In [55]:
```
import matplotlib
```

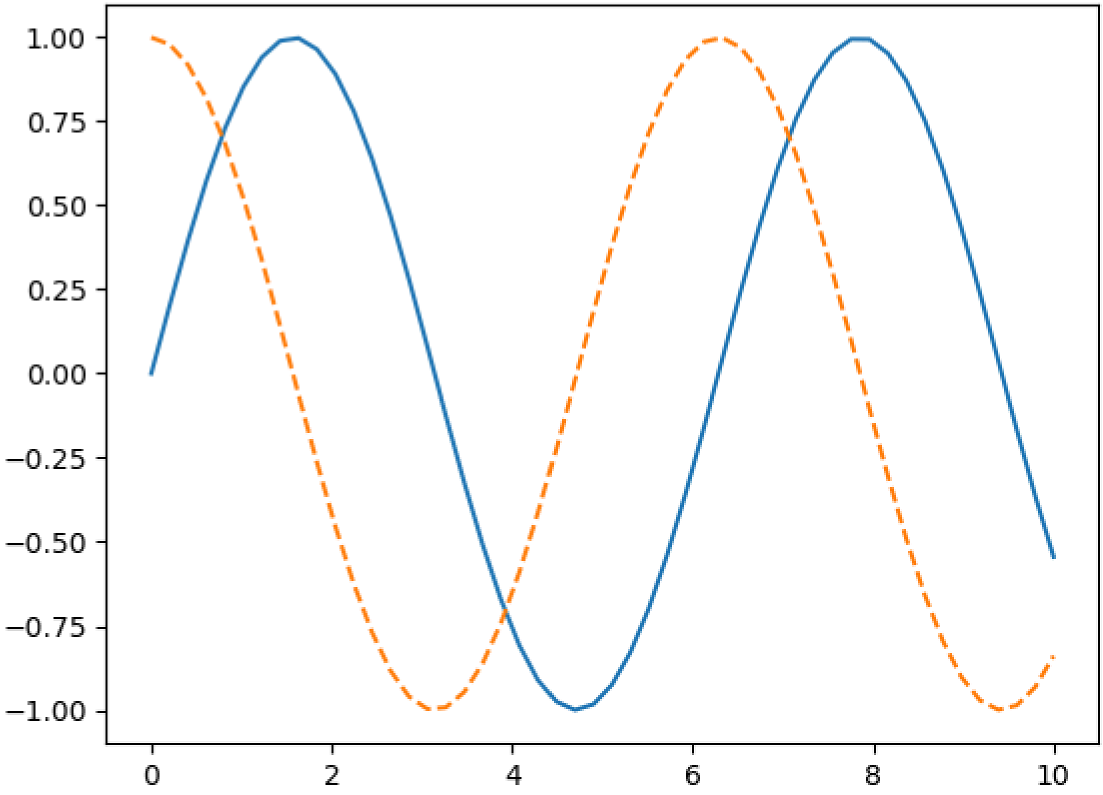
In [56]:
```
import matplotlib.pyplot
```

In [57]:
```
import matplotlib.pyplot as plt
```

In [58]:
```
# Import dependencies
import numpy as np
import pandas as pd
```

In [59]:
```
# Import Matplotlib
import matplotlib.pyplot as plt
```

In [60]:
```
%matplotlib inline
x1 = np.linspace(0, 10, 50)

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

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



In [61]:
```
plt.gcf ( )
```

Out[61]:
```
<Figure size 640x480 with 0 Axes>
```

In [62]:
```
plt.gca ( )
```

Out[62]:
```
<Axes: >
```

In [65]:
```
plt.subplot(2, 1, 1)
plt.plot(x1, np.cos(x1),  '*')
plt.show
```

Out[65]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

In [66]:
```
plt.figure()

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

plt.subplot(2, 1, 2)
plt.plot(x1, np.cos(x1));
plt.show
```

Out[66]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

In [30]:
```
print(plt.gcf( ))
```

```
Figure(640x480)
```

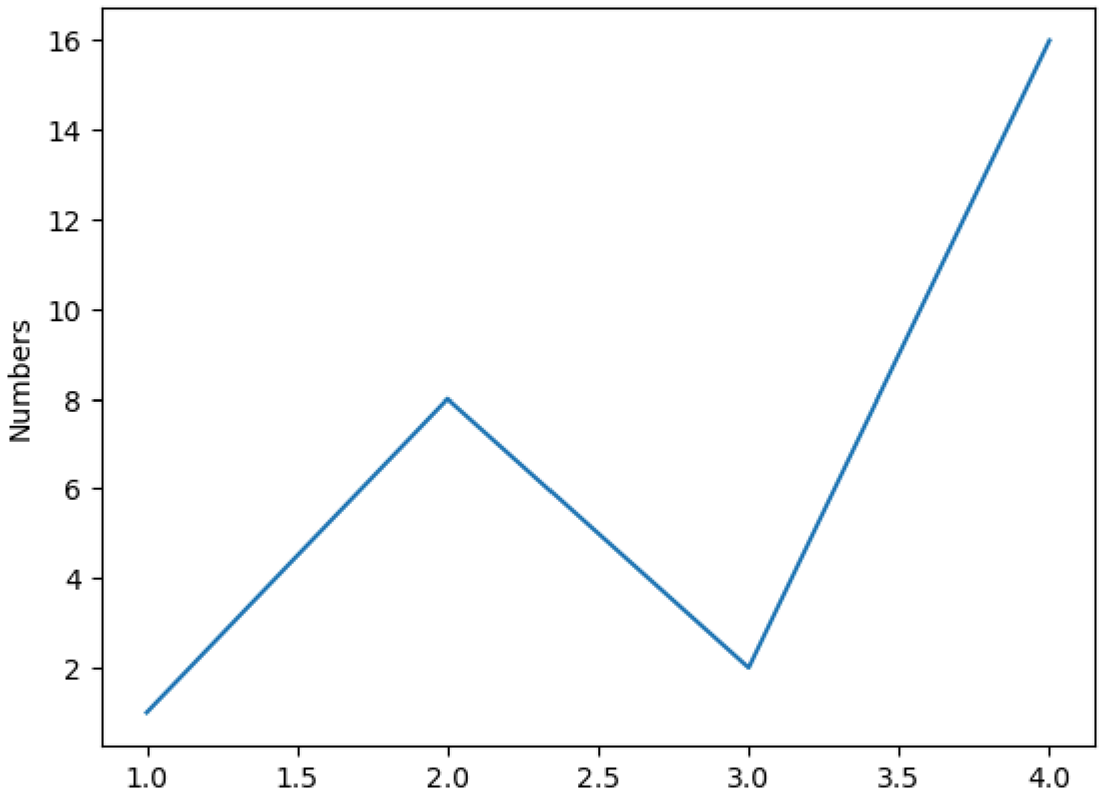
In [31]:
```
print(plt.gcf( ))
```

```
Figure(640x480)
```

In [36]:
```
print(plt.gca())
```

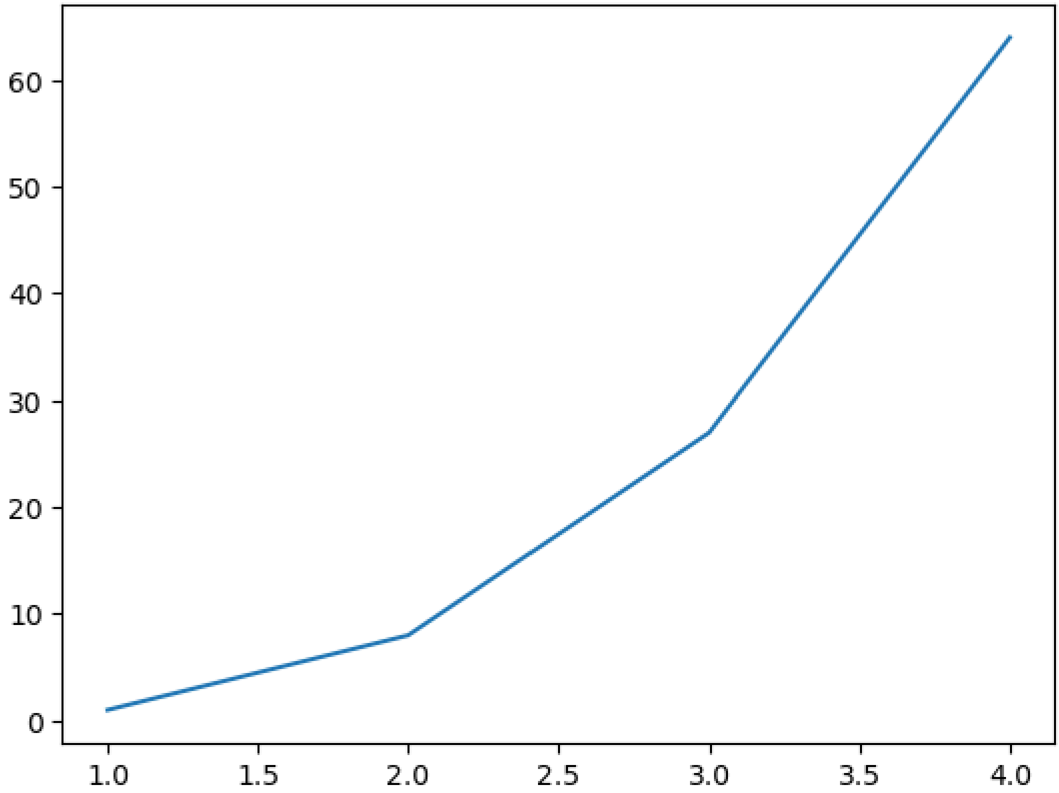
```
Axes(0.125,0.11;0.775x0.77)
```

In [35]:
```
plt.plot([1,2,3,4], [1,8,2,16])
plt.ylabel('Numbers')
plt.show()
```



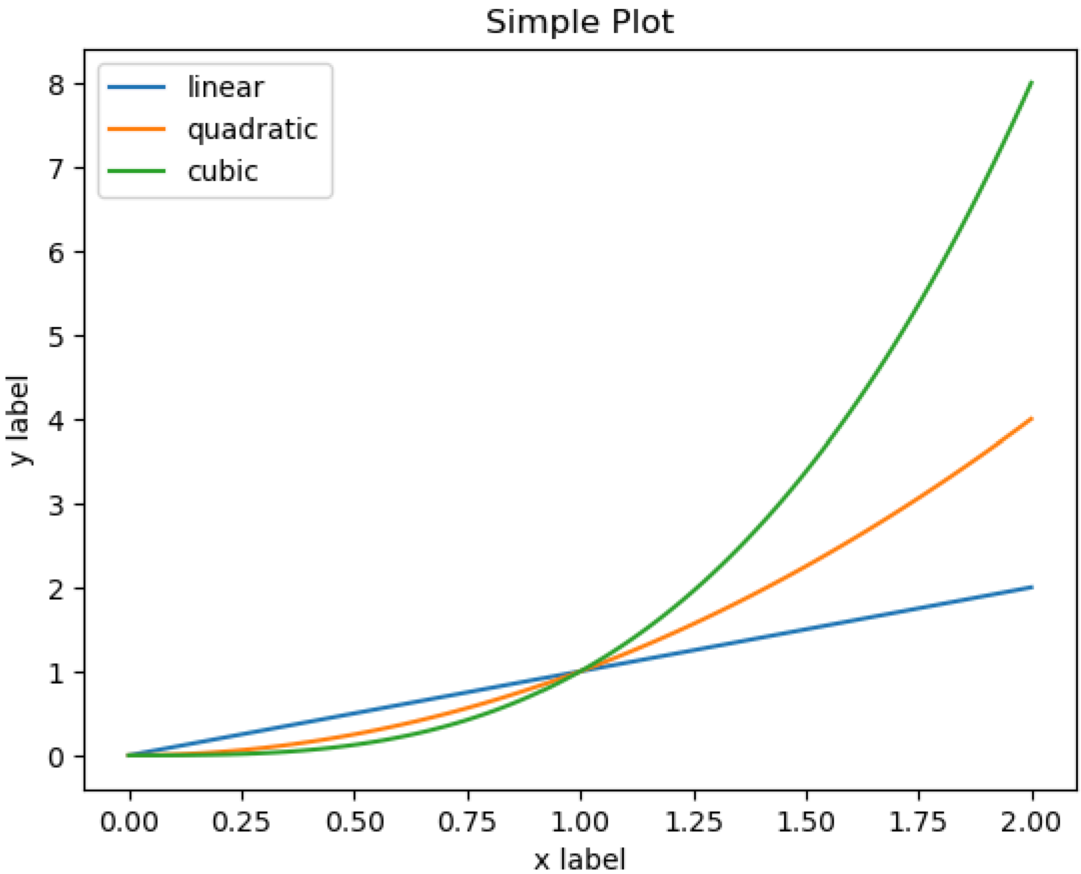
In [34]:
```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4], [1, 8, 27, 64])
```

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

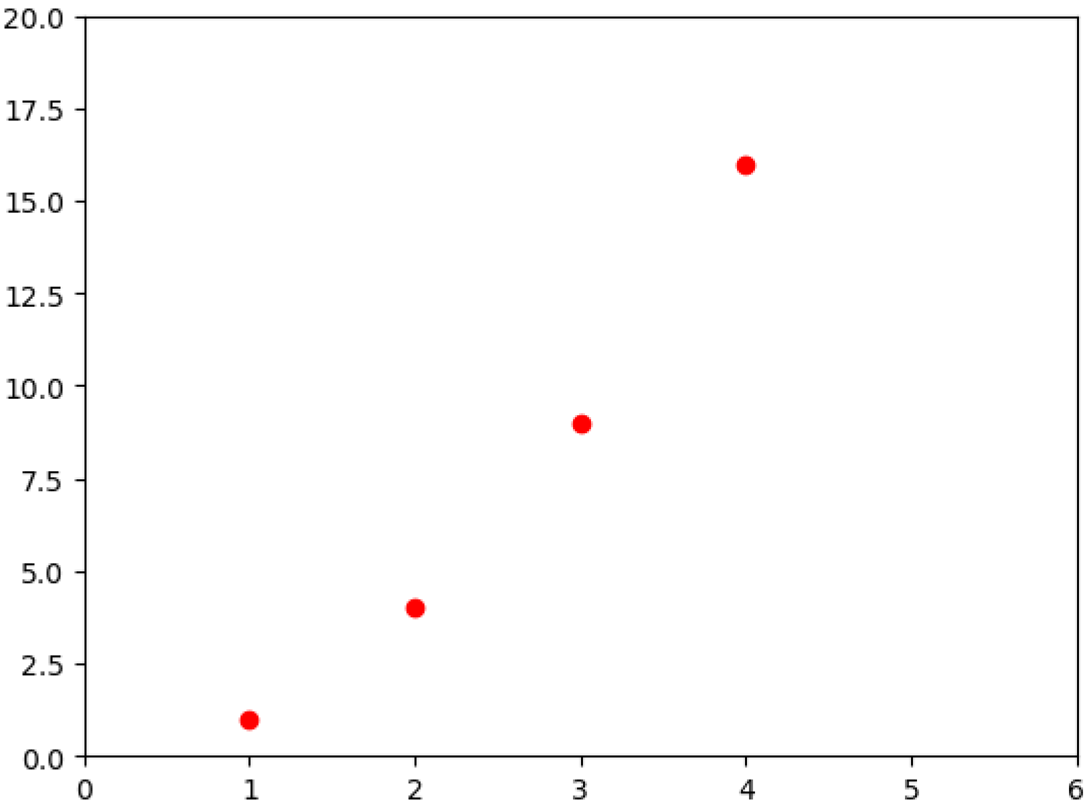


In [38]:

