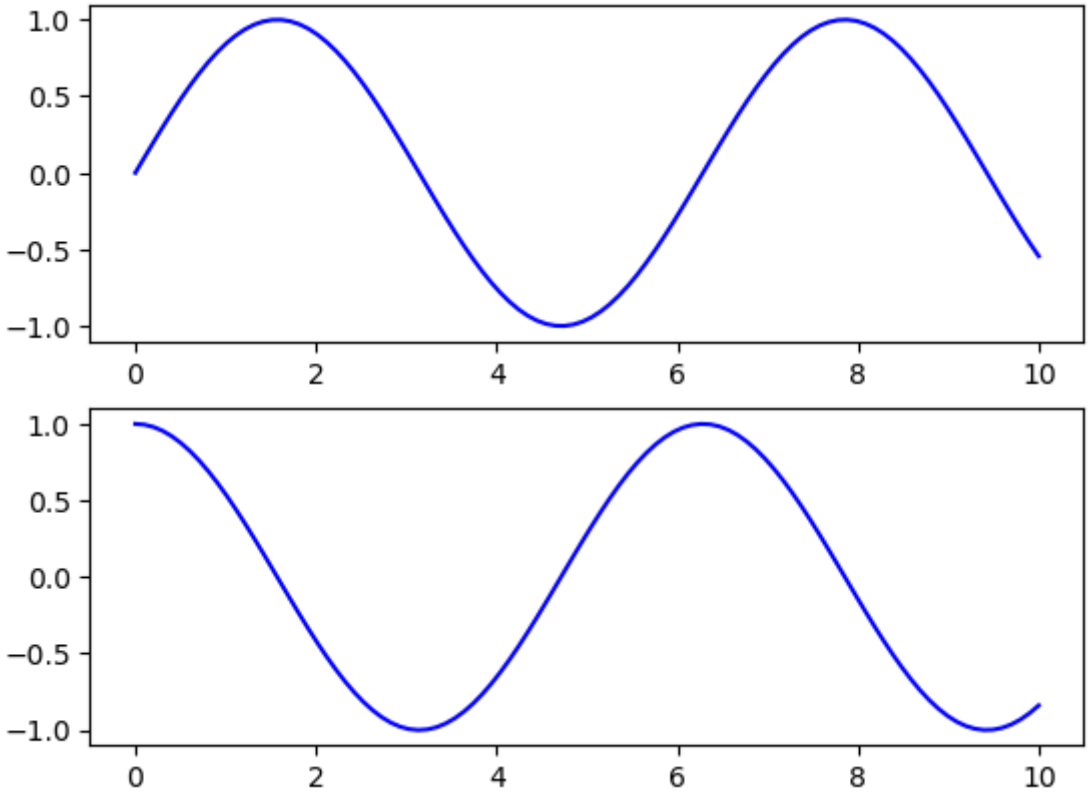
```
t = np.arange(0., 5., 0.2)
```

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



In [29]:
```
# 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[29]:
```
[<matplotlib.lines.Line2D at 0x1e45e489970>]
```

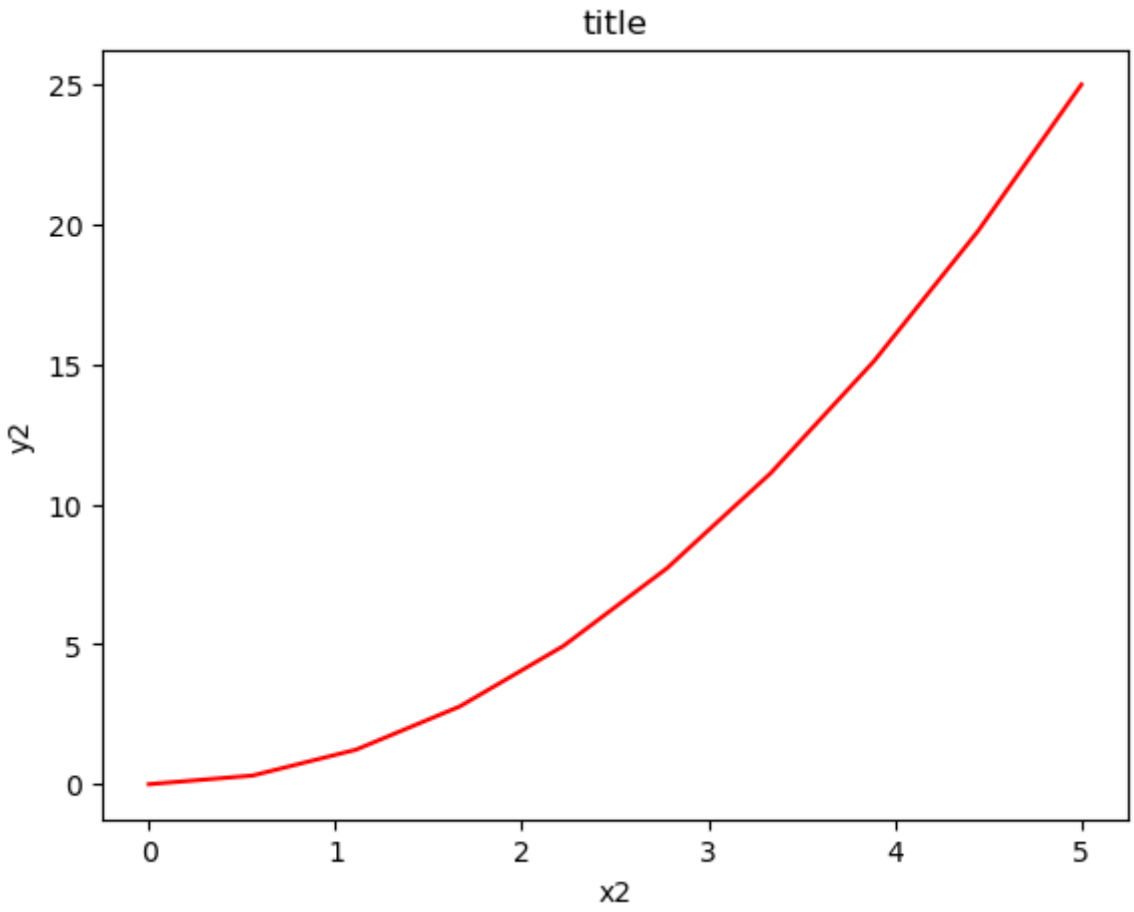


In [30]:

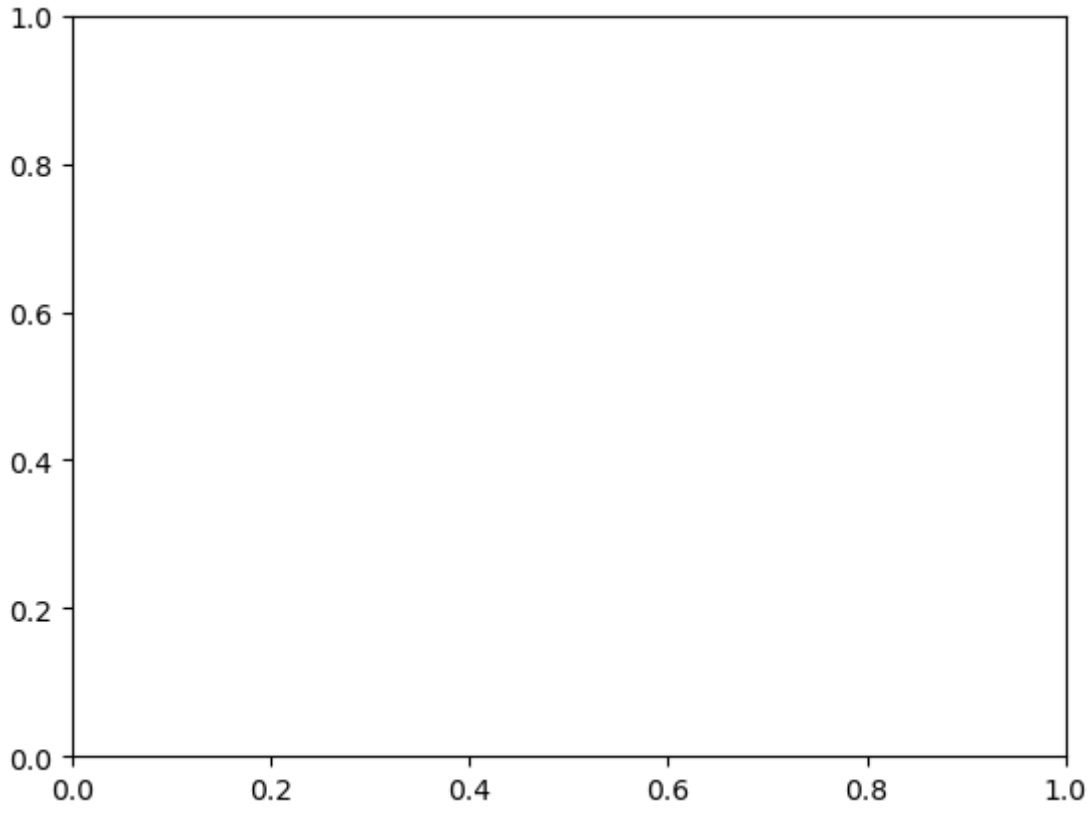
```
fig = plt.figure()

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[30]:

```
Text(0.5, 1.0, 'title')
```

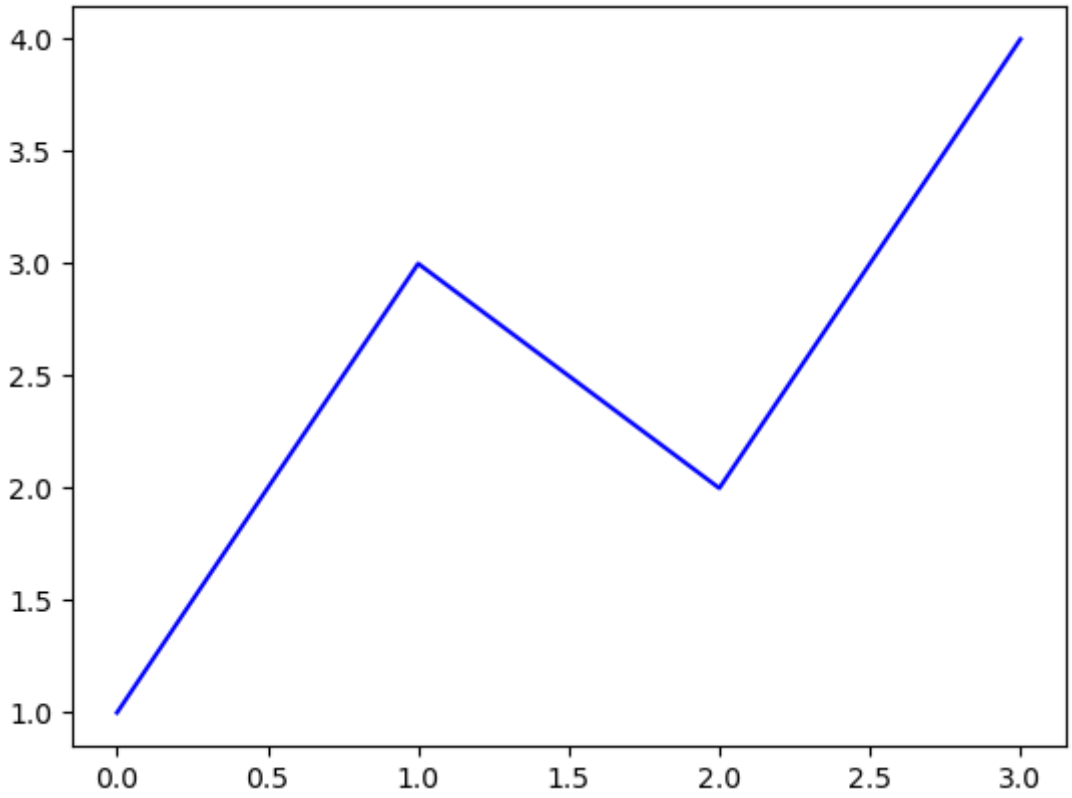


In [31]:
```
fig = plt.figure()

ax = plt.axes()
```



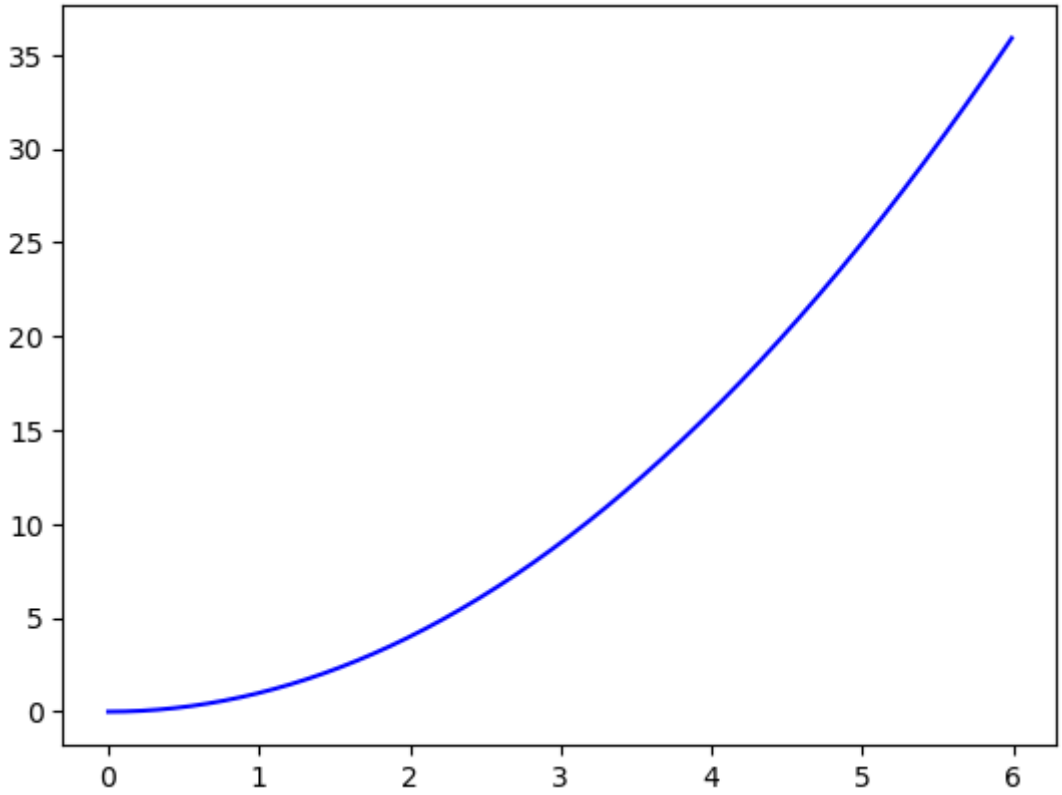
In [32]:
```
plt.plot([1, 3, 2, 4], 'b-')
```

```
plt.show( )
```



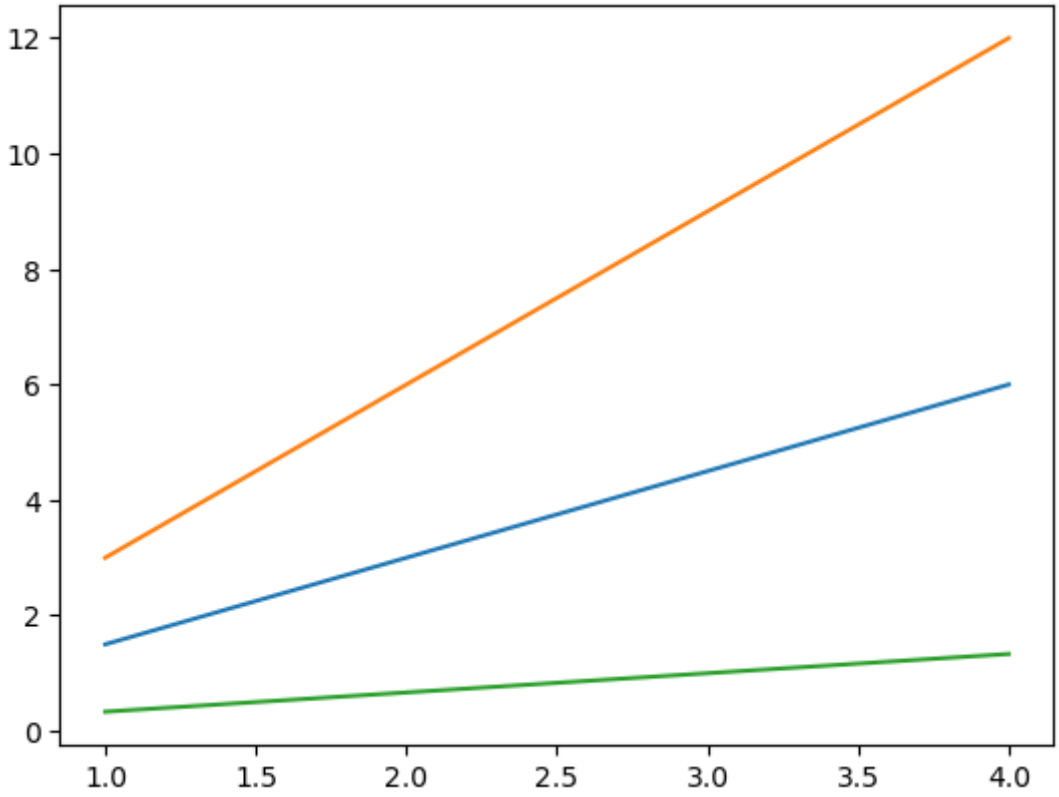
In [33]:

```
x3 = np.arange(0.0, 6.0, 0.01)

plt.plot(x3, [xi**2 for xi in x3], 'b-')

plt.show()
```



In [34]:

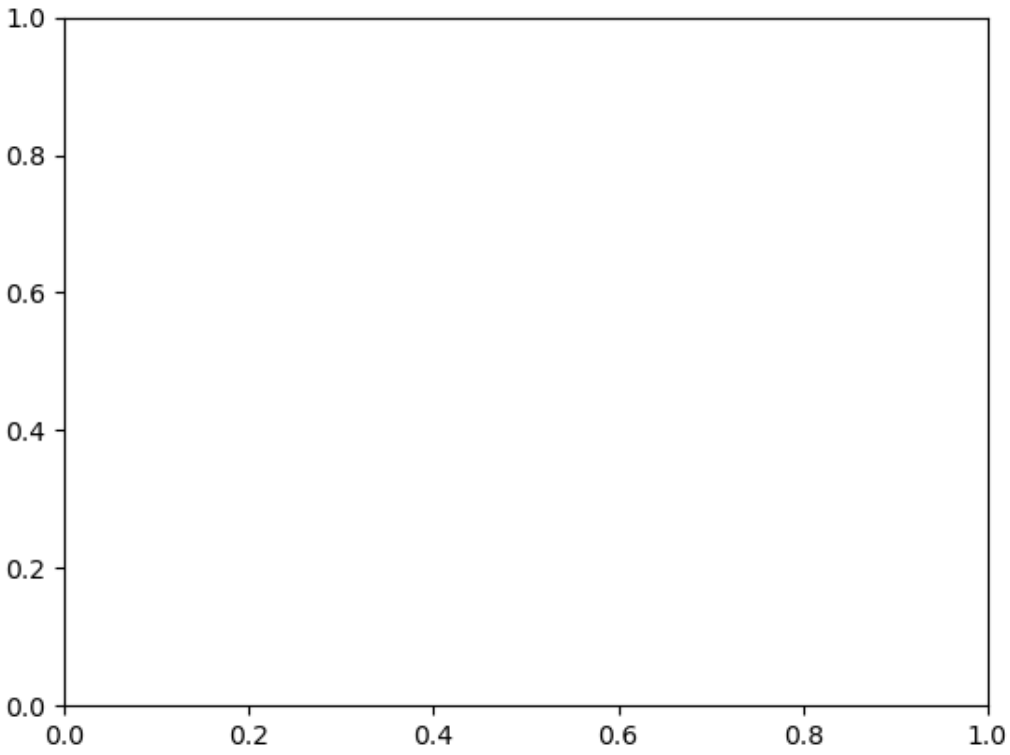
```
x4 = range(1, 5)

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 [35]:
```
fig.savefig('plot1.png')
```

In [36]:
```
from IPython.display import Image
Image('plot1.png')
```

Out[36]:



In [41]:
```
print(np. __version__)  # Just Checking the version of numpy i am using
```