```python
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()
```

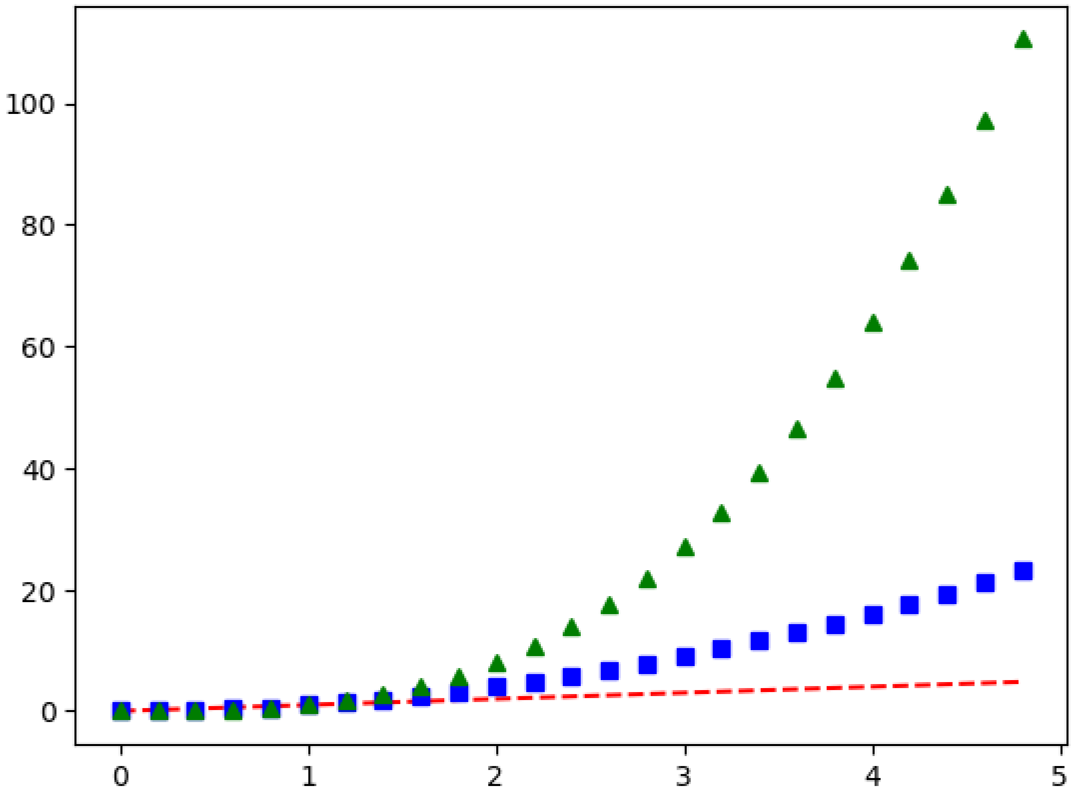


In [39]:
```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro' )
plt.axis([0, 6, 0, 20])
plt.show()
```



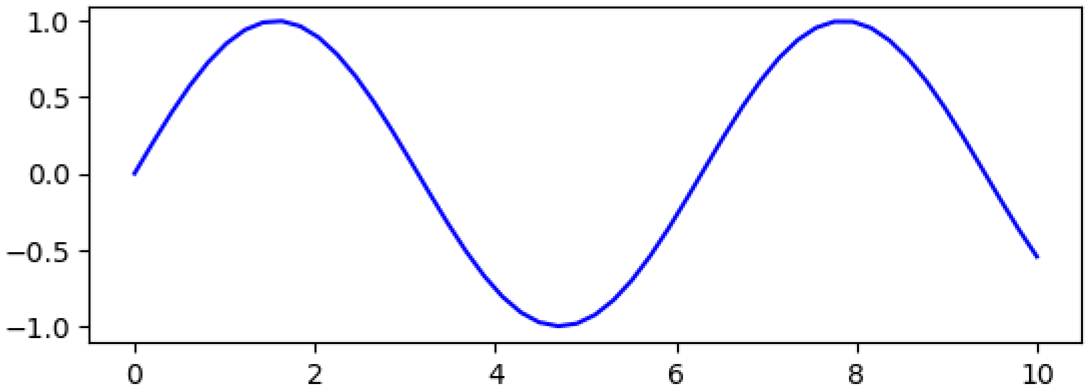
In [40]:
```
# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)
```

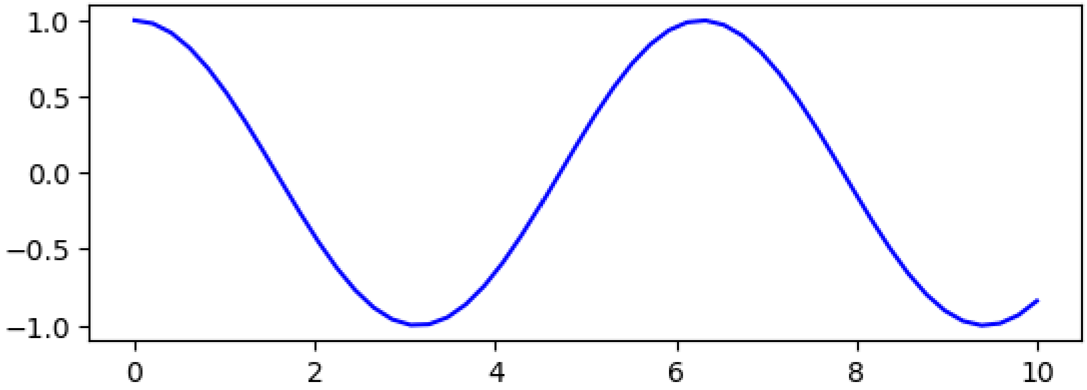
```
# red dashes, blue squares and green triangles
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```



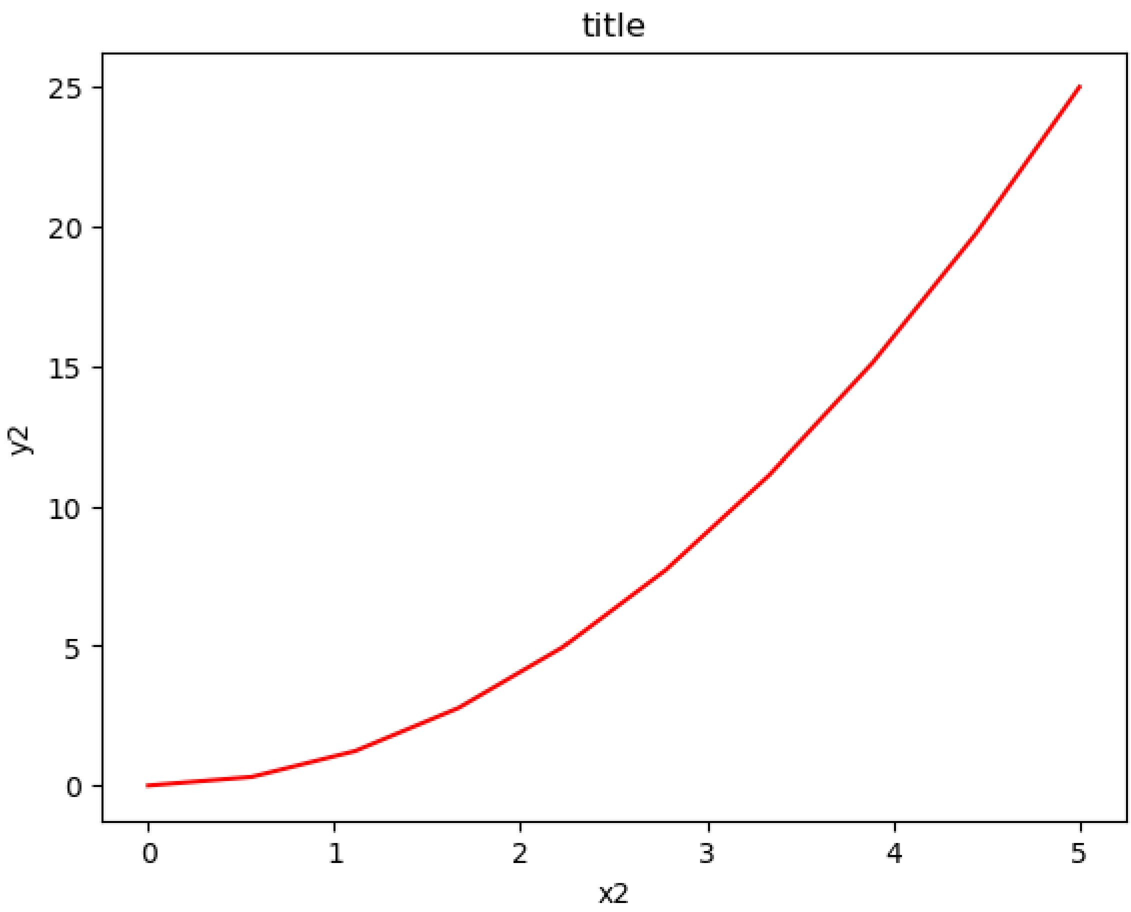
In [53]:

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

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





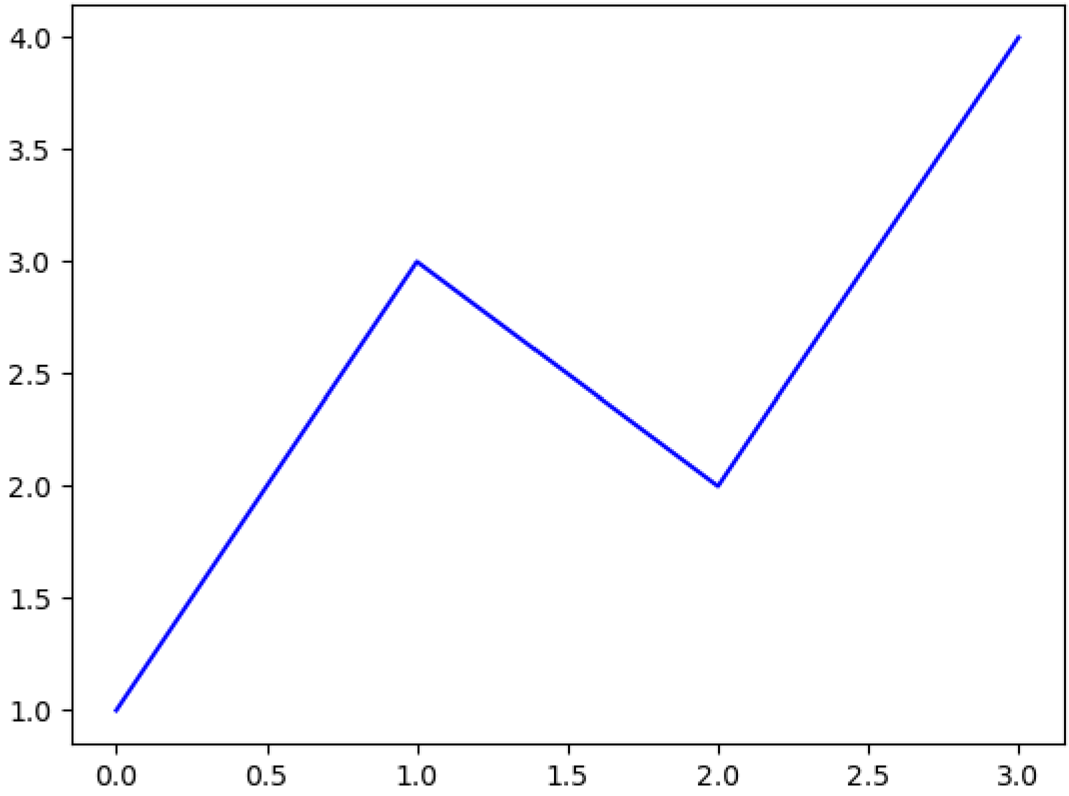
In [54]:
```
fig = plt.figure()

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');
plt.show()
```



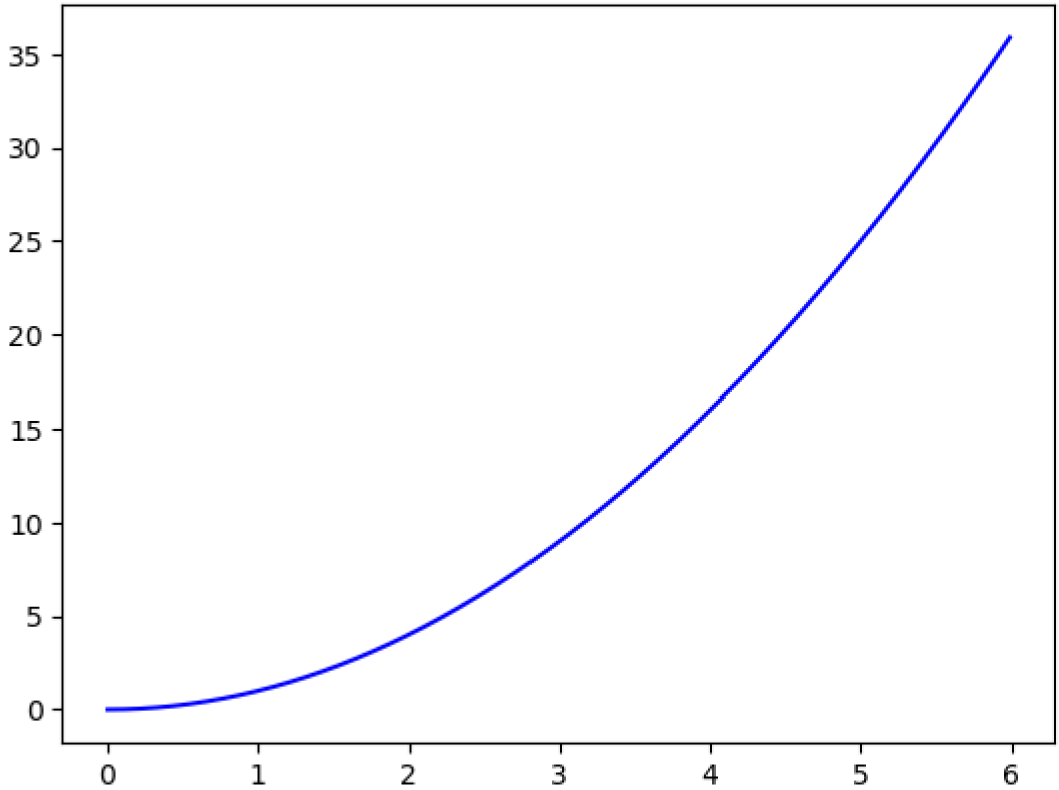
In [68]:
```
fig = plt.figure()

ax = plt.axes()
```

In [70]:
```
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)
```

In [72]:
```
plt.plot([1, 3, 2, 4], 'b-')

plt.show( )
```

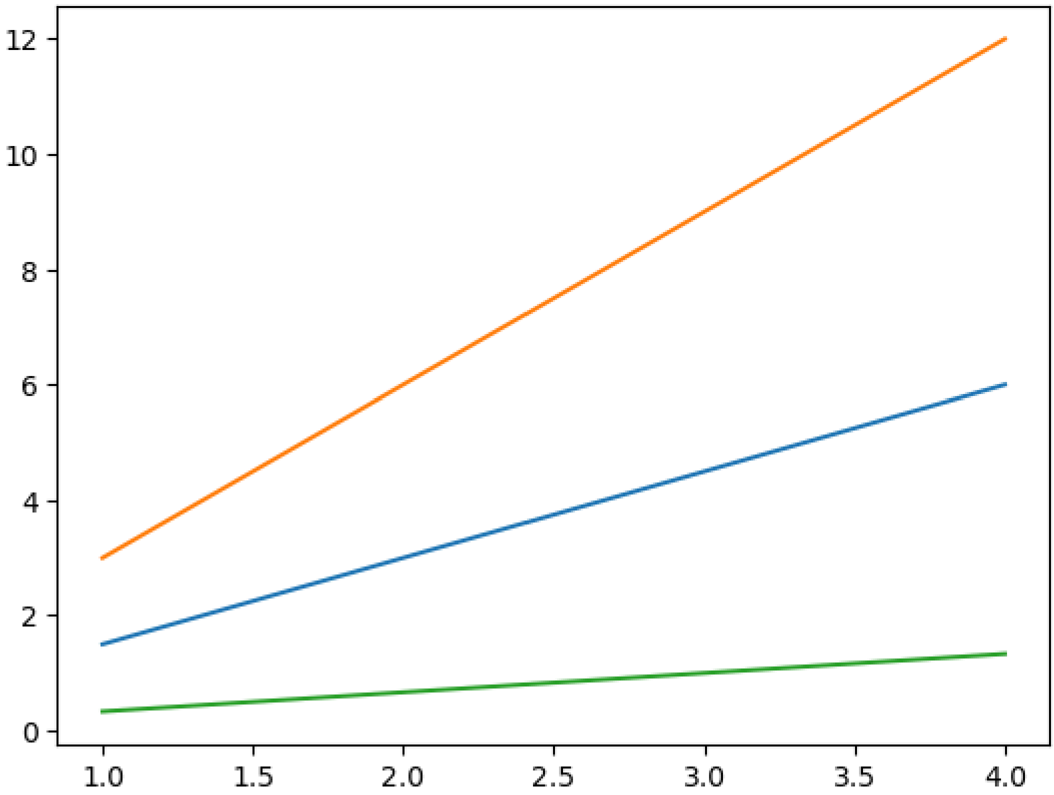


In [73]:
```
x3 = np.arange(0.0, 6.0, 0.01)

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

plt.show()
```



In [74]:
```
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()
```



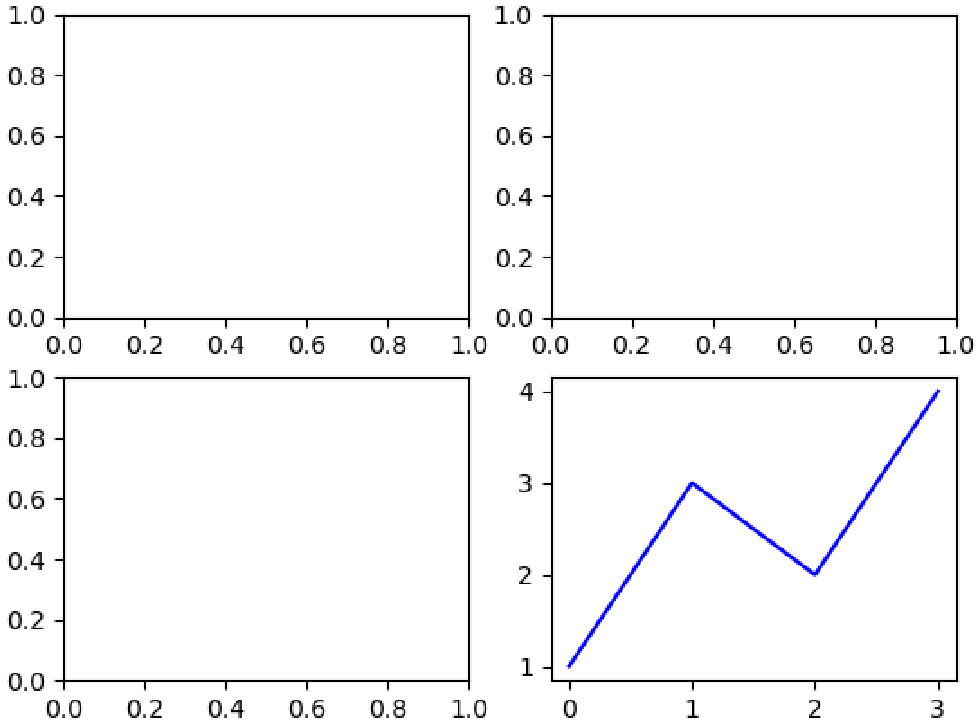
In [75]:

```
# Saving the figure

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

In [78]:

```
# Explore the contents of figure

from IPython.display import Image

Image('plot1.png')
```

Out[78]:



In [77]:

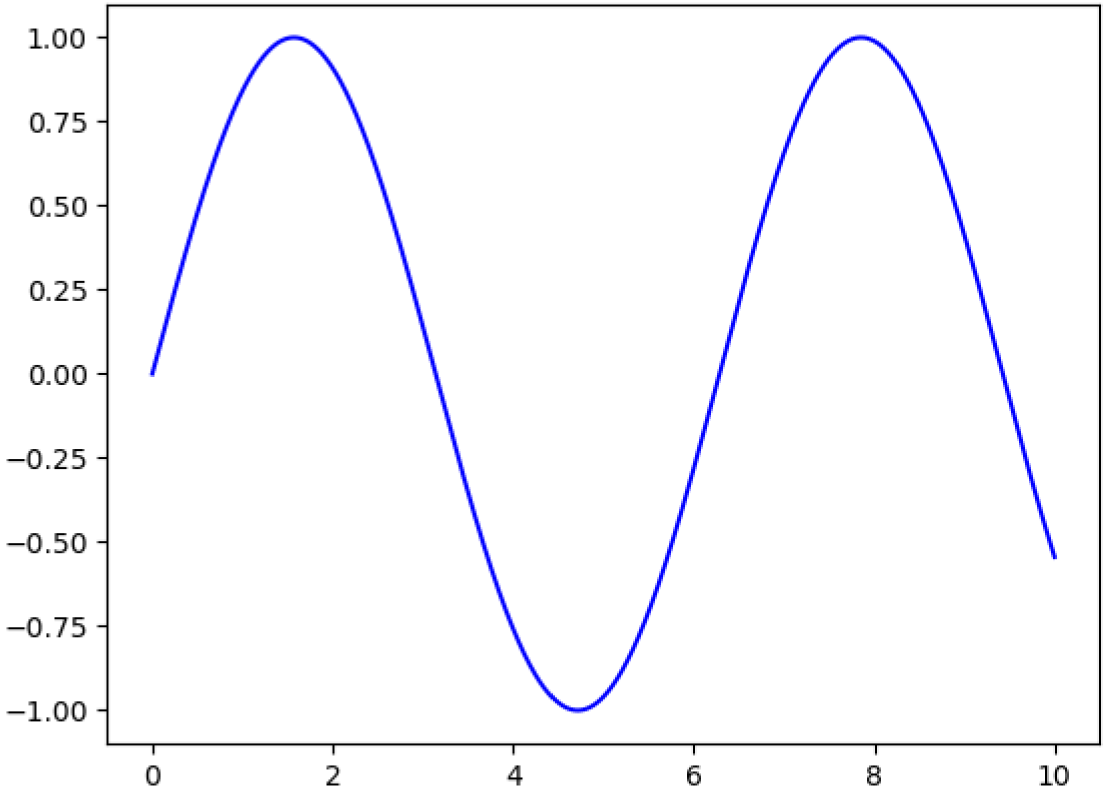
```
# Explore supported file formats