```
1.21.5
```

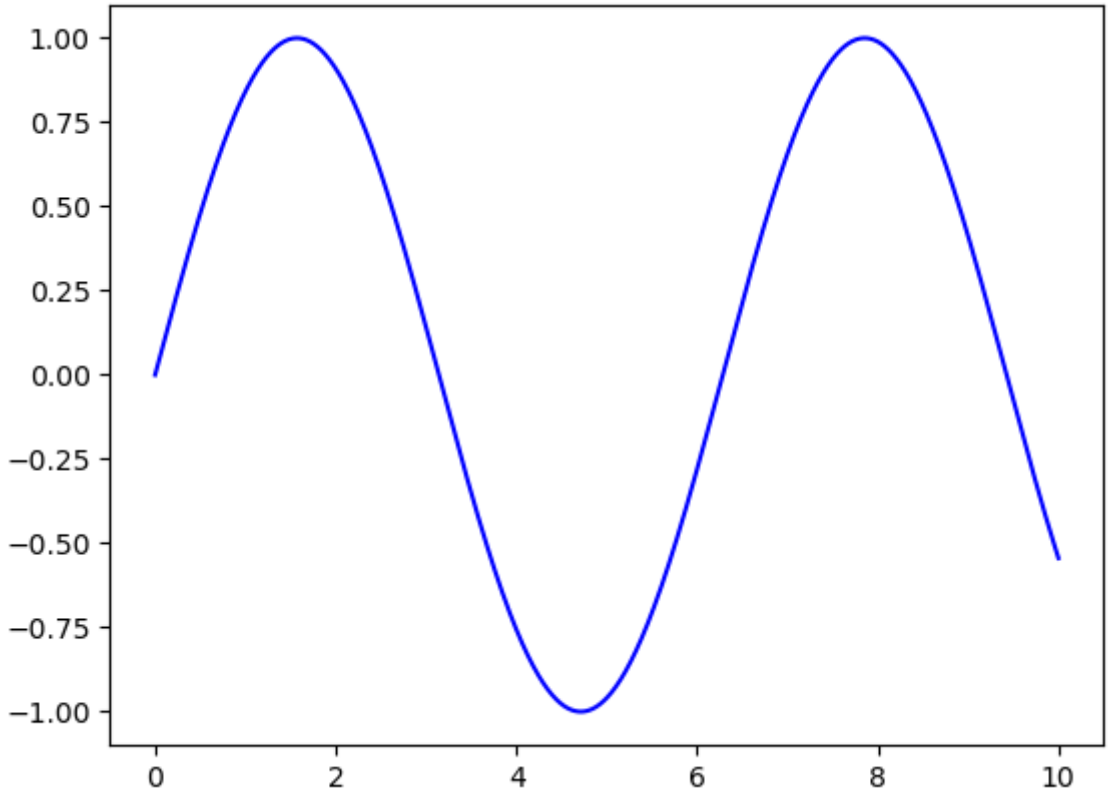
In [42]:
```
print(pd. __version__)  # Just Checking the version of pandas i am using
```

```
1.4.4
```

In [44]:
```
fig.canvas.get_supported_filetypes()
```

Out[44]:
```
{'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'}
```

In [12]:
```
# 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-');
```

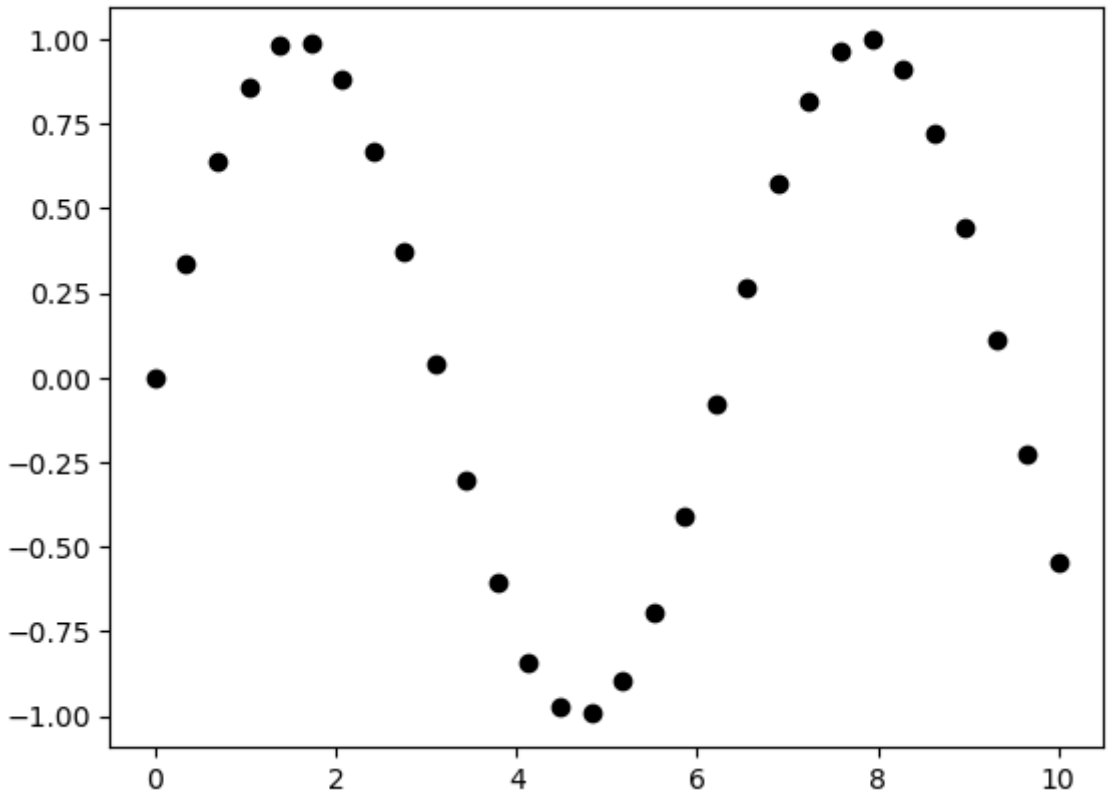


In [13]:

```
x7 = np.linspace(0, 10, 30)

y7 = np.sin(x7)