fig.canvas.get_supported_filetypes()
```

Out[77]:

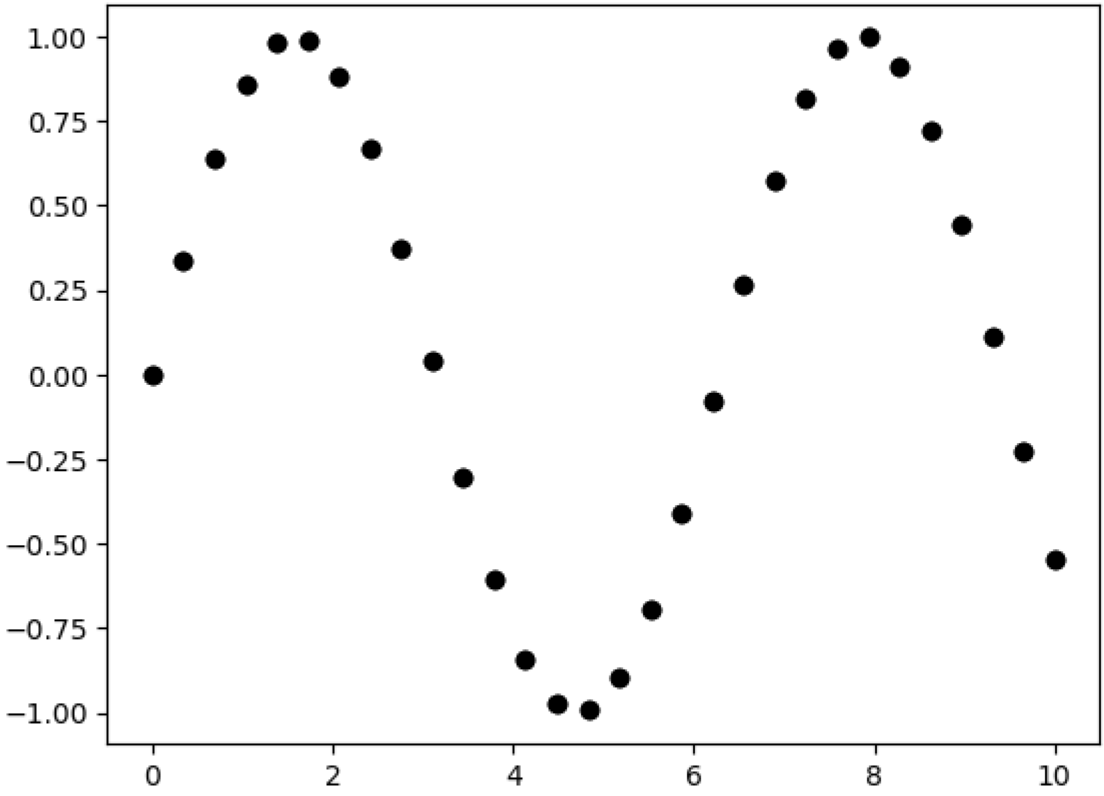
```
{'eps': 'Encapsulated Postscript',
 'jpg': 'Joint Photographic Experts Group',
 'jpeg': 'Joint Photographic Experts Group',
 'pdf': 'Portable Document Format',
 'pgf': 'PGF code for LaTeX',
 'png': 'Portable Network Graphics',
 'ps': 'Postscript',
 'raw': 'Raw RGBA bitmap',
 'rgba': 'Raw RGBA bitmap',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

In [83]:

```
# 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-');
plt.show()
```

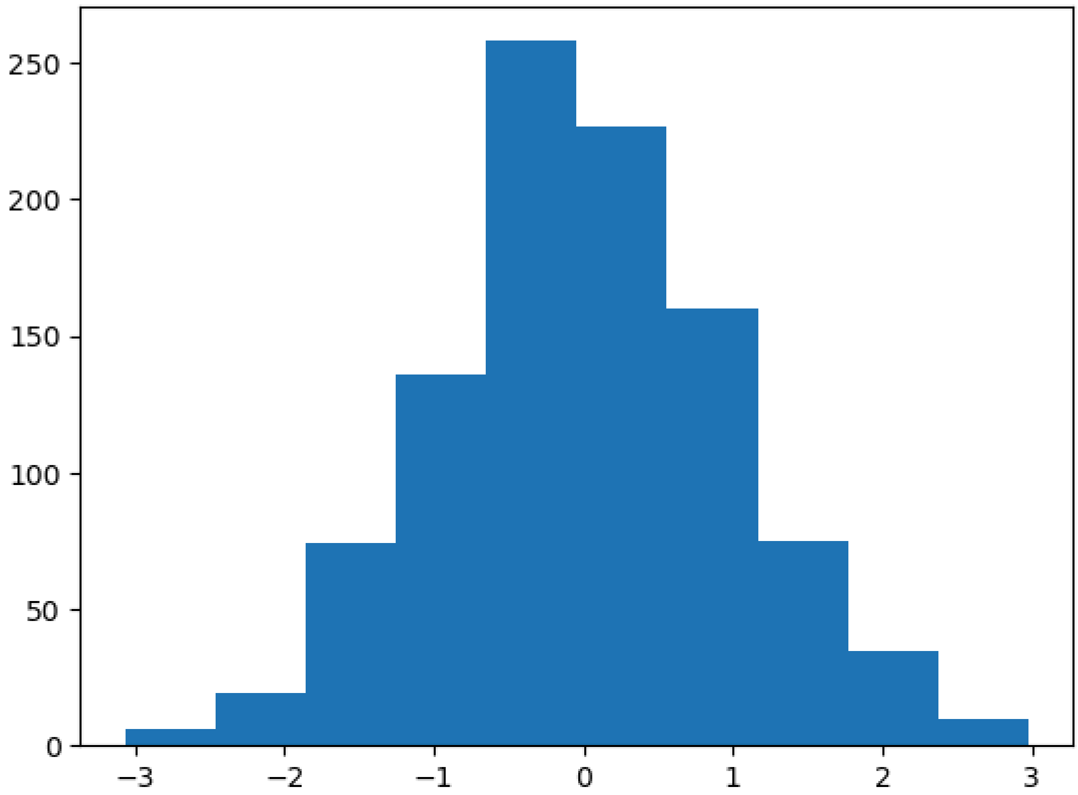


In [85]:
```
x7 = np.linspace(0, 10, 30)

y7 = np.sin(x7)

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

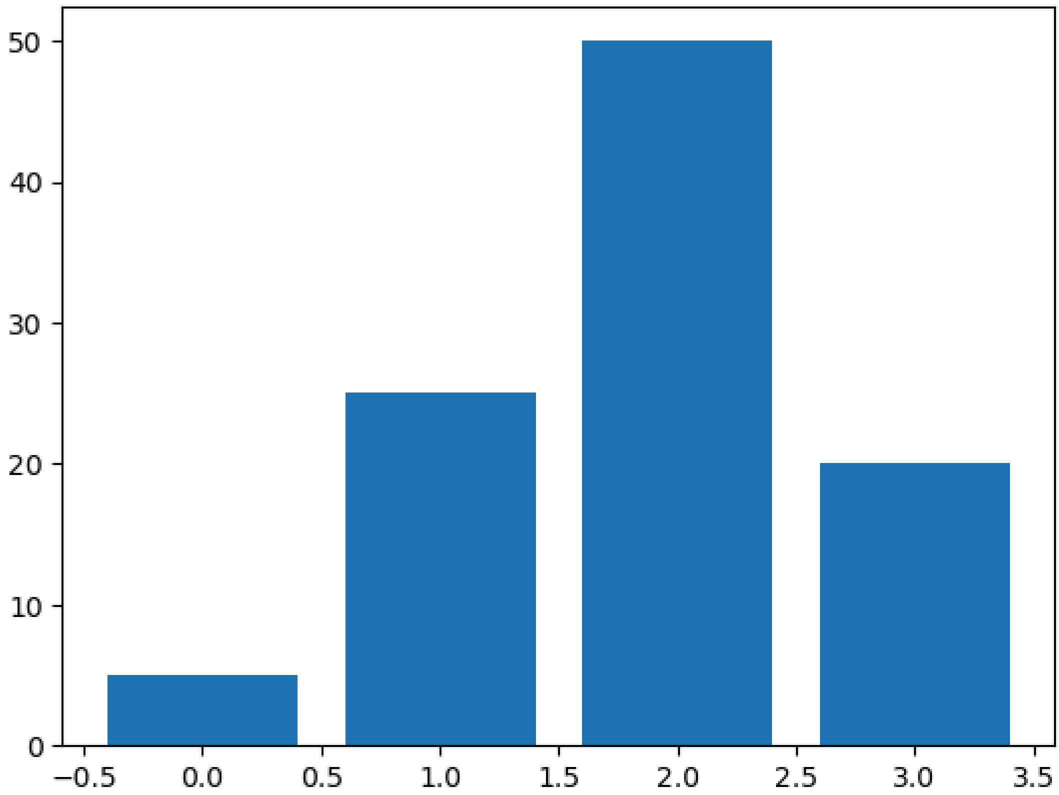


In [88]:
```
data1 = np.random.randn(1000)
```

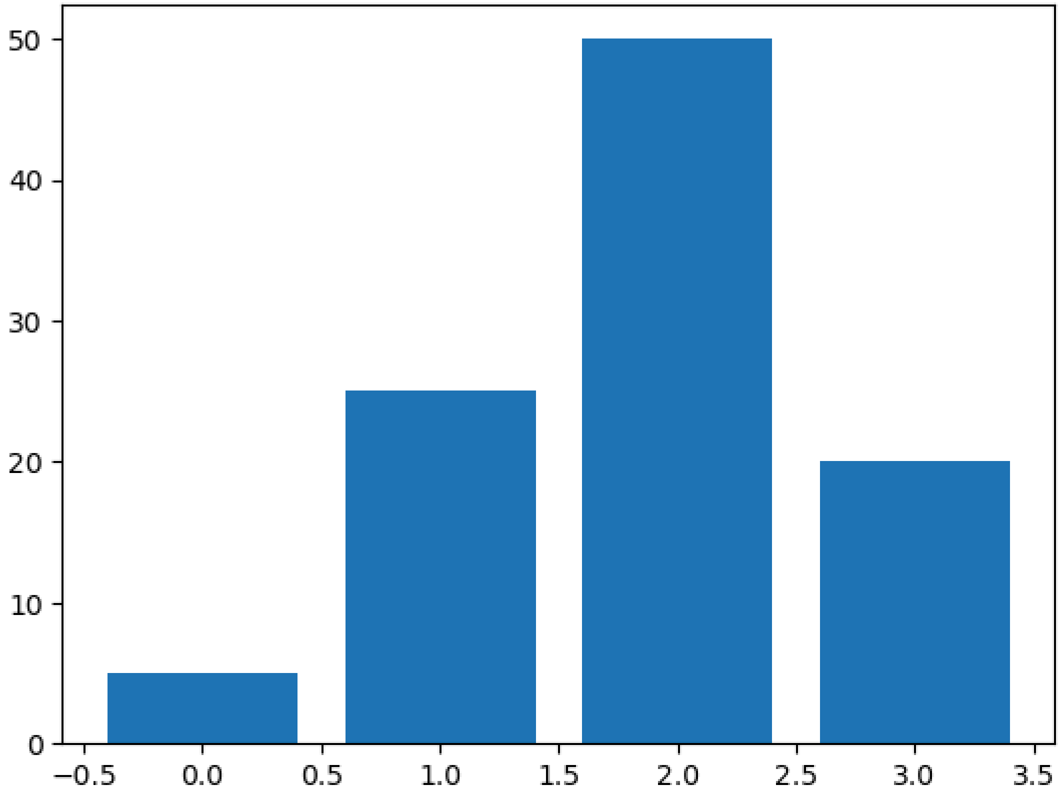
```
plt.hist(data1);
plt.show()
```



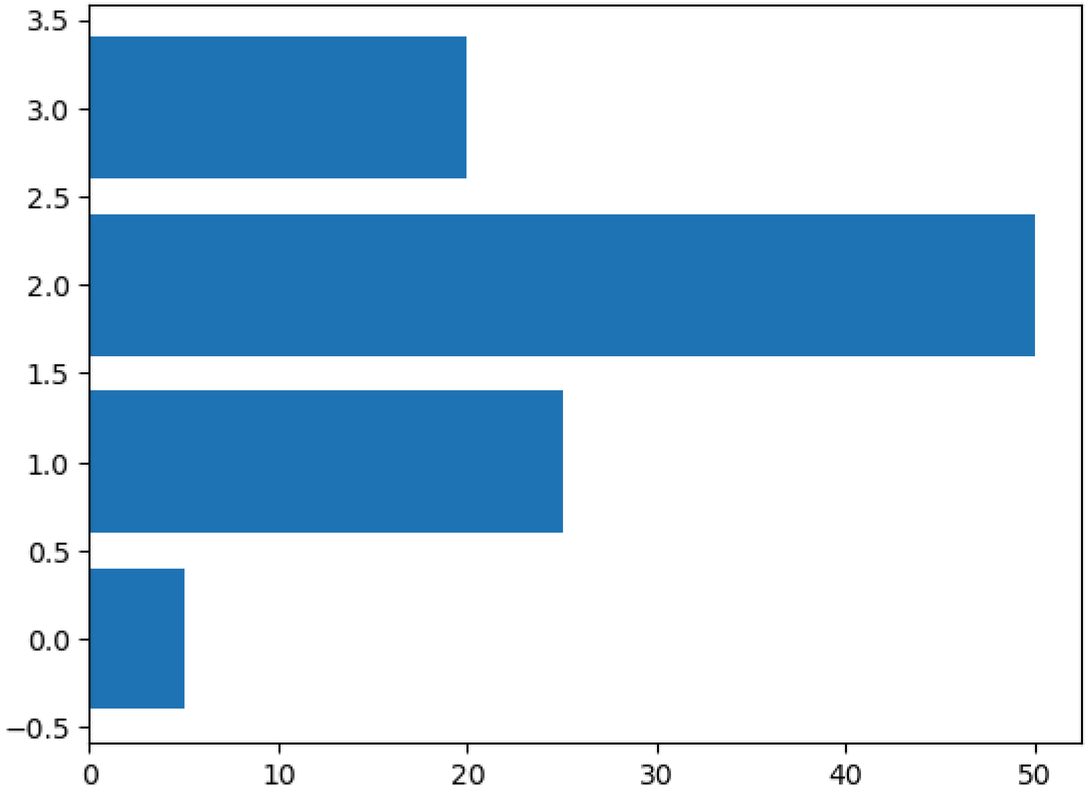
In [89]:
```
data2 = [5. , 25. , 50. , 20.]

plt.bar(range(len(data2)), data2)

plt.show()
```



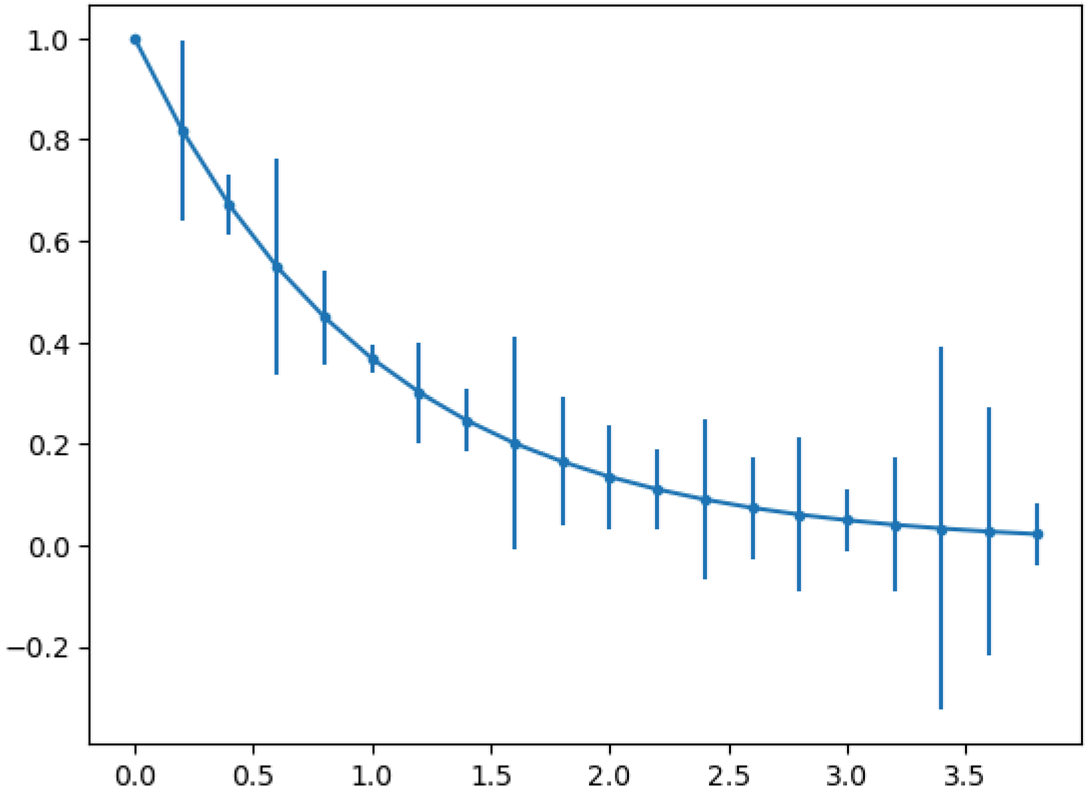
In [90]:
```
data2 = [5. , 25. , 50. , 20.]

plt.bar(range(len(data2)), data2)

plt.show()
```



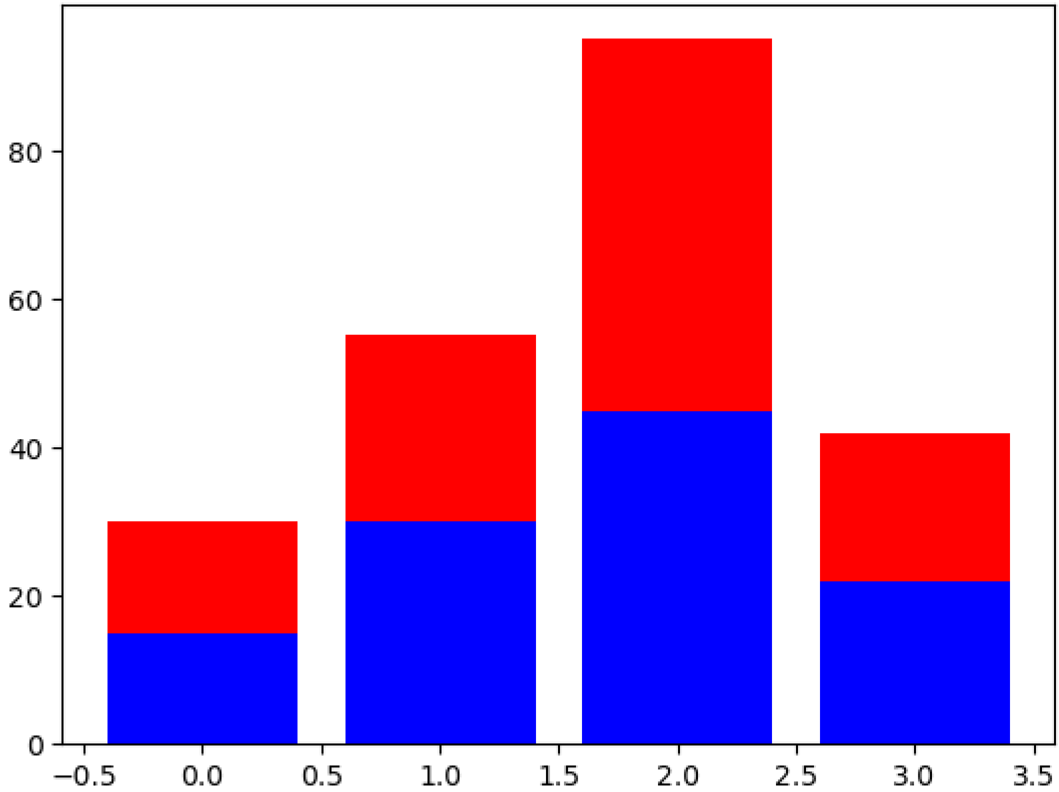
In [91]:
```
data2 = [5. , 25. , 50. , 20.]

plt.barh(range(len(data2)), data2)

plt.show()
```



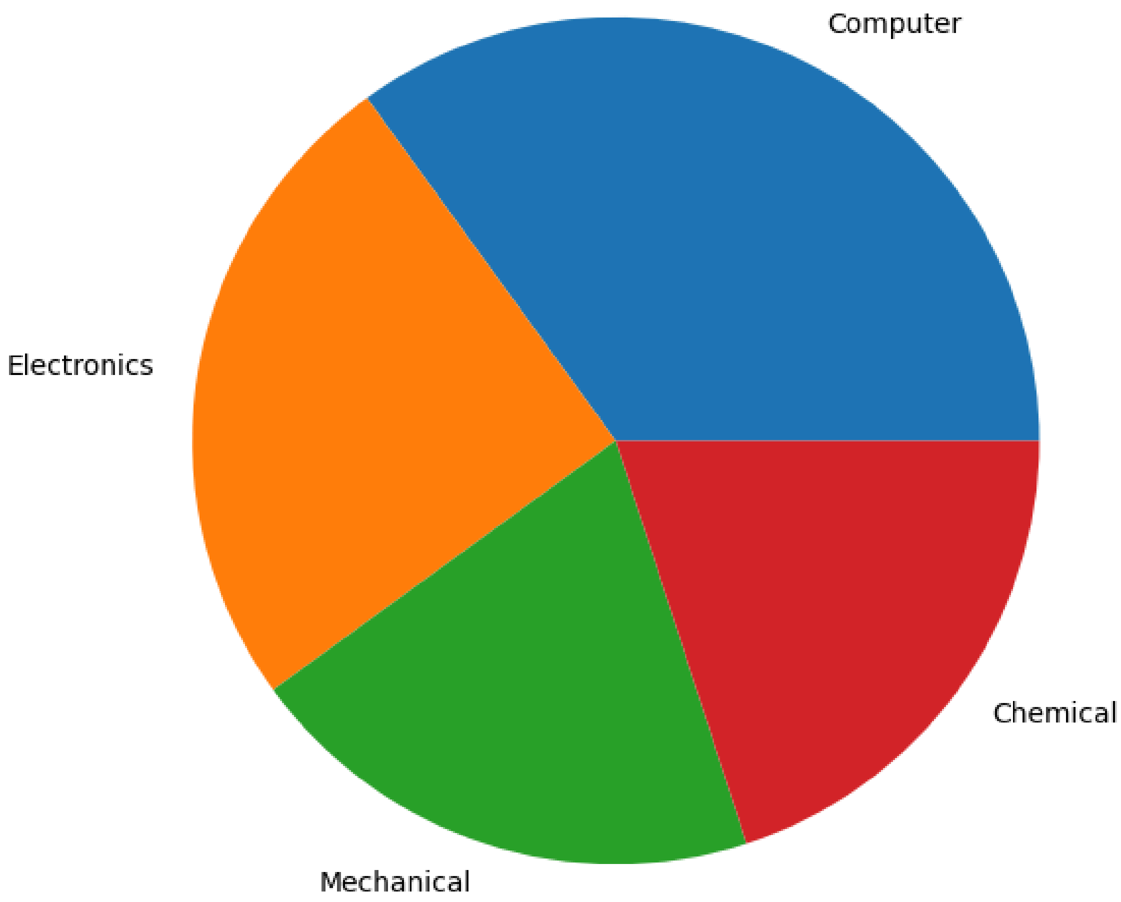
In [92]:
```
x9 = np.arange(0, 4, 0.2)

y9 = np.exp(-x9)

e1 = 0.1 * np.abs(np.random.randn(len(y9)))

plt.errorbar(x9, y9, yerr = e1, fmt = '.-')

plt.show();
```



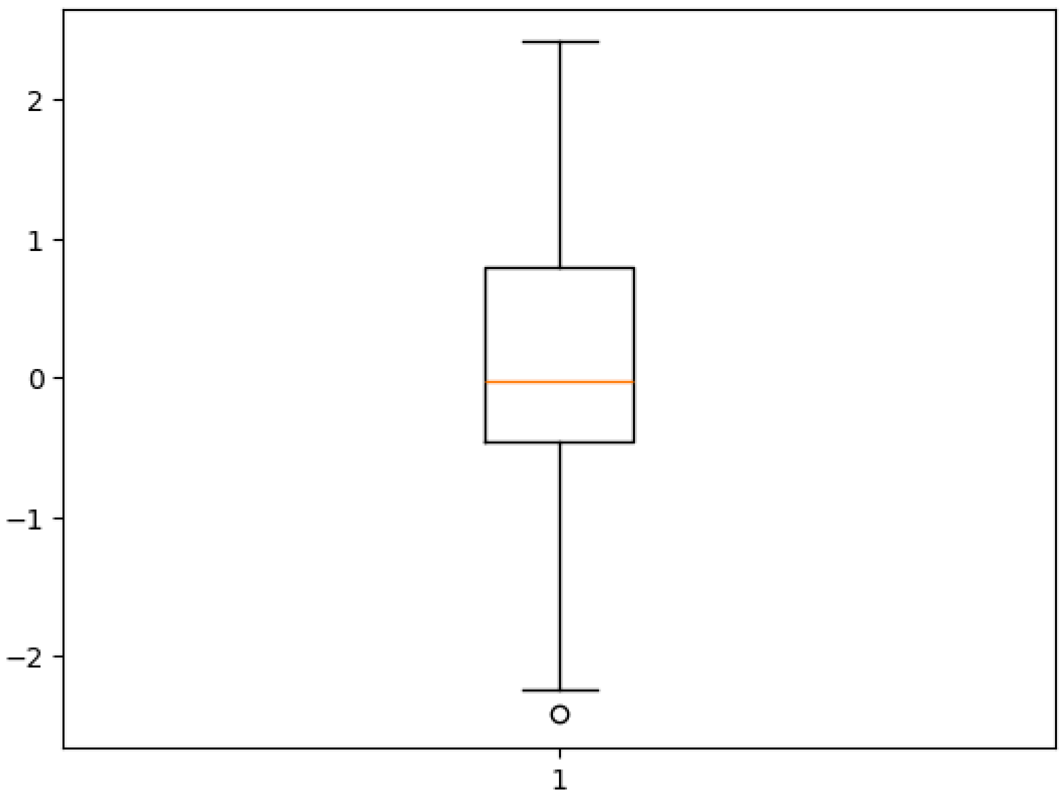
In [93]:
```
A = [15., 30., 45., 22.]