plt.plot(x7,y7, 'o', color = 'black')
```

Out[13]:

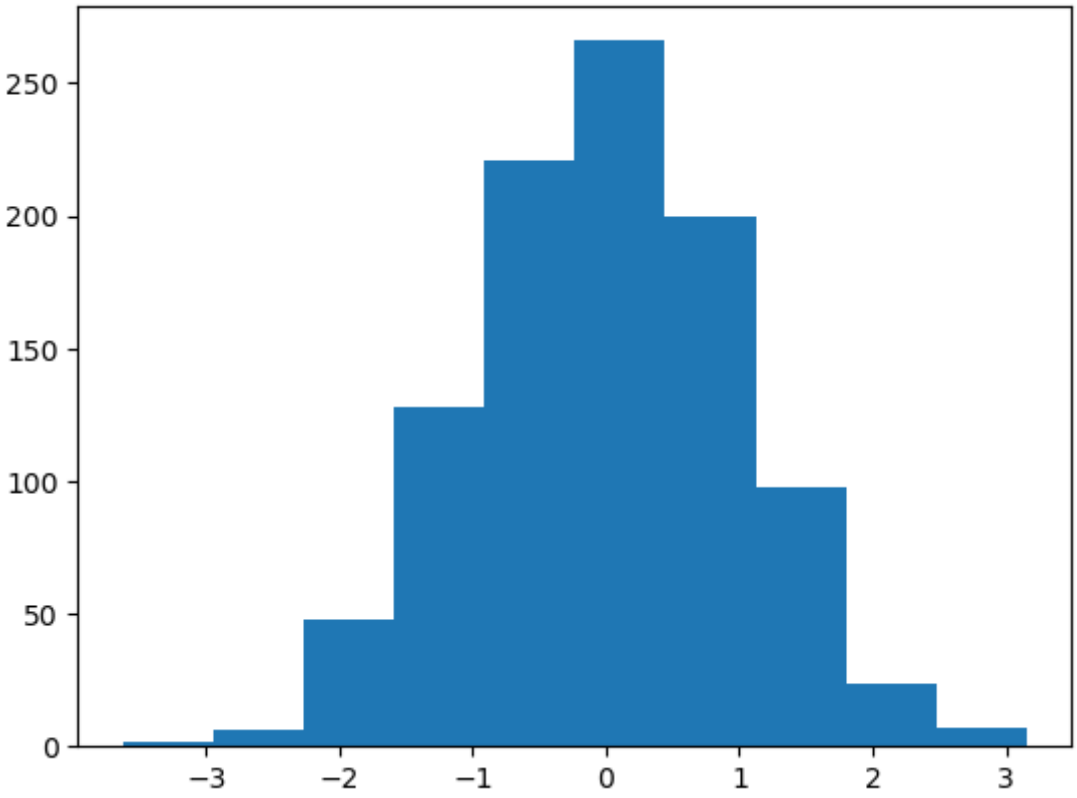
```
[<matplotlib.lines.Line2D at 0x27698764160>]
```



In [14]:
```
data1 = np.random.randn(1000)
```

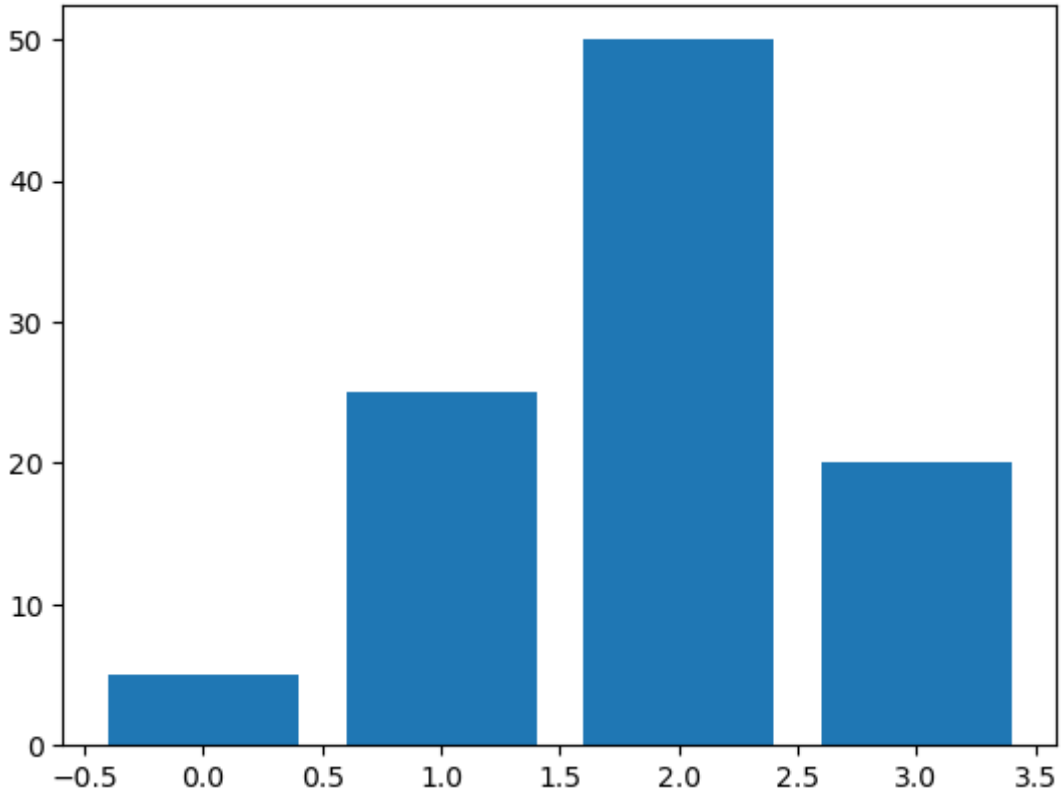
In [15]:
```
plt.hist(data1)
```

Out[15]:
```
(array([  2.,   6.,  48., 128., 221., 266., 200.,  98.,  24.,   7.]),
 array([-3.62321847, -2.9454857 , -2.26775293, -1.59002016, -0.91228739,
        -0.23455461,  0.44317816,  1.12091093,  1.7986437 ,  2.47637647,
         3.15410924]),
 <BarContainer object of 10 artists>)
```

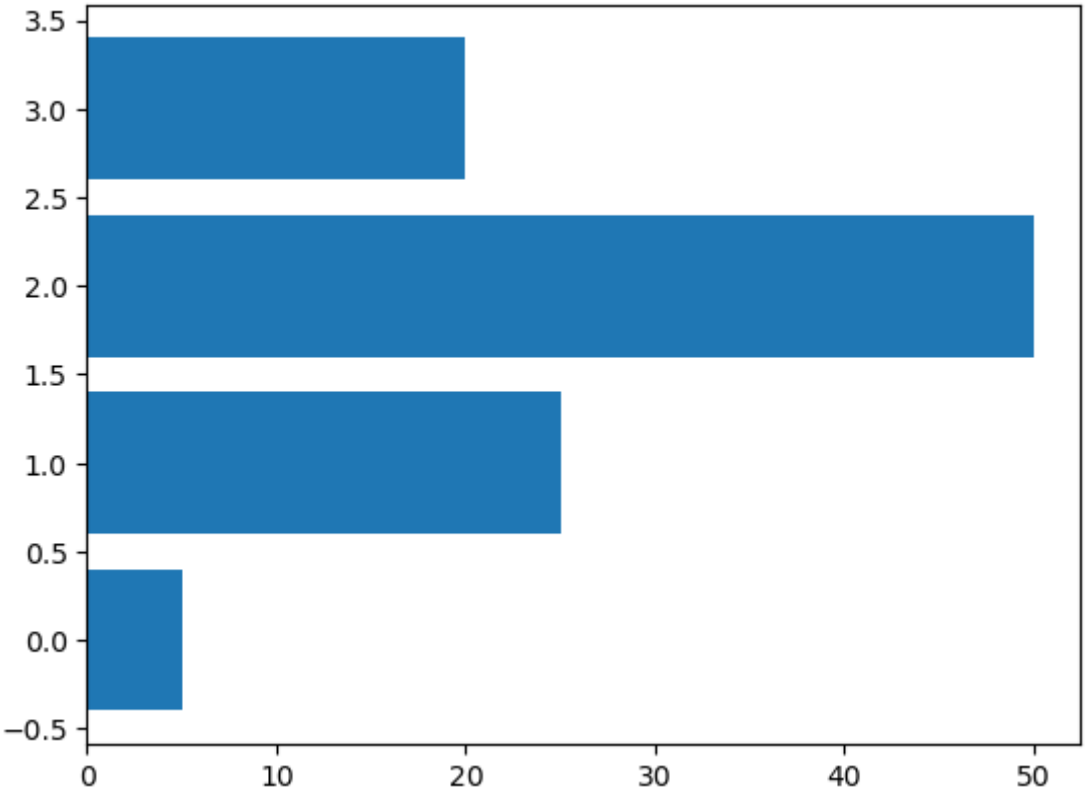


In [17]:
```
data2 = [5., 25., 50., 20. ]
```

In [19]:
```
plt.bar(range(len(data2)), data2)
plt.show()
```



In [20]:
```
plt.barh(range(len(data2)), data2)
plt.show()
```

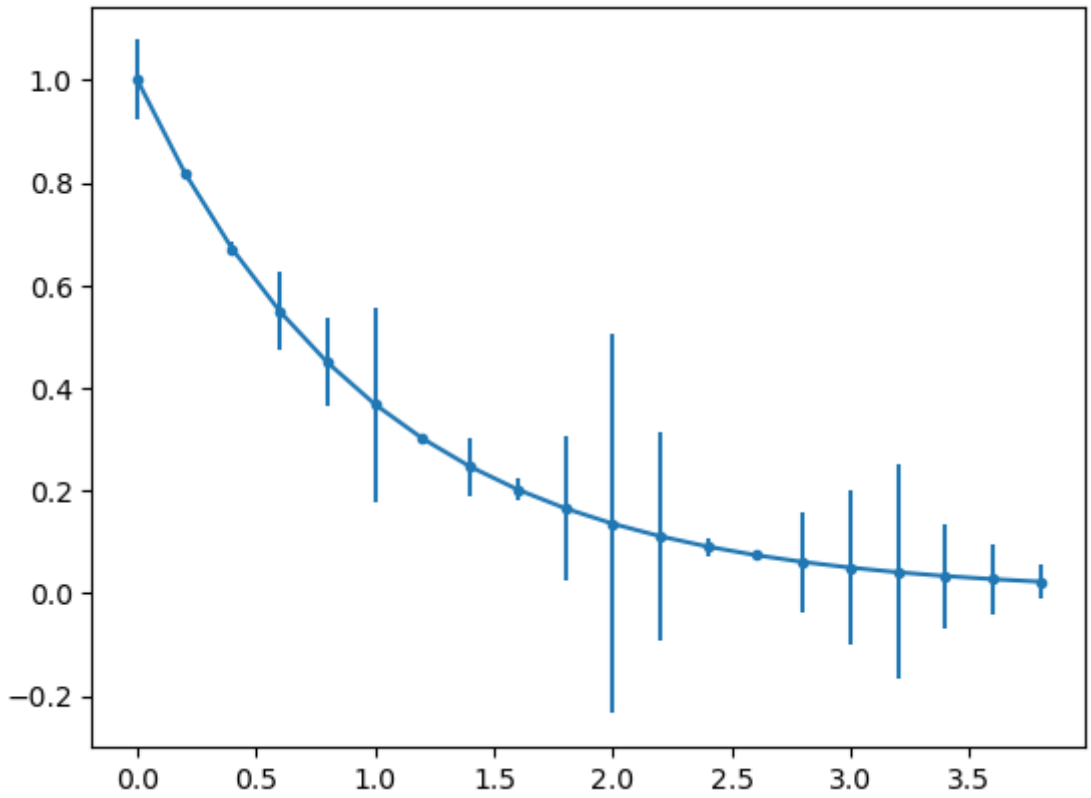


In [21]:
```
x9 = np.arange(0, 4, 0.2)
```

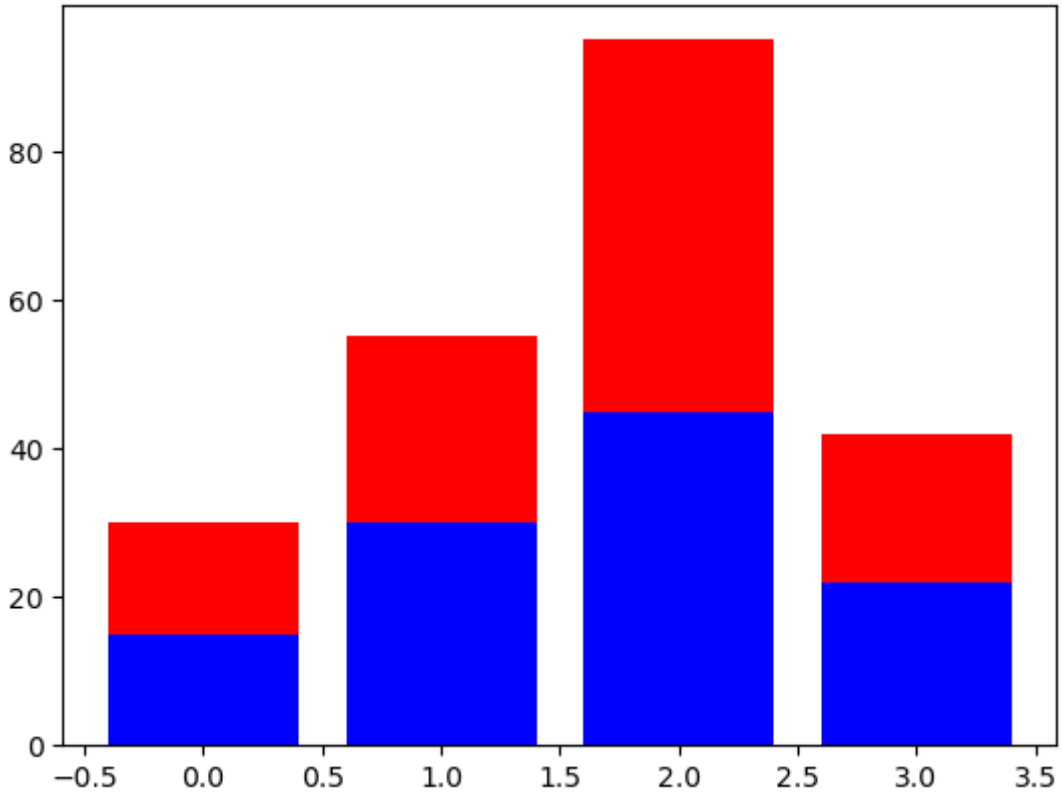
In [22]:
```
y9 = np.exp(-x9)
```

In [23]:
```
e1 = 0.1 * np.abs(np.random.randn(len(y9)))
```

In [26]:
```
plt.errorbar(x9, y9, yerr = e1, fmt = '.-')
plt.show()
```

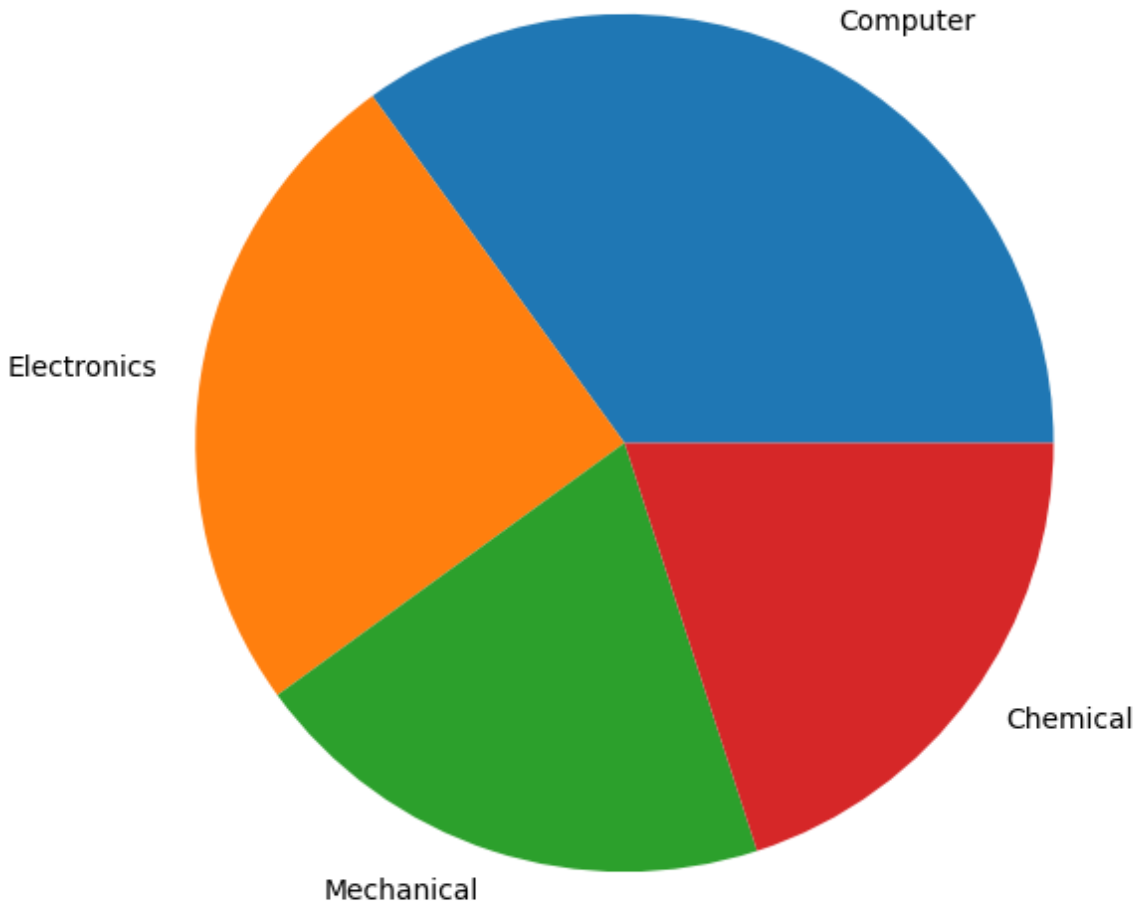


In [27]:
```
A = [15., 30., 45., 22.]