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

z2 = range(4)

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

plt.show()
```



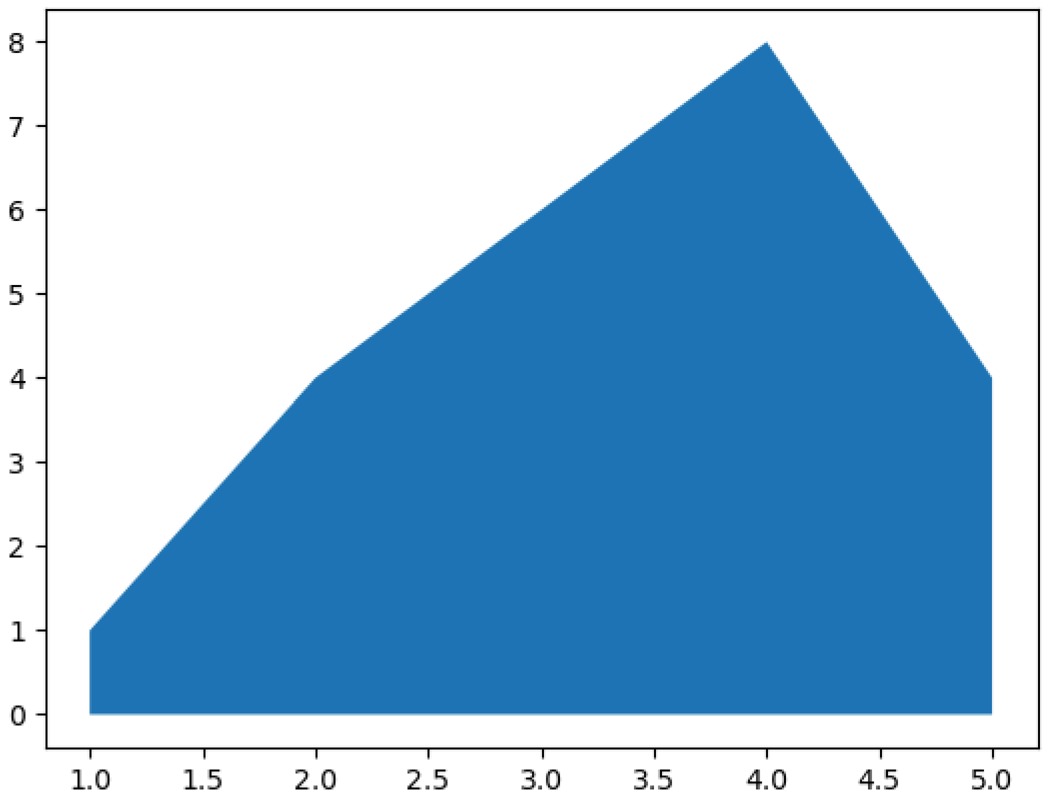
In [94]:
```
plt.figure(figsize=(7,7))

x10 = [35, 25, 20, 20]

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

plt.pie(x10, labels=labels);

plt.show()
```



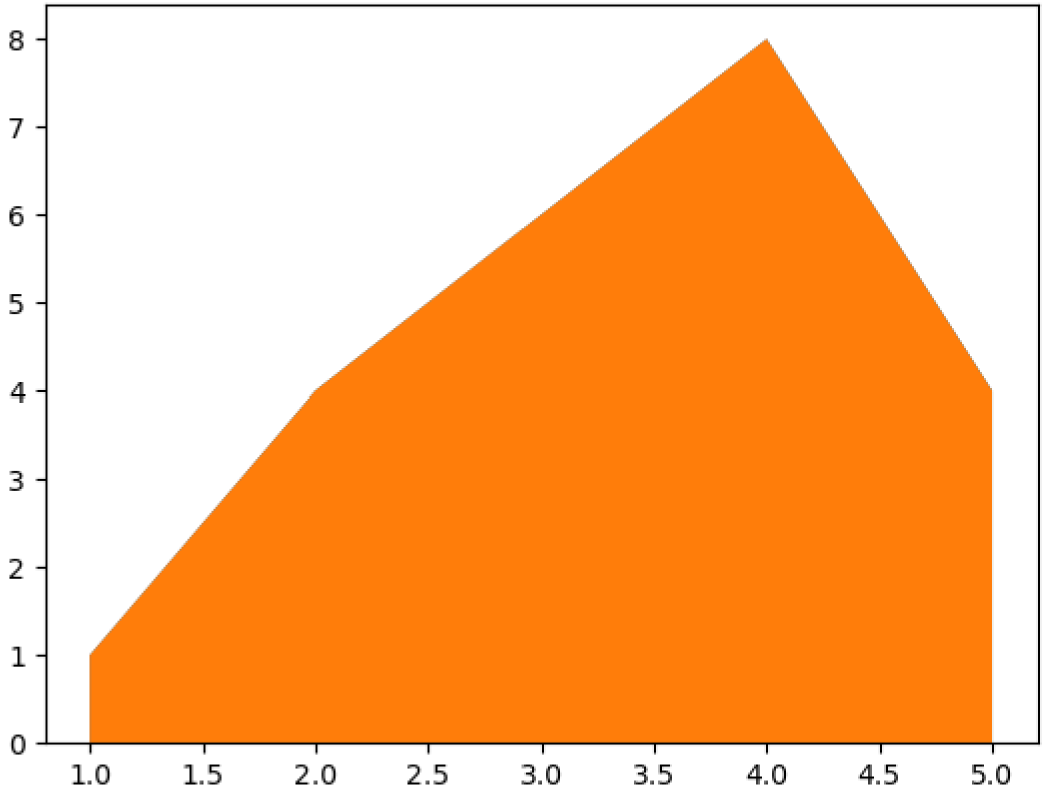
In [95]:
```
data3 = np.random.randn(100)

plt.boxplot(data3)

plt.show();
```



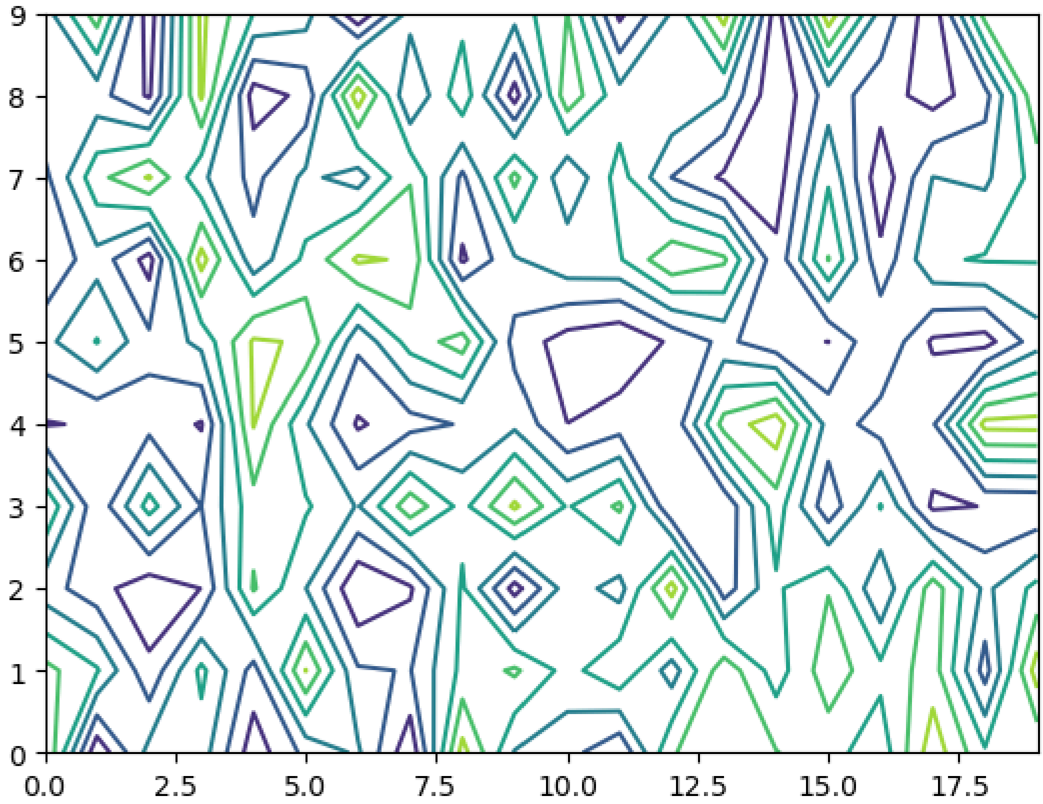
In [96]:
```
# Create some data
x12 = range(1, 6)
y12 = [1, 4, 6, 8, 4]

# Area plot
plt.fill_between(x12, y12)
plt.show()
```



In [98]:
```
plt.stackplot(x12, y12)
plt.show()
```



In [99]:
```
# Create a matrix
matrix1 = np.random.rand(10, 20)

cp = plt.contour(matrix1)

plt.show()
```



In [104…

```
print(plt.style.available)
```

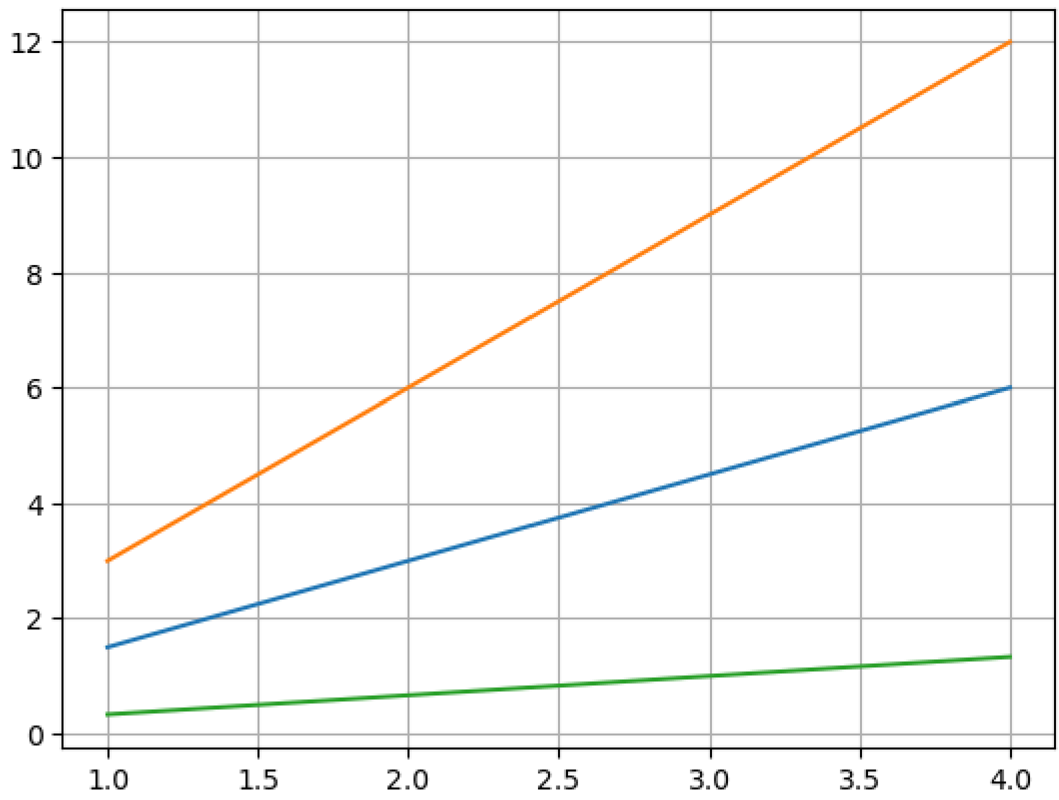
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'bmh', 'classic', 'dark_background', 'fast', 'fivethirtyeight', 'ggplot', 'grayscale', 'petroff10', 'seaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblind', 'seaborn-v0_8-dark', 'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-deep', 'seaborn-v0_8-muted', 'seaborn-v0_8-notebook', 'seaborn-v0_8-paper', 'seaborn-v0_8-pastel', 'seaborn-v0_8-poster', 'seaborn-v0_8-talk', 'seaborn-v0_8-ticks', 'seaborn-v0_8-white', 'seaborn-v0_8-whitegrid', 'tableau-colorblind10']

In [121…