B = [15., 25., 50., 20.]

z2 = range(4)

plt.bar(z2, A, color = 'b')
plt.bar(z2, B, color = 'r', bottom = A)

plt.show()
```



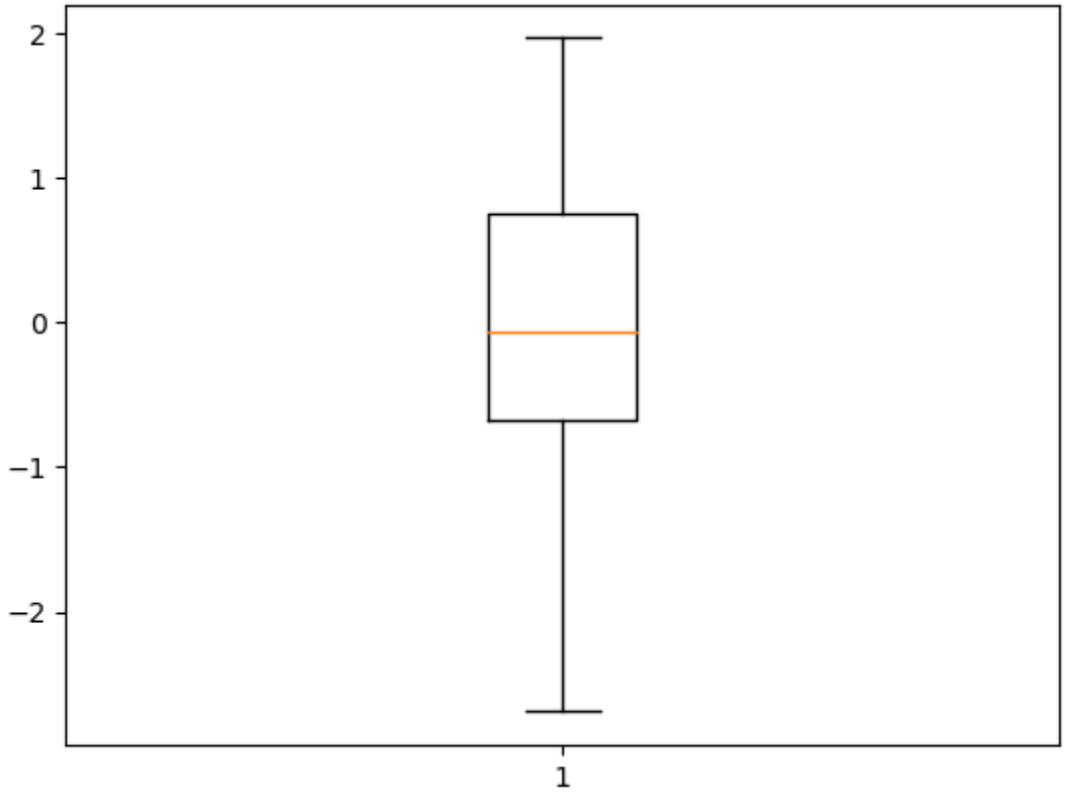
In [28]:

```
plt.figure(figsize=(7,7))

x10 = [35, 25, 20, 20]

labels = ['Computer', 'Electronics', 'Mechanical', 'Chemical']

plt.pie(x10, labels=labels);

plt.show()
```



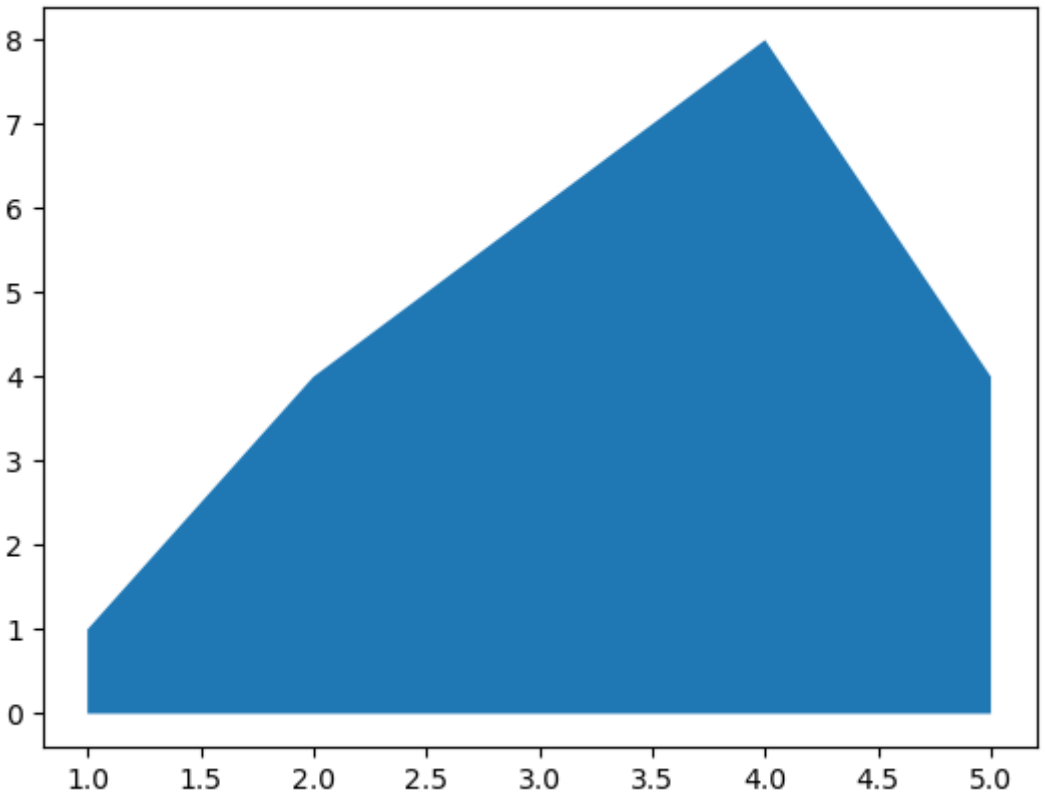
In [29]:
```
data3 = np.random.randn(100)
```

In [32]:
```
plt.boxplot(data3)
plt.show()
```



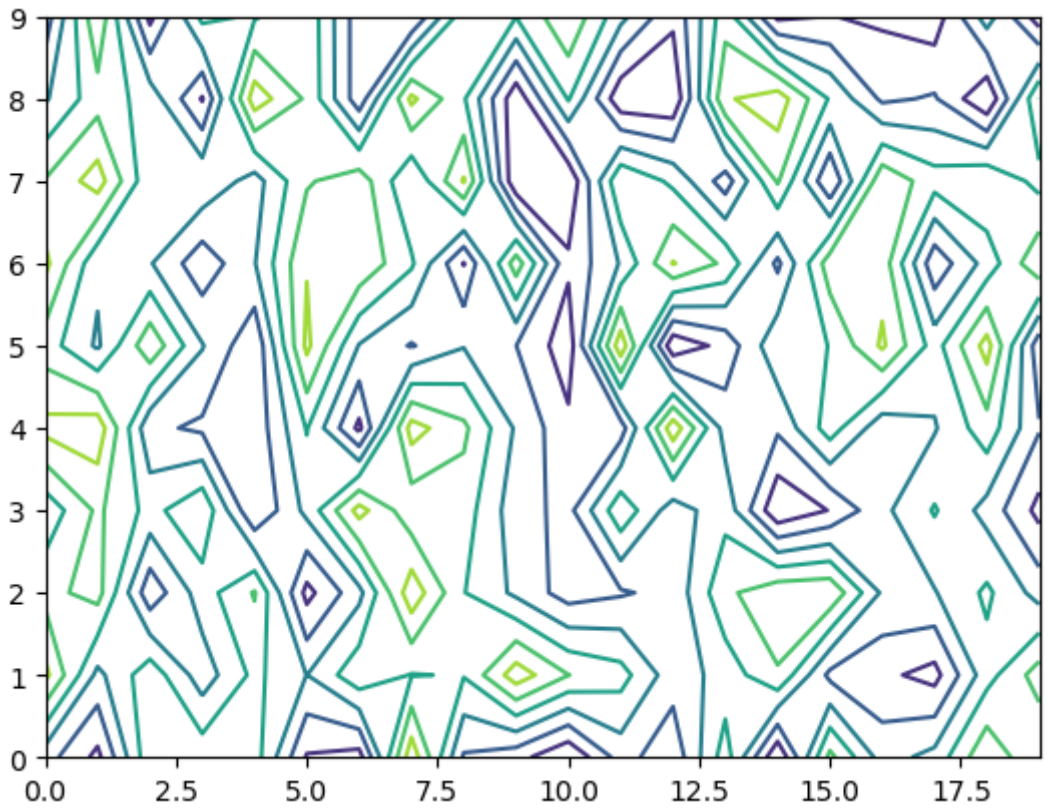
In [33]:
```
x12 = range(1,6)
y12 = [1,4,6,8,4]
```

In [34]:
```
plt.fill_between(x12,y12)
plt.show()
```



In [37]:
```
matrix1 = np.random.rand(10, 20)
```

In [40]:
```
cp = plt.contour(matrix1)
plt.show()
```



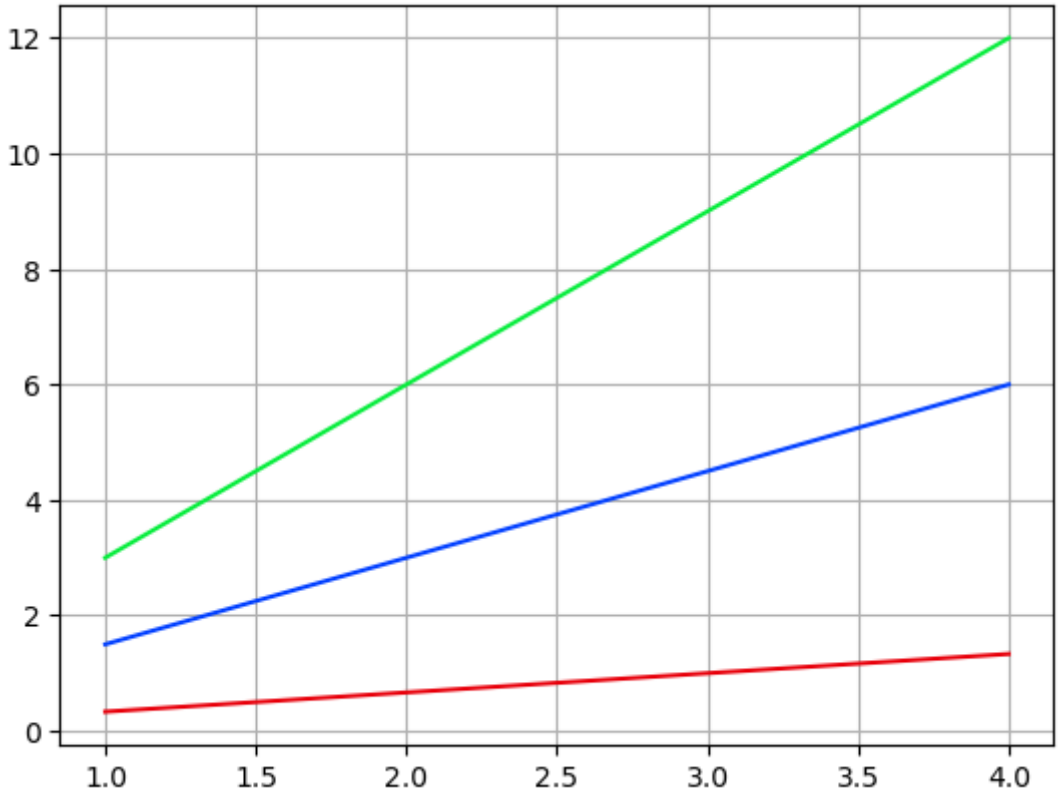
In [41]:
```
print(plt.style.available)
```