```
plt.style.use('Solarize_Light2')
```

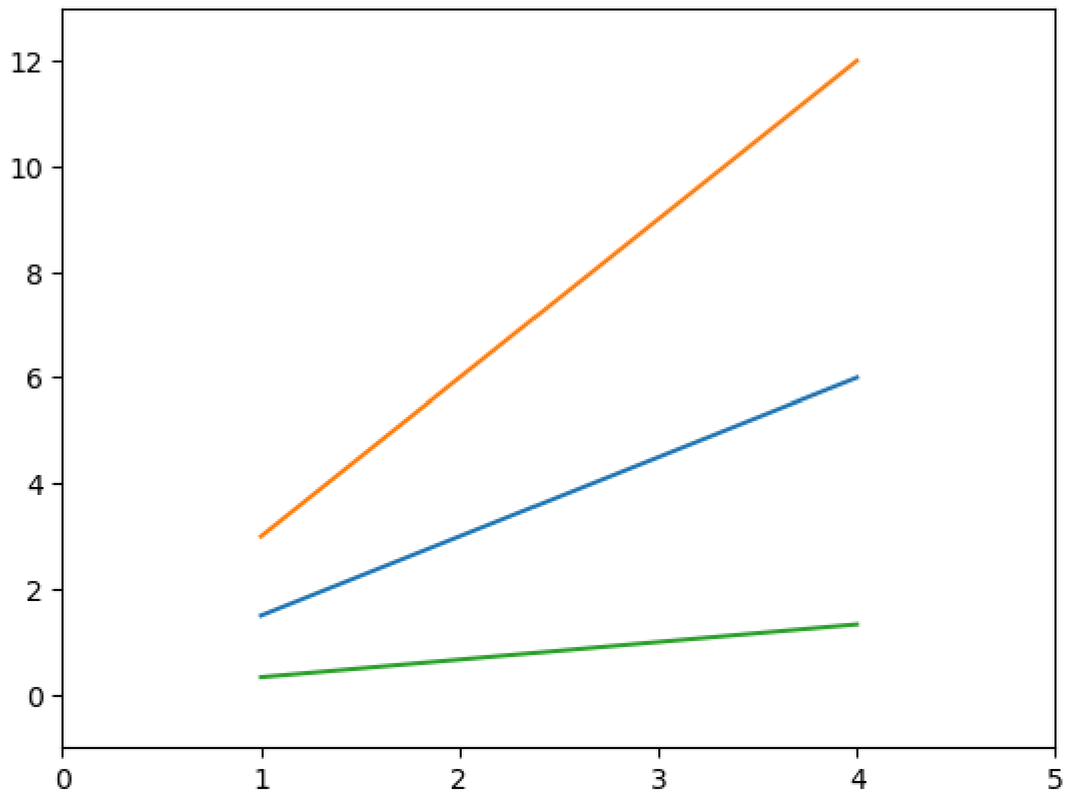
In [105…

```
x15 = np.arange(1, 5)

plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)

plt.grid(True)

plt.show()
```



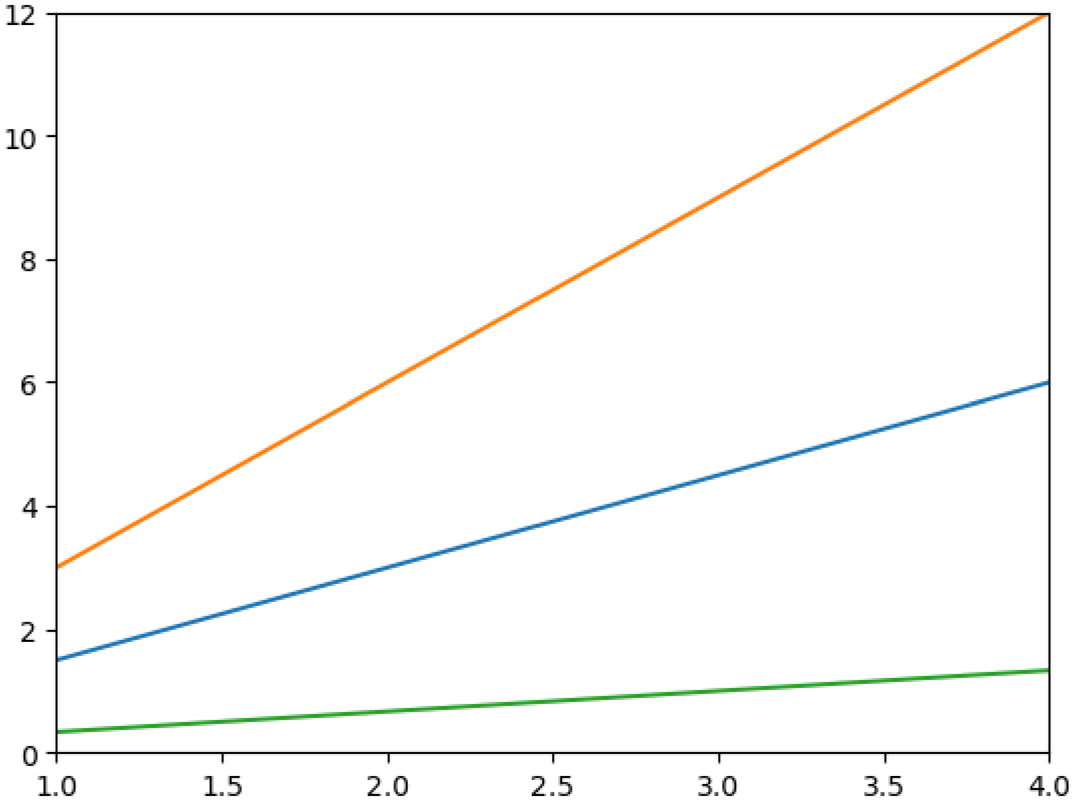
In [106…

```
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()
```



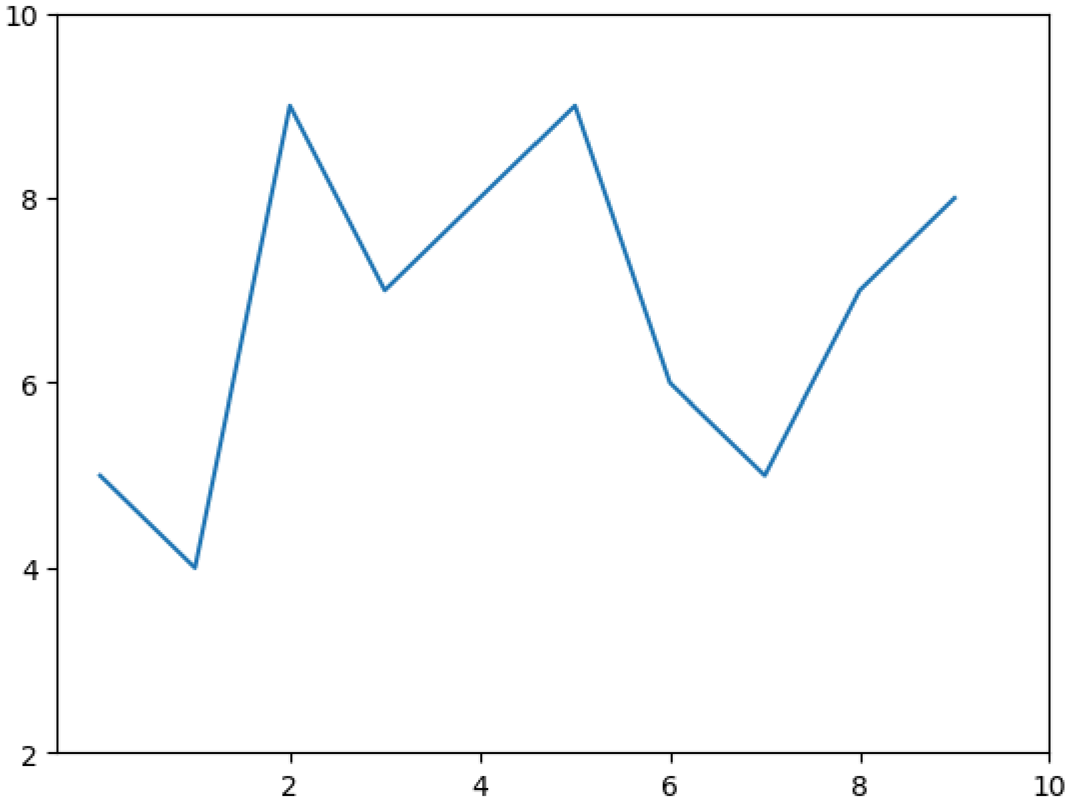
In [109…

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



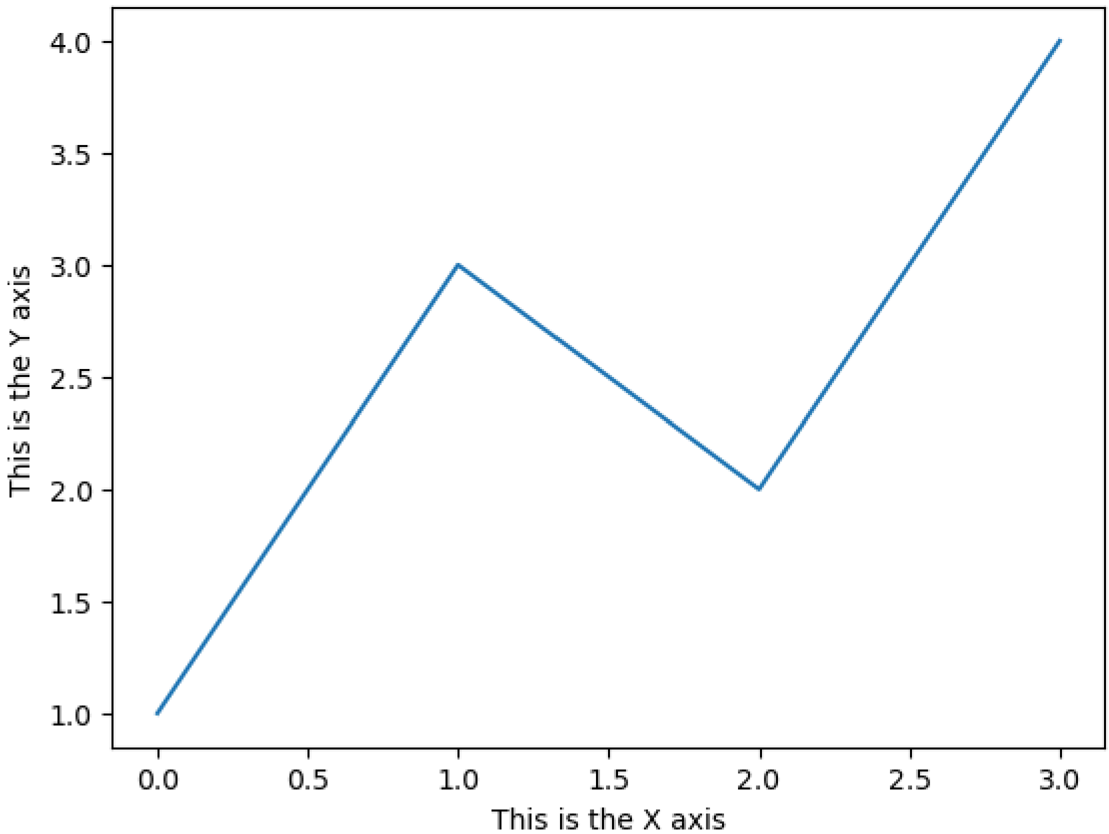
In [110…