```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid',
'bmh', 'classic', 'dark_background', 'fast', 'fivethirtyeight', 'ggplot', 'graysca
le', 'seaborn', 'seaborn-bright', 'seaborn-colorblind', 'seaborn-dark', 'seaborn-d
ark-palette', 'seaborn-darkgrid', 'seaborn-deep', 'seaborn-muted', 'seaborn-notebo
ok', 'seaborn-paper', 'seaborn-pastel', 'seaborn-poster', 'seaborn-talk', 'seaborn
-ticks', 'seaborn-white', 'seaborn-whitegrid', 'tableau-colorblind10']
```

In [42]:
```
plt.style.use('seaborn-bright')
```

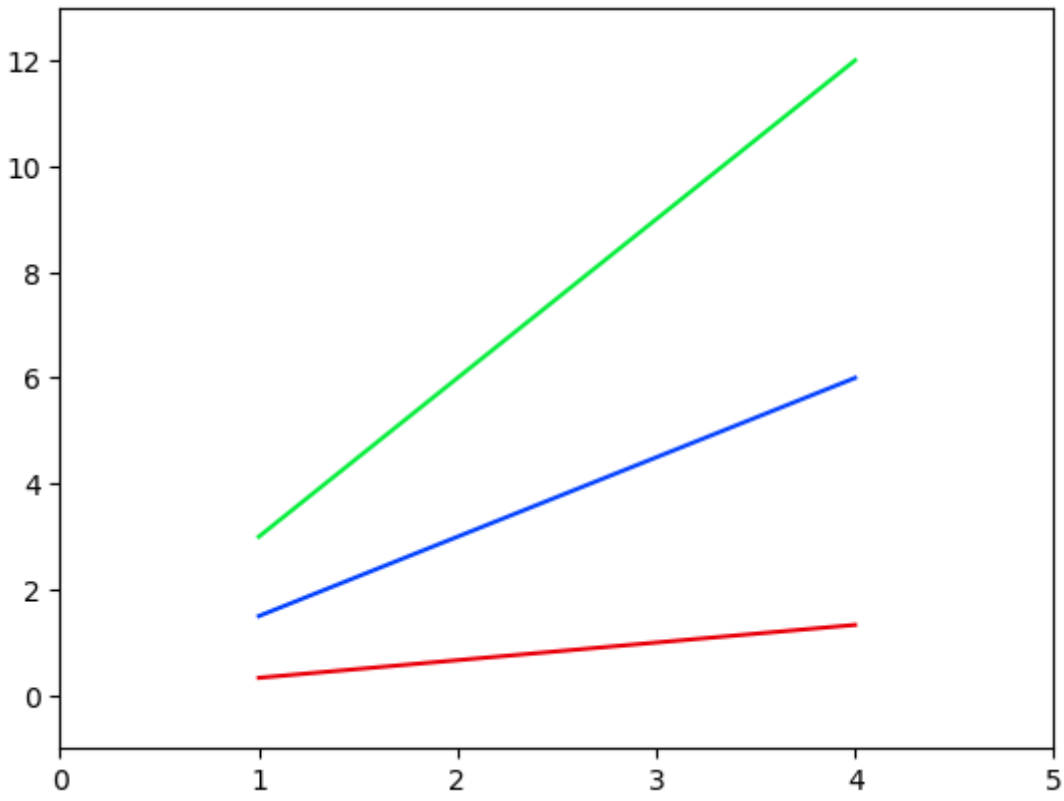
In [43]:
```
x15 = np.arange(1, 5)
```

In [46]:
```
plt.plot(x15, x15*1.5, x15, x15*3.0,x15,x15/3.0)
plt.grid(True)
plt.show()
```

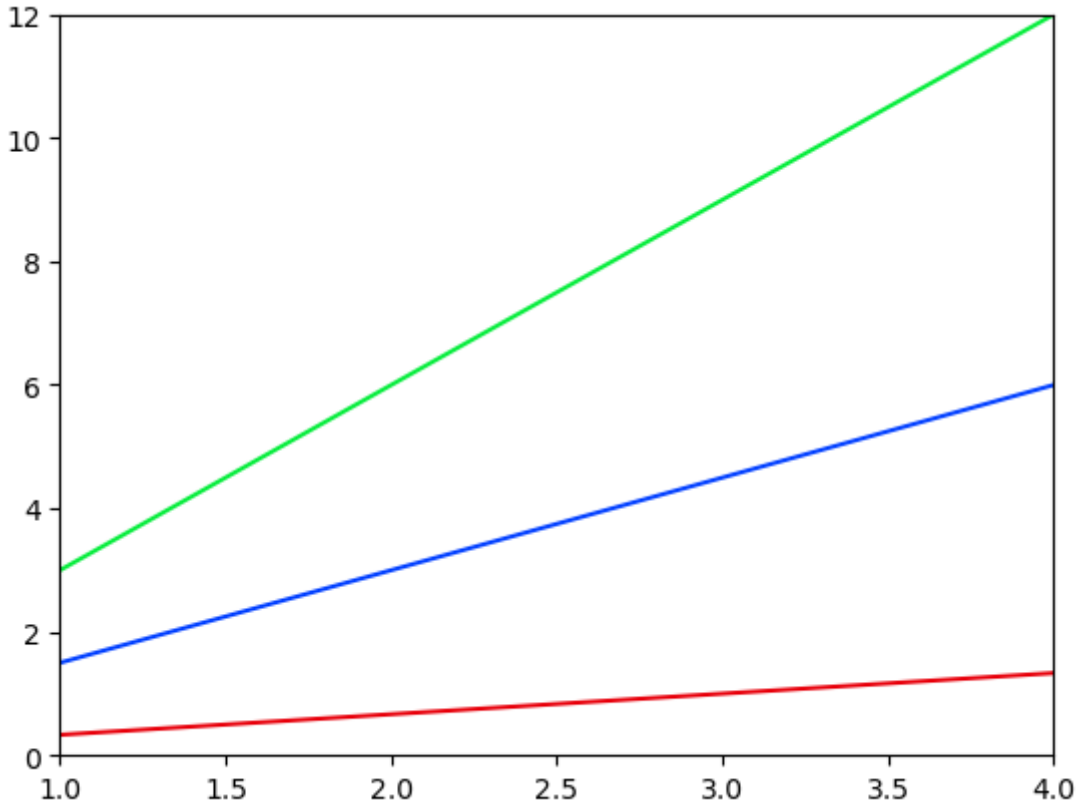


In [47]:
```
x15 = np.arange(1,5)
```

In [50]:
```
plt.plot(x15, x15*1.5, x15, x15*3.0,x15,x15/3.0)
plt.axis()
plt.axis([0, 5, -1, 13])
plt.show()
```

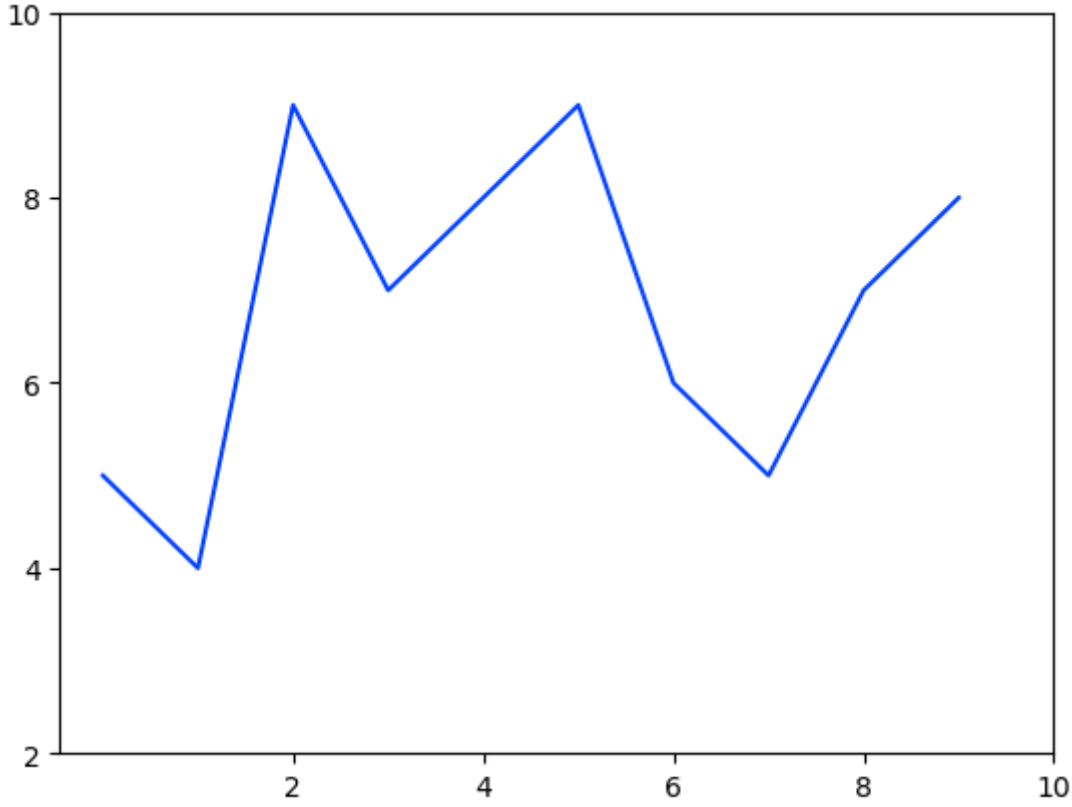


In [52]:
```
plt.plot(x15, x15*1.5, x15, x15*3.0,x15,x15/3.0)
plt.xlim([1.0, 4.0])
plt.ylim([0, 12.0])
plt.show()
```



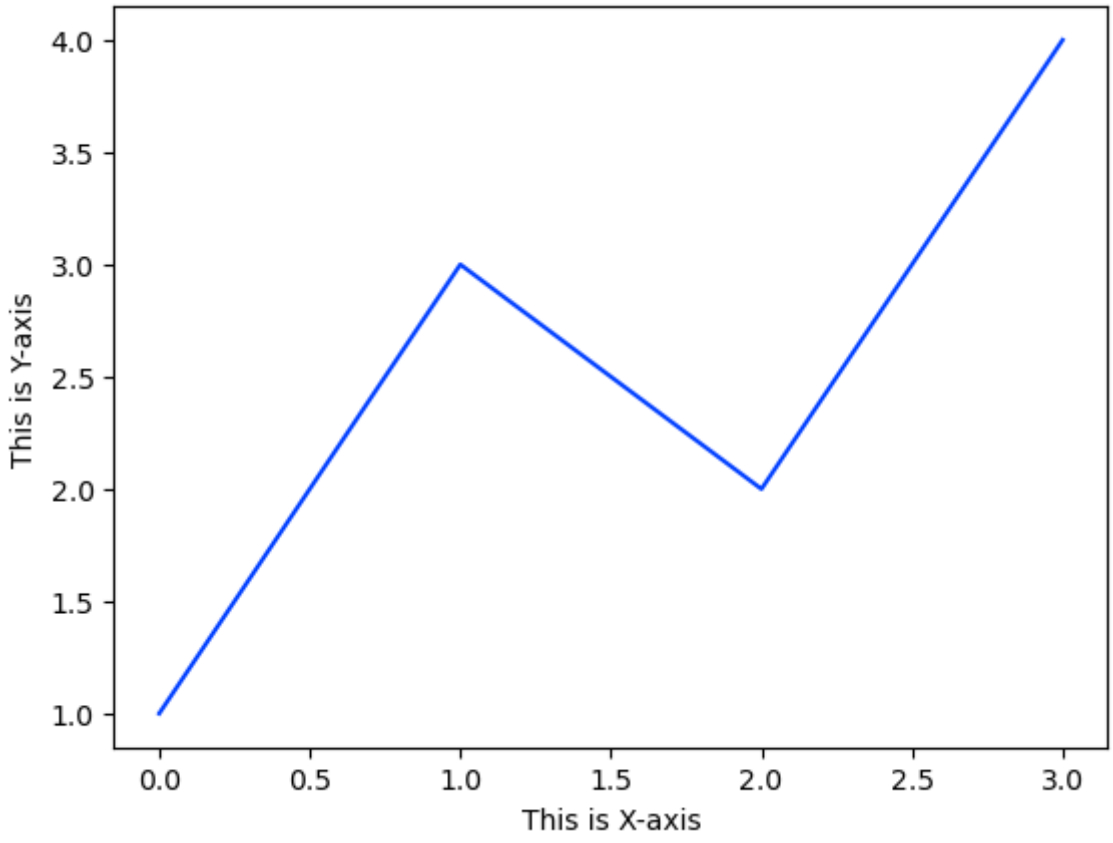
In [53]:
```
u = [5, 4, 9, 7, 8, 9, 6, 5, 7, 8]
```

In [56]:
```
plt.plot(u)
plt.xticks([2,4,6,8,10])
plt.yticks([2,4,6,8,10])
plt.show()
```



In [60]:

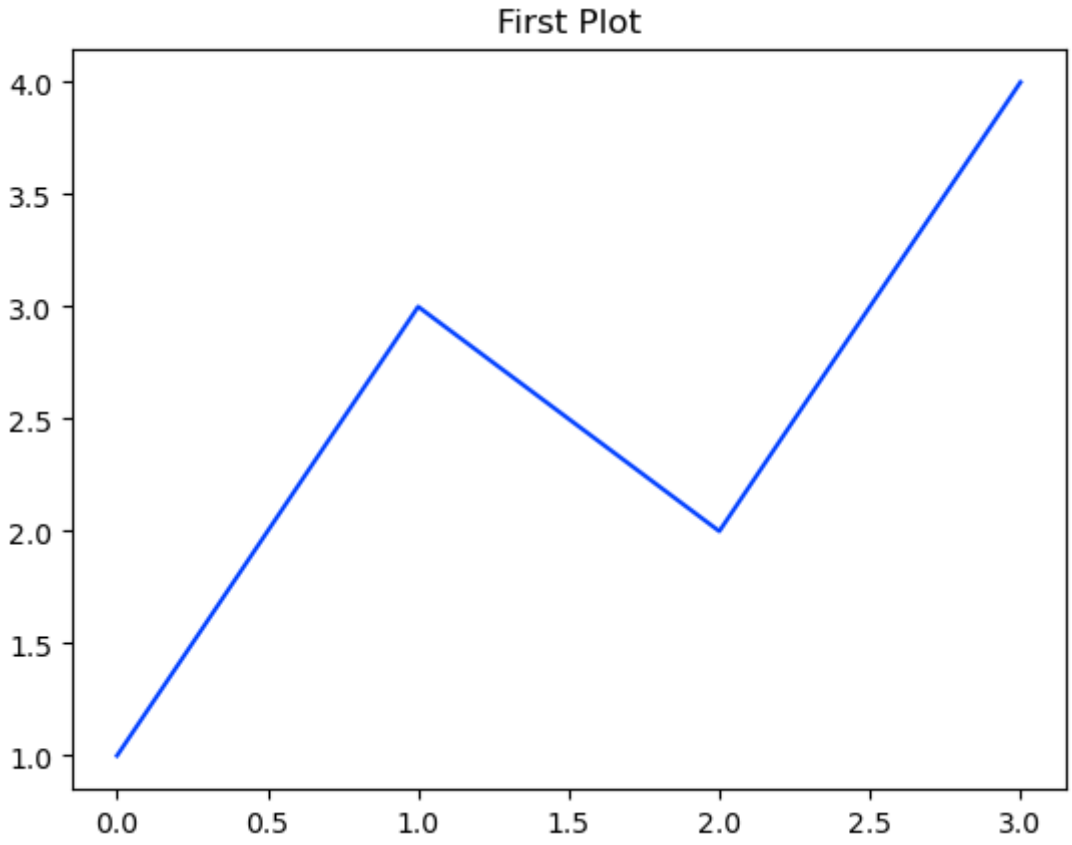
```
plt.plot([1,3,2,4])
plt.xlabel('This is X-axis')
plt.ylabel('This is Y-axis')
plt.show()
```



In [62]:

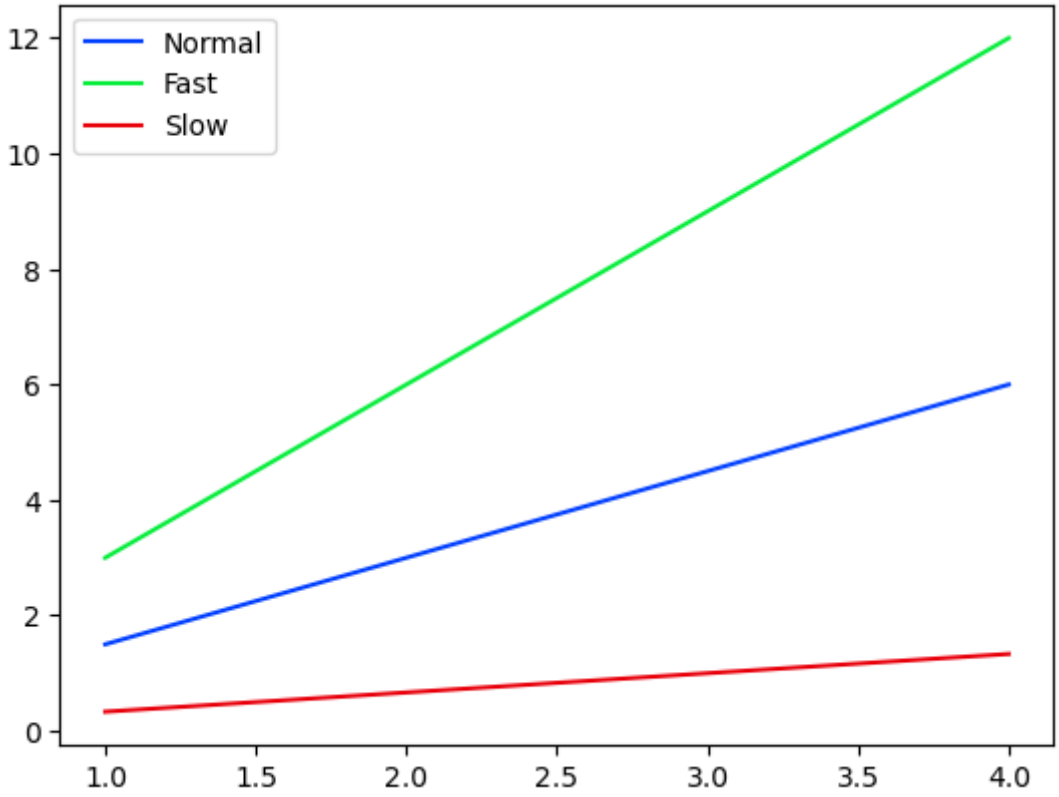
```
plt.plot([1,3,2,4])
```

```
plt.title('First Plot')

plt.show()
```



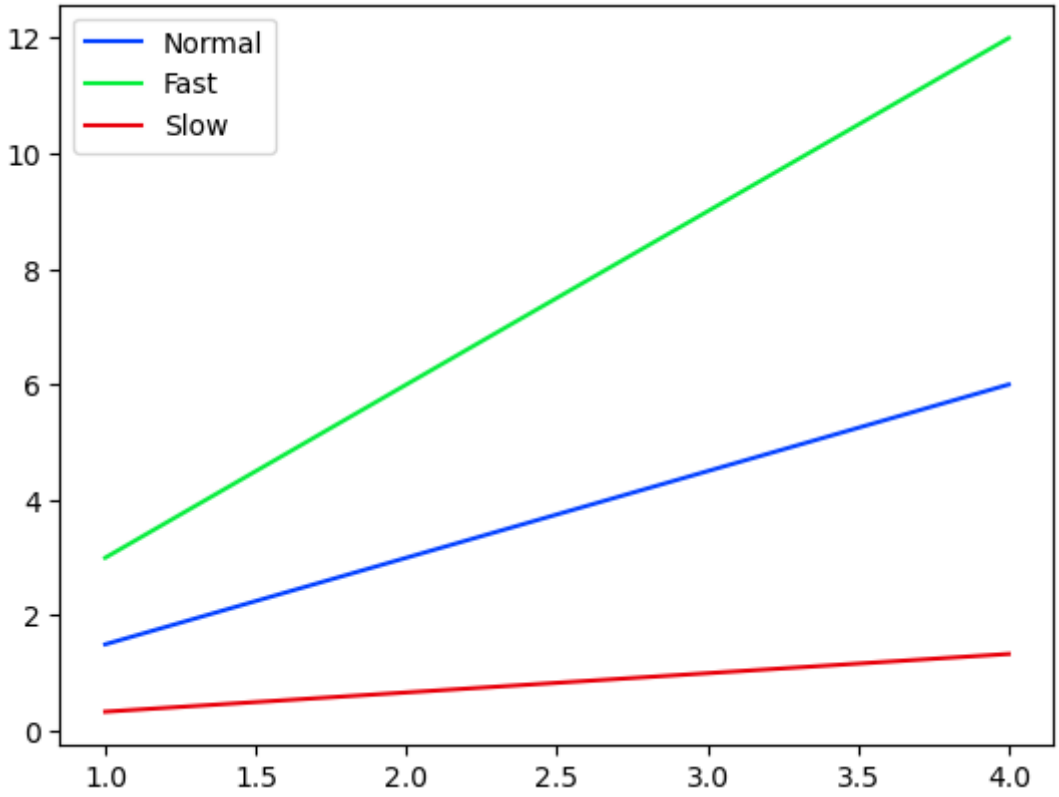
In [63]:
```
x15 = np.arange(1,5)
fig,ax = plt.subplots()
ax.plot(x15, x15*1.5)
ax.plot(x15, x15*3.0)
ax.plot(x15, x15/3.0)

ax.legend(['Normal','Fast','Slow'])
```

Out[63]:
```
<matplotlib.legend.Legend at 0x27698864f70>
```



In [65]:

```
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();
```



ax.legend(loc=0) # let Matplotlib decide the optimal location

ax.legend(loc=1) # upper right corner

ax.legend(loc=2) # upper left corner

ax.legend(loc=3) # lower left corner

ax.legend(loc=4) # lower right corner

ax.legend(loc=5) # right

ax.legend(loc=6) # center left

ax.legend(loc=7) # center right

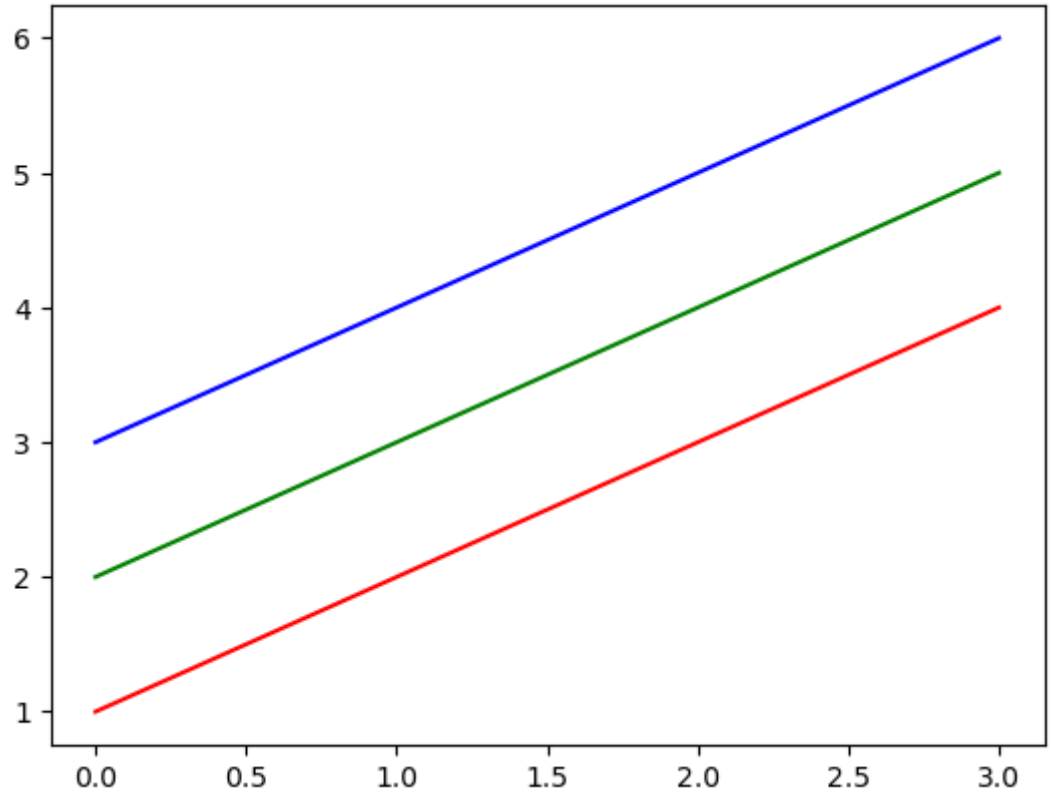
ax.legend(loc=8) # lower center

ax.legend(loc=9) # upper center

ax.legend(loc=10) # center

In [66]:

```
x16 = np.arange(1,5)

plt.plot(x16, 'r')
plt.plot(x16+1, 'g')
plt.plot(x16+2, 'b')

plt.show()
```



Colour abbreviation Colour name

b blue

c cyan

g green

k black

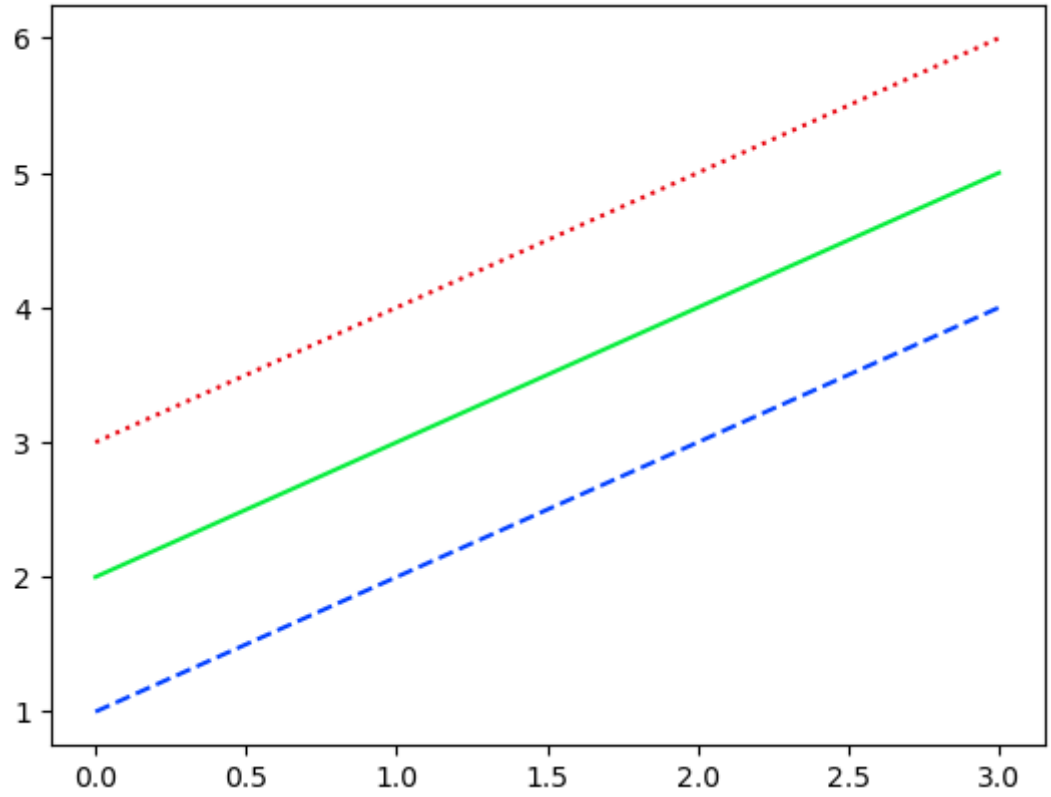
m magenta

r red

w white

y yellow

In [67]:
```
plt.plot(x16, '--', x16+1, '-', x16+2, ':')
plt.show()
```



Style abbreviation Style

solid line -- dashed line

-. dash-dot line

: dotted line