```
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()
```



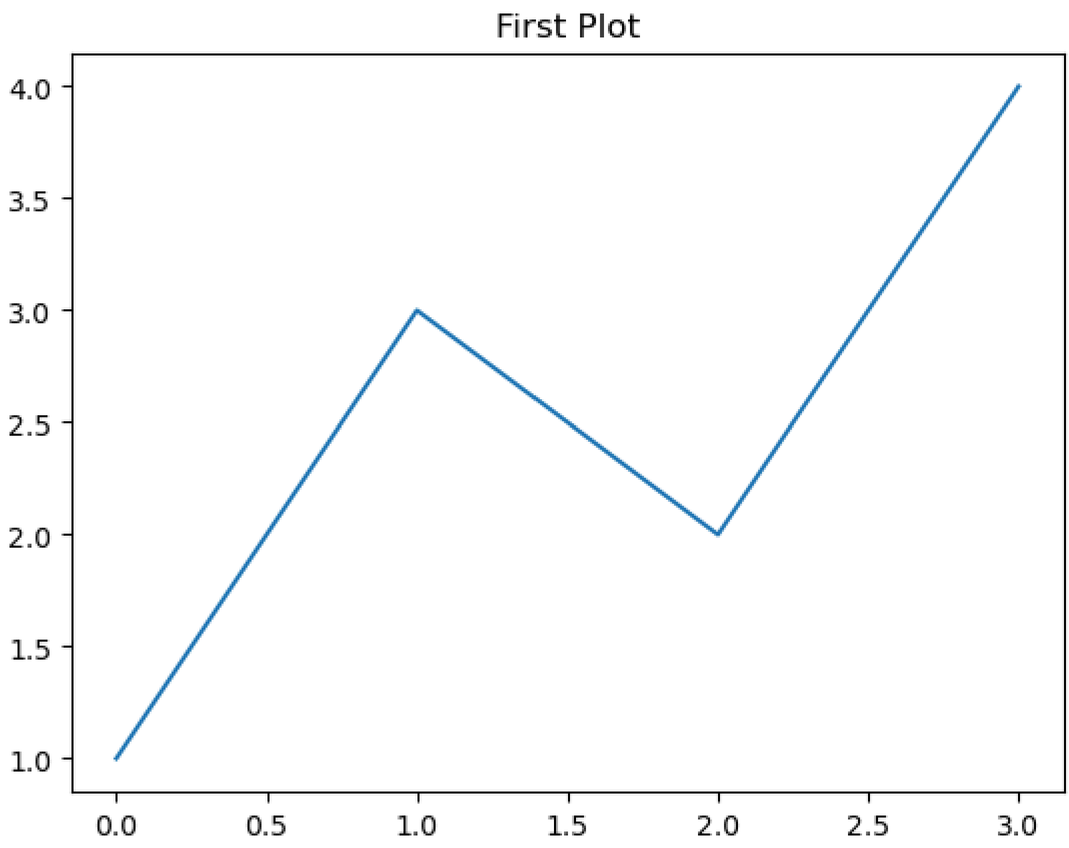
In [111…

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

plt.xlabel('This is the X axis')

plt.ylabel('This is the Y axis')

plt.show()
```



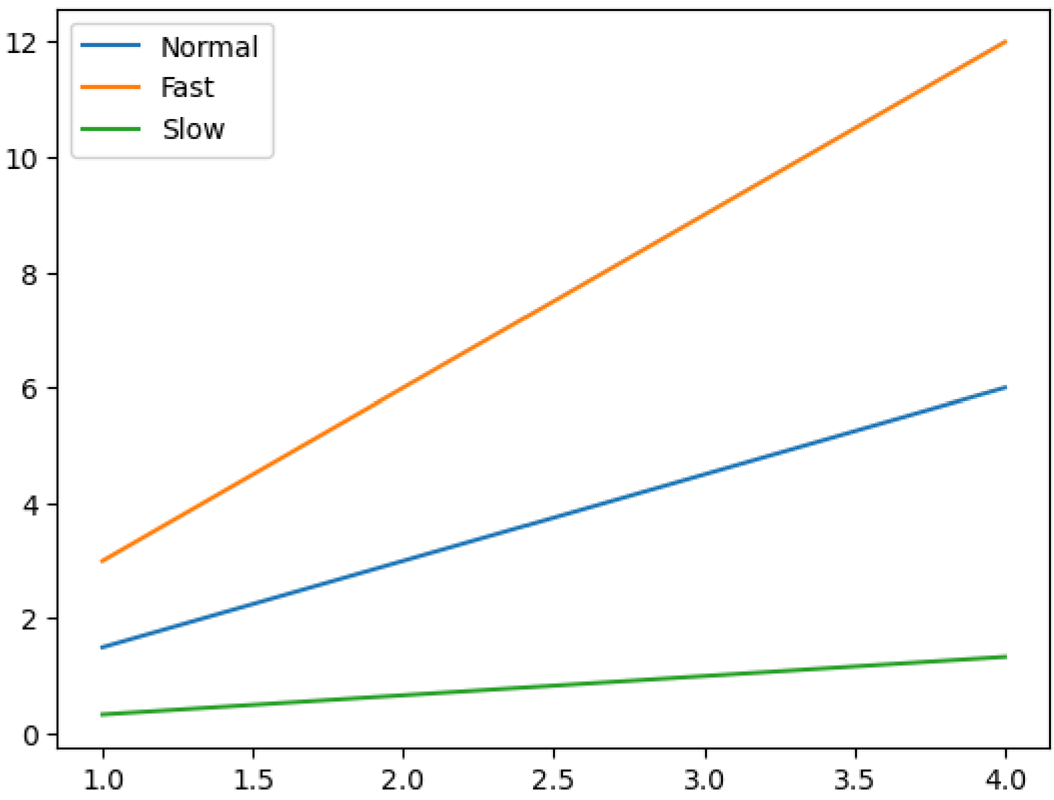
In [112…

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

plt.title('First Plot')

plt.show()
```



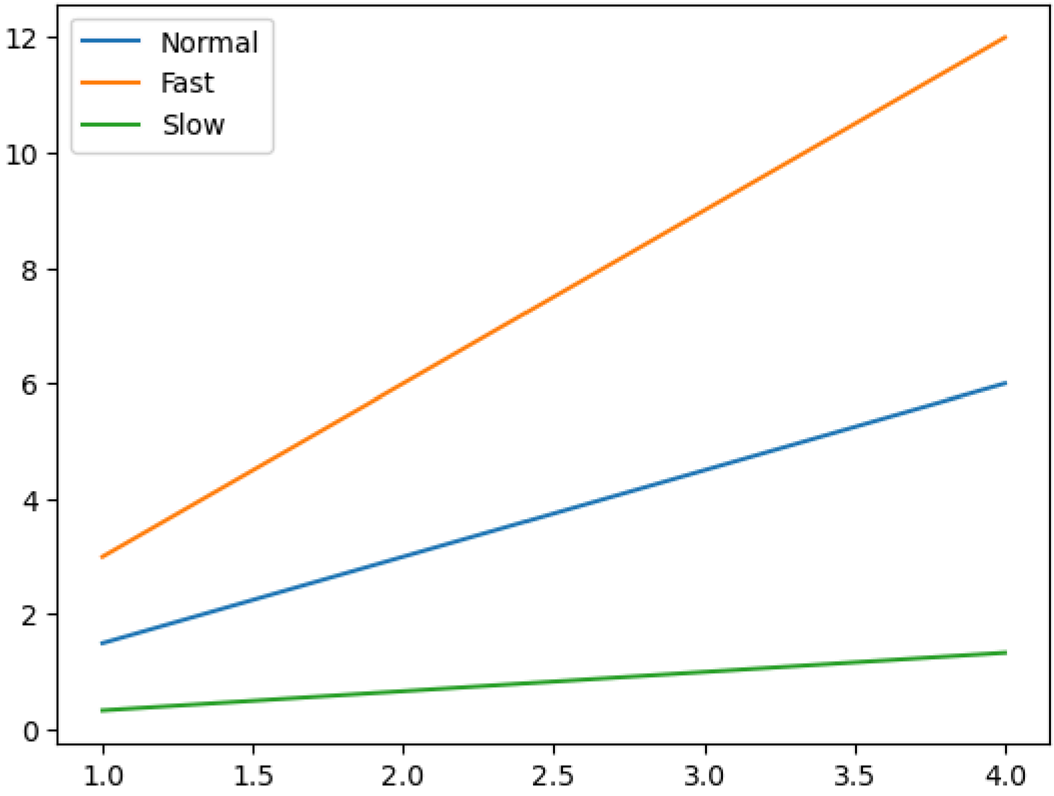
In [115…

```
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()
```



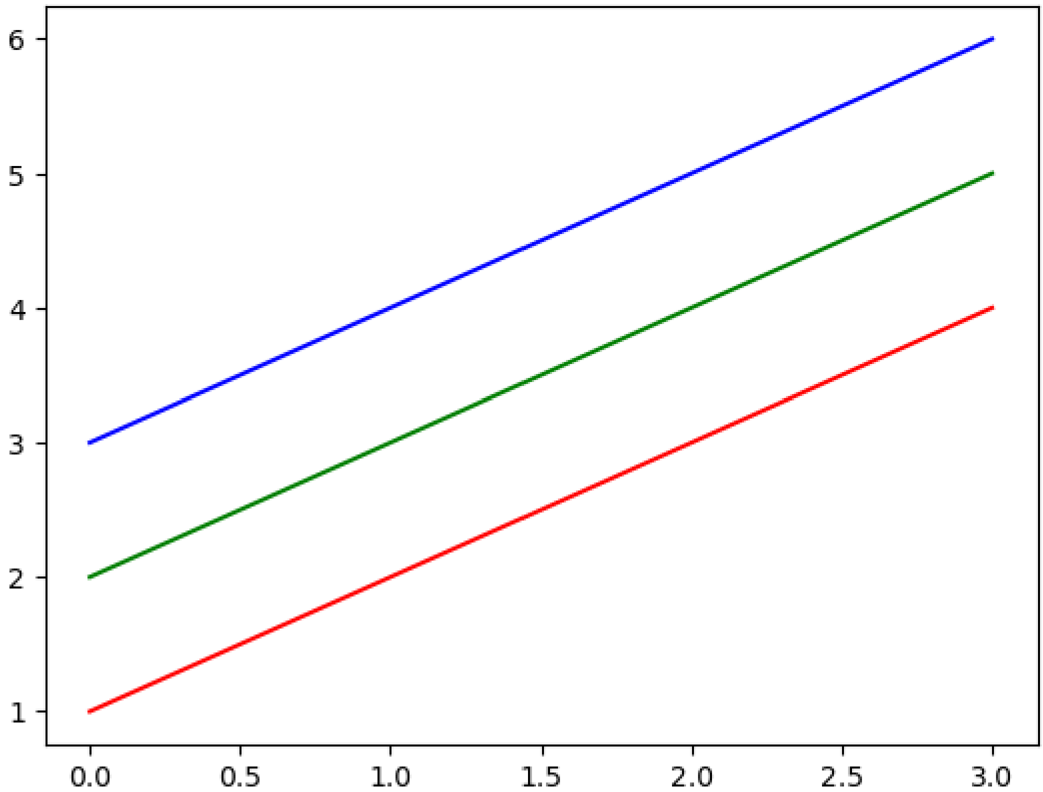
In [118…

```
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()
```



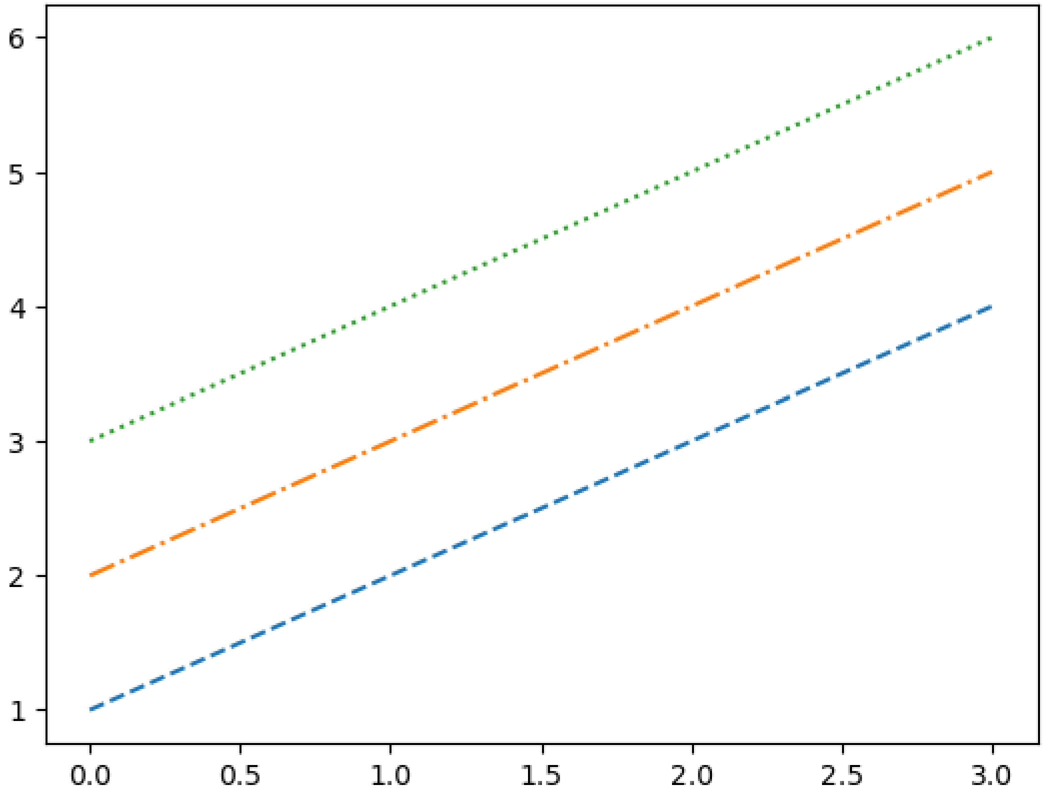
In [119…

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

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

plt.show()
```



In [120…

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

plt.plot(x16, '--', x16+1, '-.', x16+2, ':')

plt.show()
```



In [ ]: