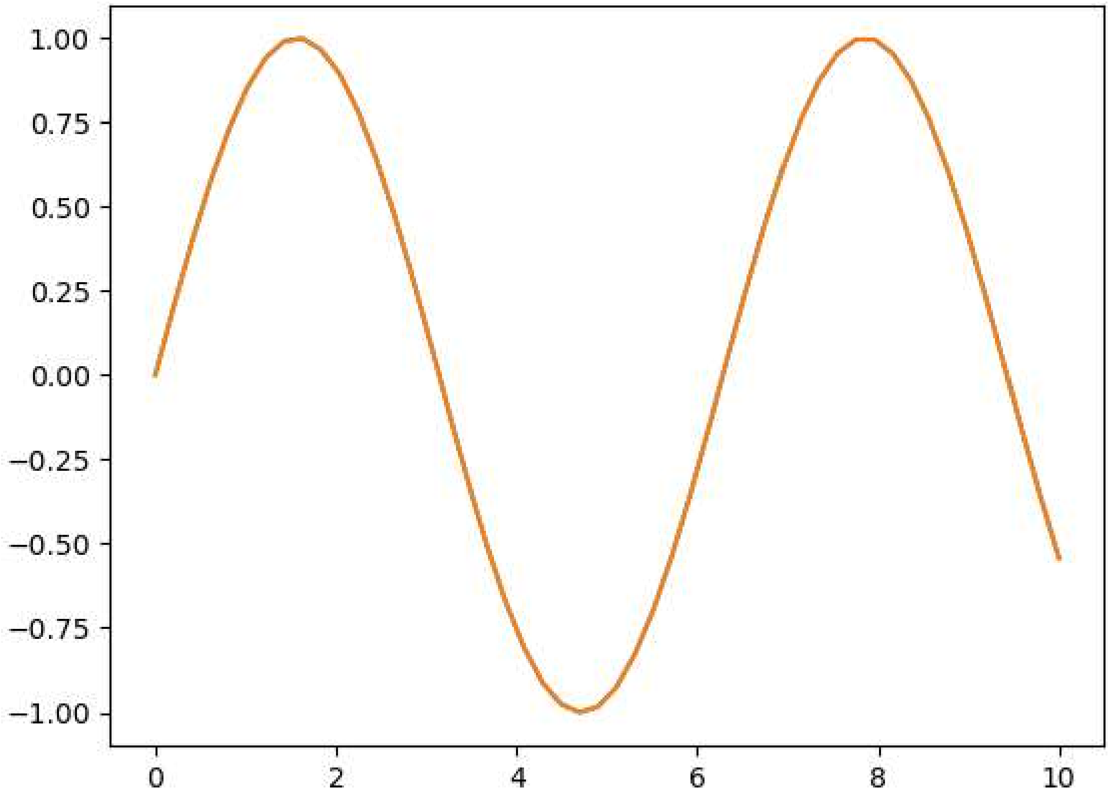
04/11/25

In [8]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
```

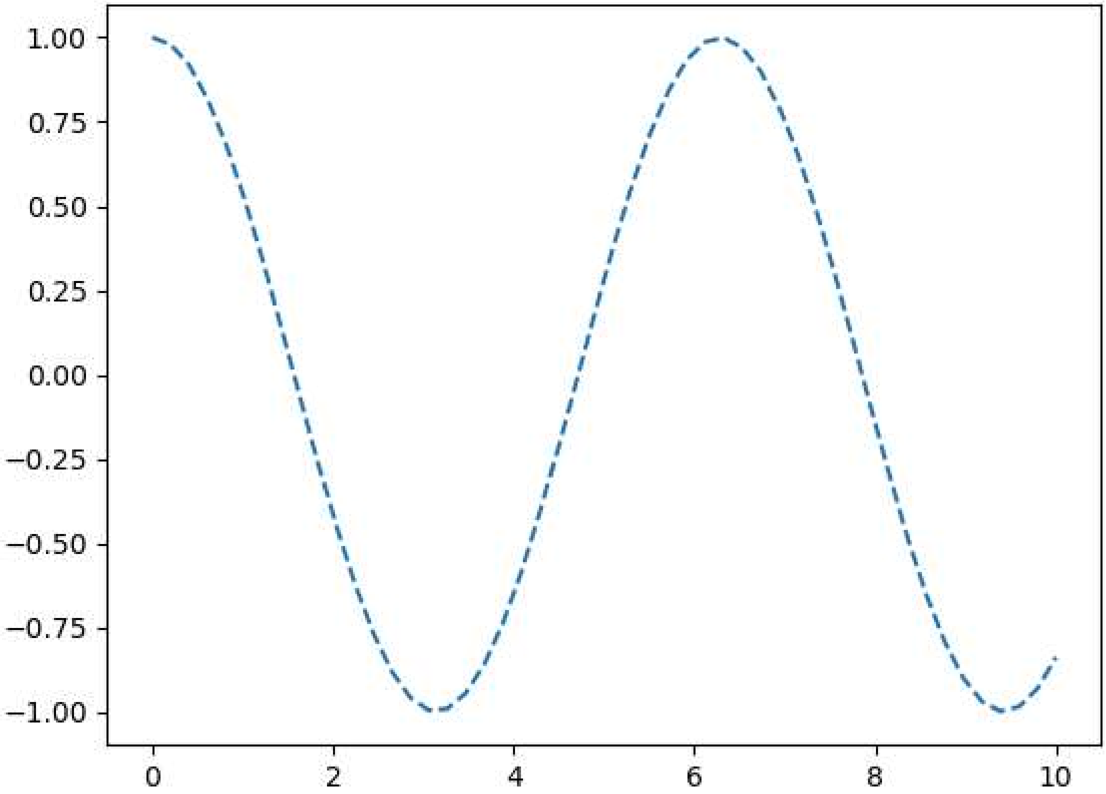
In [12]:
```
x1=np.linspace(0,10,50)
x1
```

Out[12]:
```
array([ 0.        ,  0.20408163,  0.40816327,  0.6122449 ,  0.81632653,
        1.02040816,  1.2244898 ,  1.42857143,  1.63265306,  1.83673469,
        2.04081633,  2.24489796,  2.44897959,  2.65306122,  2.85714286,
        3.06122449,  3.26530612,  3.46938776,  3.67346939,  3.87755102,
        4.08163265,  4.28571429,  4.48979592,  4.69387755,  4.89795918,
        5.10204082,  5.30612245,  5.51020408,  5.71428571,  5.91836735,
        6.12244898,  6.32653061,  6.53061224,  6.73469388,  6.93877551,
        7.14285714,  7.34693878,  7.55102041,  7.75510204,  7.95918367,
        8.16326531,  8.36734694,  8.57142857,  8.7755102 ,  8.97959184,
        9.18367347,  9.3877551 ,  9.59183673,  9.79591837, 10.        ])
```

In [16]:
```
plt.plot(x1,np.sin(x1),'-')
plt.show()
```

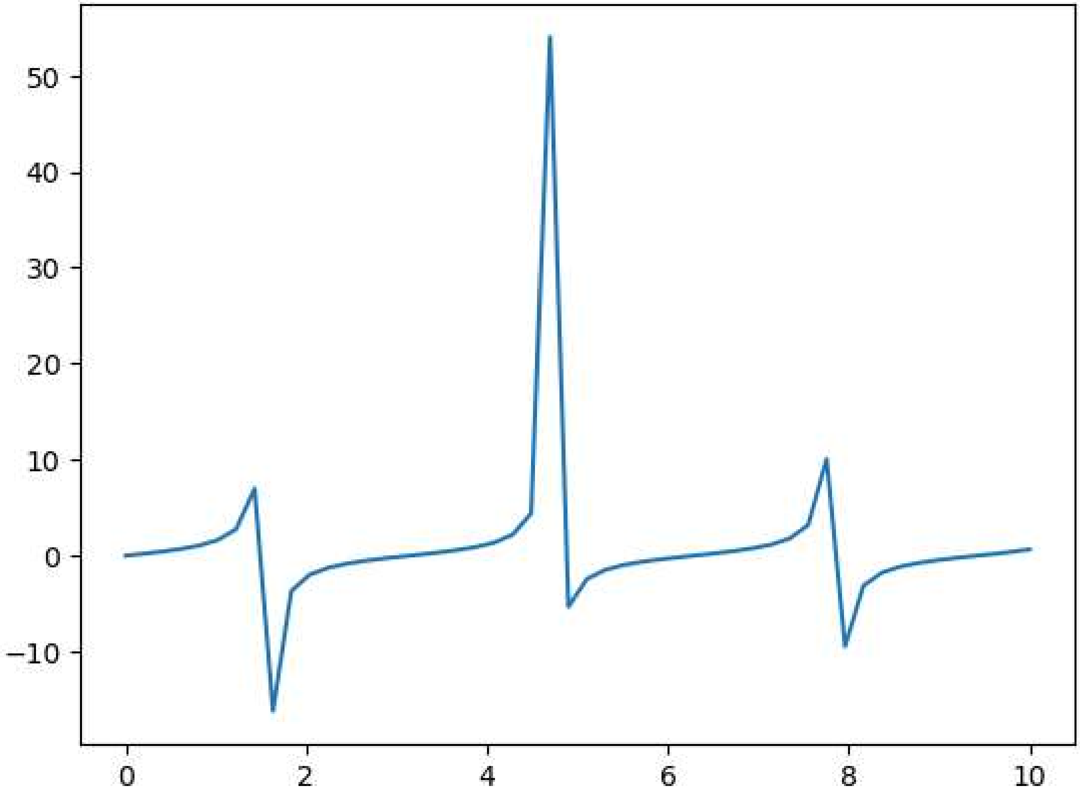


In [20]:
```
plt.plot(x1,np.cos(x1),'-')
plt.show()
```



In [22]:

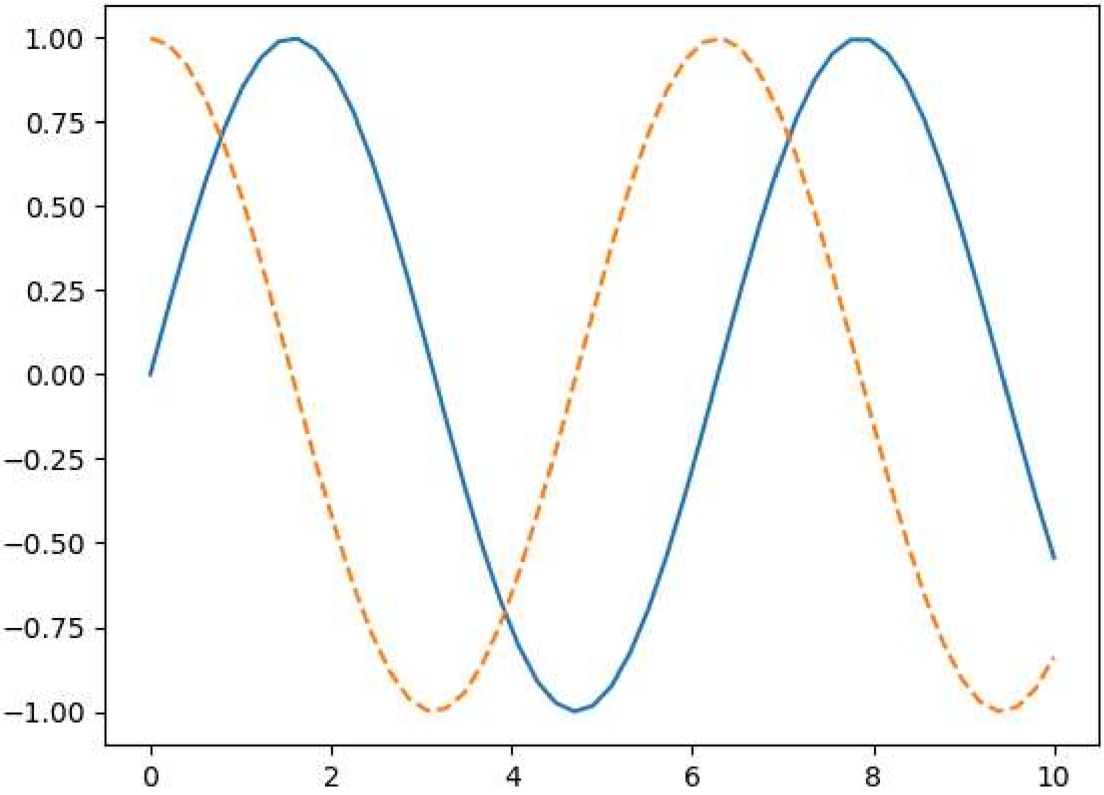
```
plt.plot(x1,np.tan(x1),'-')
plt.show()
```



In [26]:

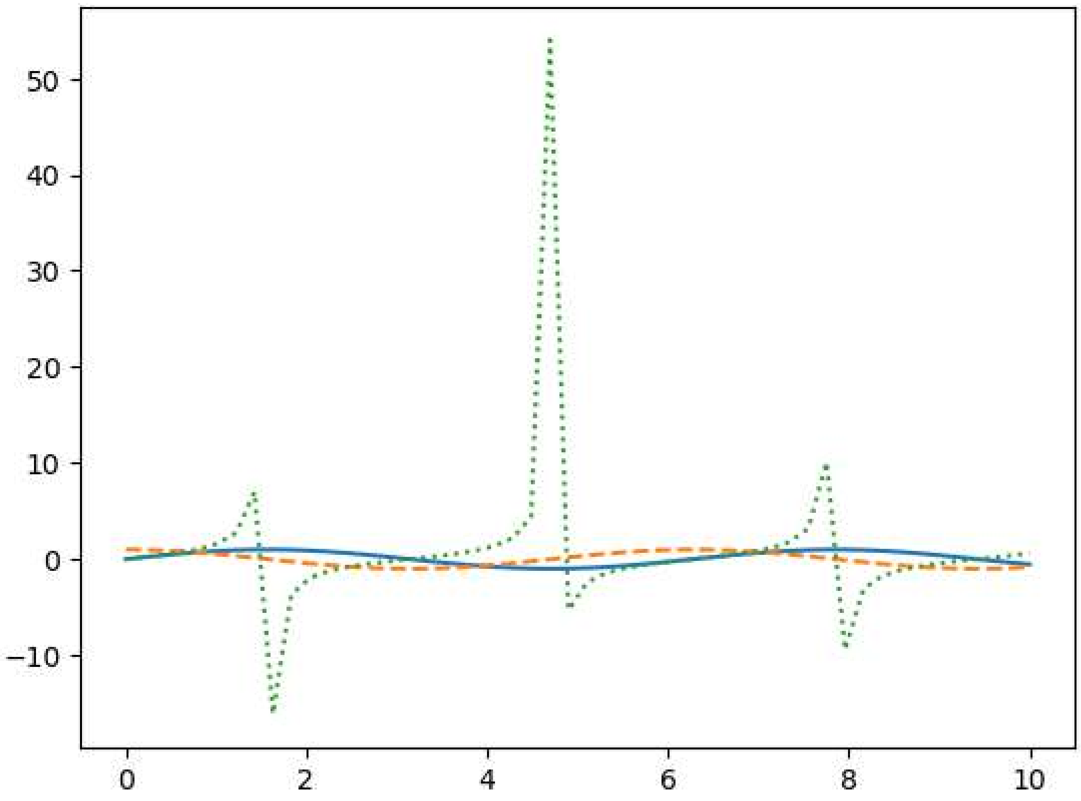
```
plt.plot(x1,np.sin(x1),'-')
plt.plot(x1,np.cos(x1),'--')
```

```
#plt.plot(x1,np.tan(x1),':')
plt.show()
```

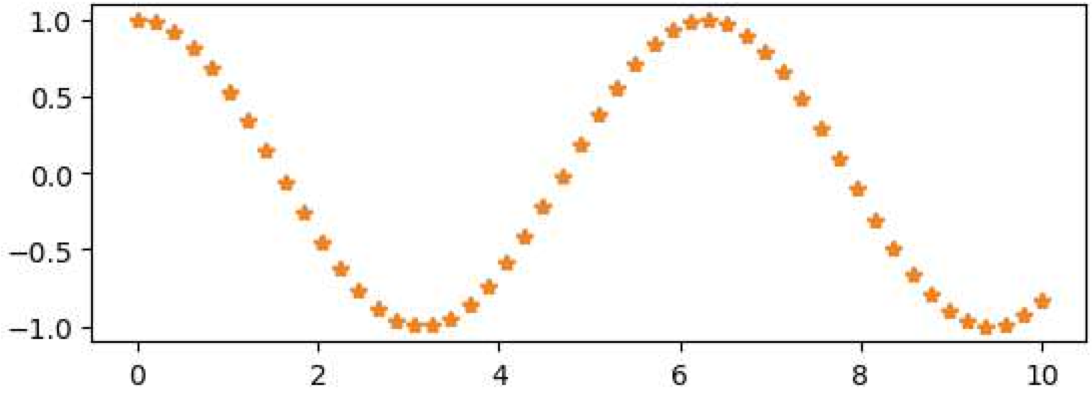


In [28]:

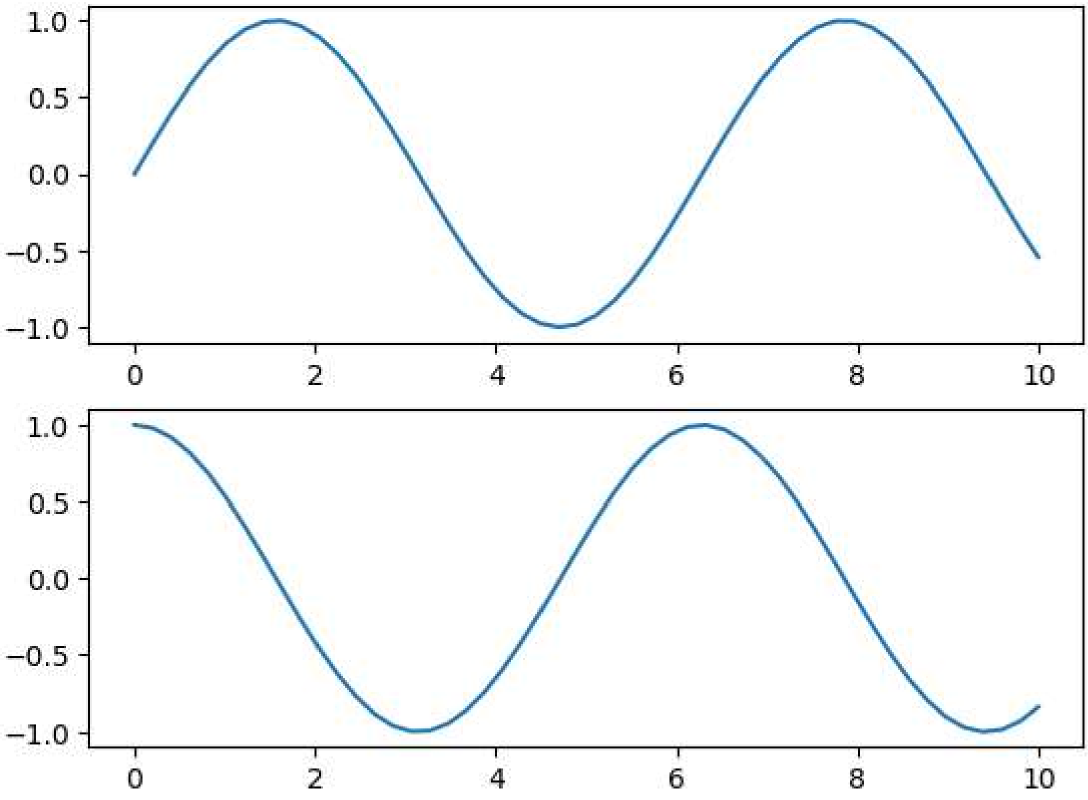
```
plt.plot(x1,np.sin(x1),'-')
plt.plot(x1,np.cos(x1),'--')
plt.plot(x1,np.tan(x1),':')
plt.show()
```



In [32]:
```
plt.subplot(2,1,1) #2-rows,1-column, 1-panels number
plt.plot(x1,np.cos(x1),'*')
plt.show()
```



In [38]:
```
# create a plot figure
plt.figure()
# create the first of two panels and set current axis
plt.subplot(2,1,1)
plt.plot(x1,np.sin(x1))
# create the second of two panels and set current axis
plt.subplot(2,1,2)
plt.plot(x1,np.cos(x1));
plt.show()
```



In [48]:

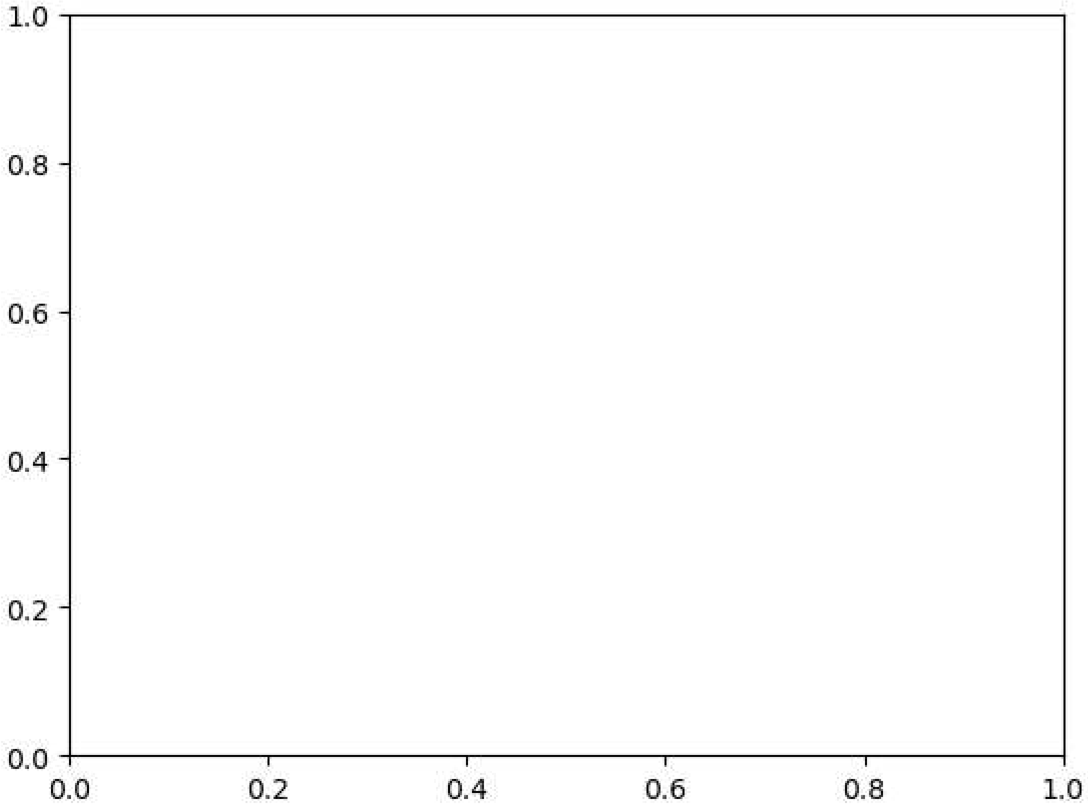
```
# get current figure information
print(plt.gcf())
```

```
Figure(640x480)
```

In [46]:

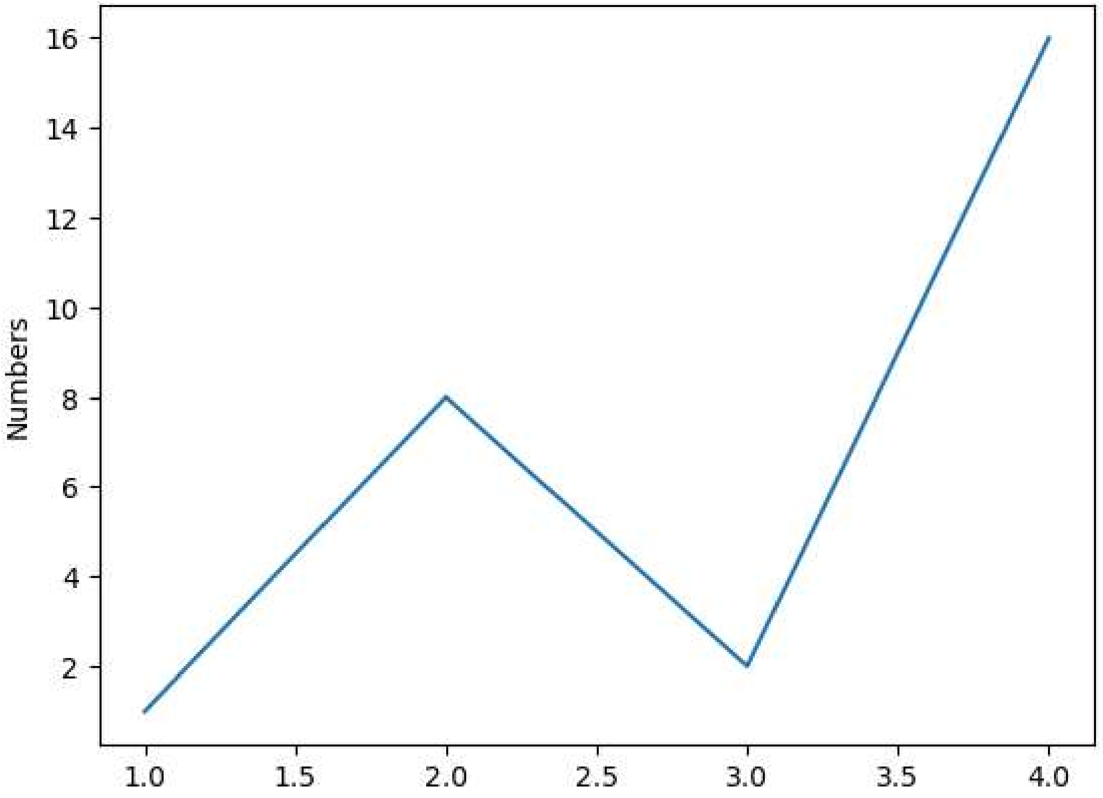
```
# get current axis information
print(plt.gca())
plt.show()
```

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

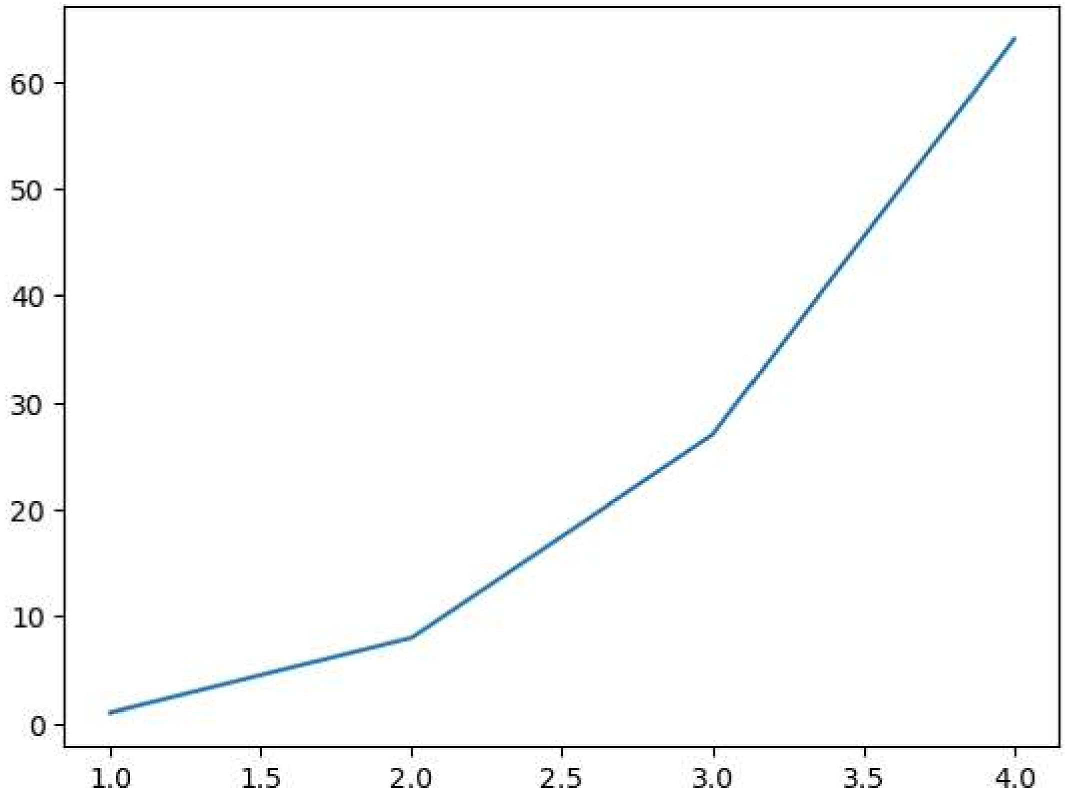


In [50]:

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



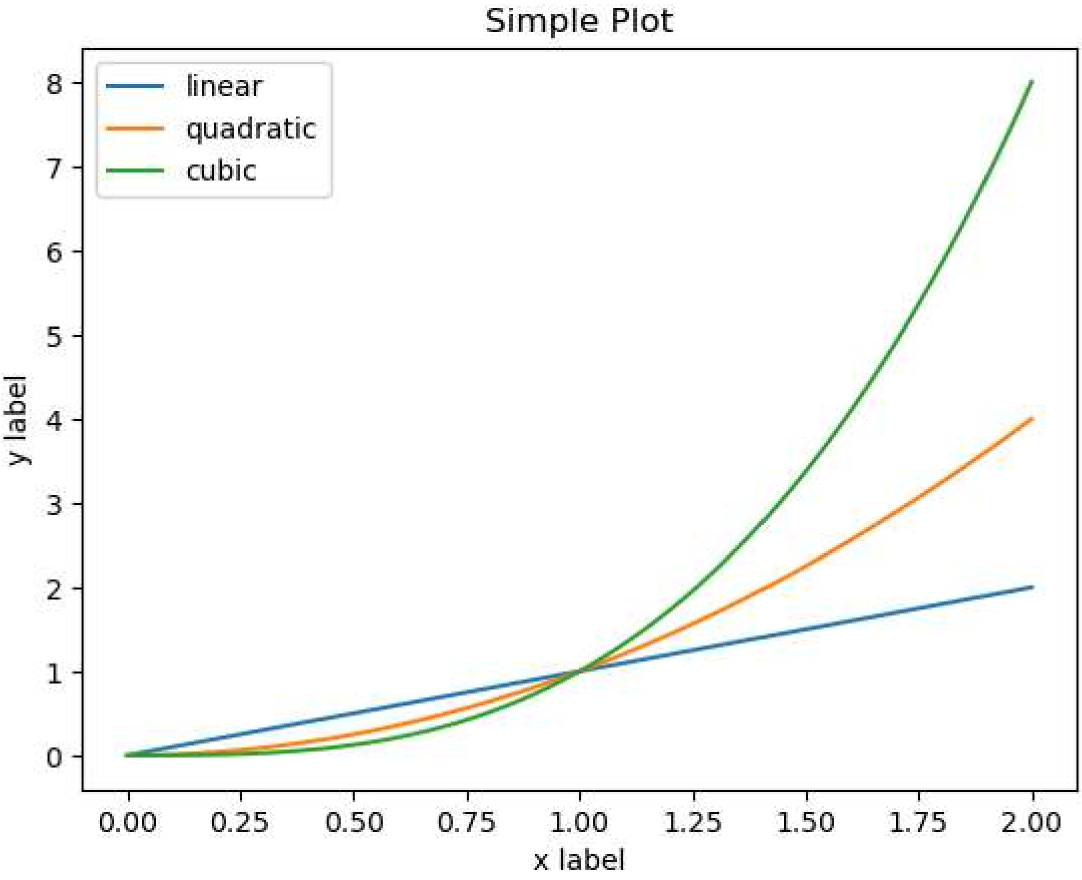
In [52]:
```
plt.plot([1, 2, 3, 4], [1, 8, 27, 64])
plt.show()
```



In [54]:
```
x = np.linspace(0, 2, 100)
x
```

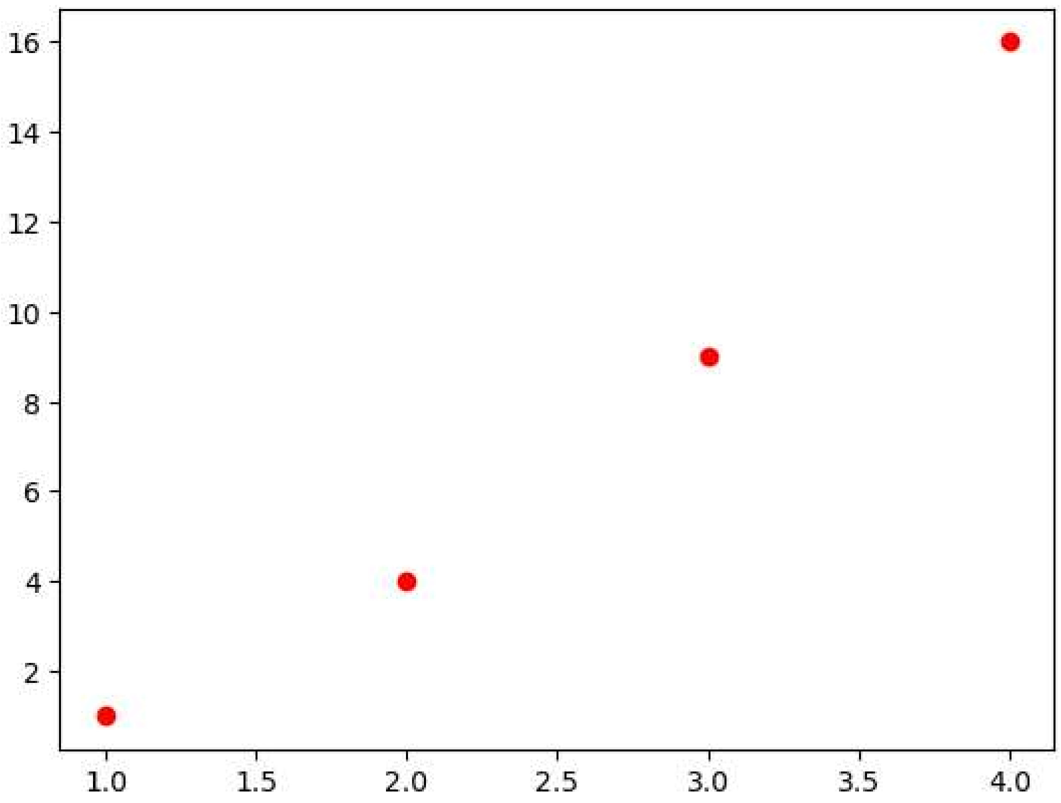
Out[54]:
```
array([0.        , 0.02020202, 0.04040404, 0.06060606, 0.08080808,
       0.1010101 , 0.12121212, 0.14141414, 0.16161616, 0.18181818,
       0.2020202 , 0.22222222, 0.24242424, 0.26262626, 0.28282828,
       0.3030303 , 0.32323232, 0.34343434, 0.36363636, 0.38383838,
       0.4040404 , 0.42424242, 0.44444444, 0.46464646, 0.48484848,
       0.50505051, 0.52525253, 0.54545455, 0.56565657, 0.58585859,
       0.60606061, 0.62626263, 0.64646465, 0.66666667, 0.68686869,
       0.70707071, 0.72727273, 0.74747475, 0.76767677, 0.78787879,
       0.80808081, 0.82828283, 0.84848485, 0.86868687, 0.88888889,
       0.90909091, 0.92929293, 0.94949495, 0.96969697, 0.98989899,
       1.01010101, 1.03030303, 1.05050505, 1.07070707, 1.09090909,
       1.11111111, 1.13131313, 1.15151515, 1.17171717, 1.19191919,
       1.21212121, 1.23232323, 1.25252525, 1.27272727, 1.29292929,
       1.31313131, 1.33333333, 1.35353535, 1.37373737, 1.39393939,
       1.41414141, 1.43434343, 1.45454545, 1.47474747, 1.49494949,
       1.51515152, 1.53535354, 1.55555556, 1.57575758, 1.5959596 ,
       1.61616162, 1.63636364, 1.65656566, 1.67676768, 1.6969697 ,
       1.71717172, 1.73737374, 1.75757576, 1.77777778, 1.7979798 ,
       1.81818182, 1.83838384, 1.85858586, 1.87878788, 1.8989899 ,
       1.91919192, 1.93939394, 1.95959596, 1.97979798, 2.        ])
```

In [56]:
```
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()
```

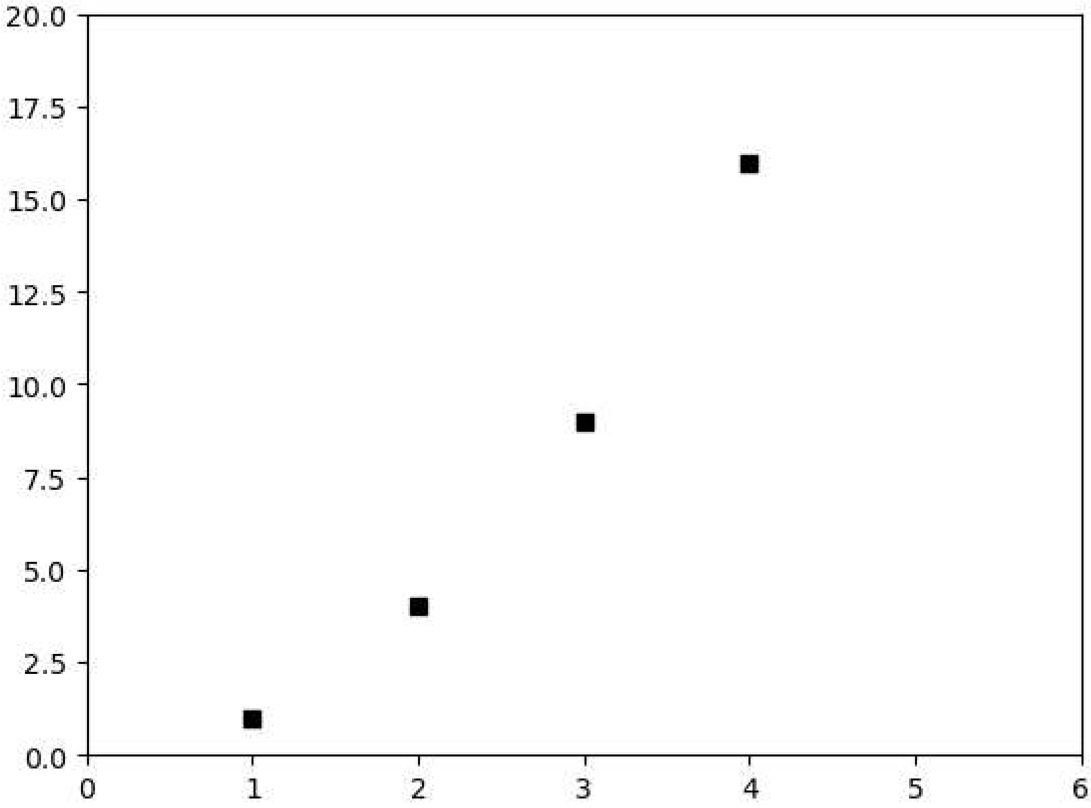


In [68]:

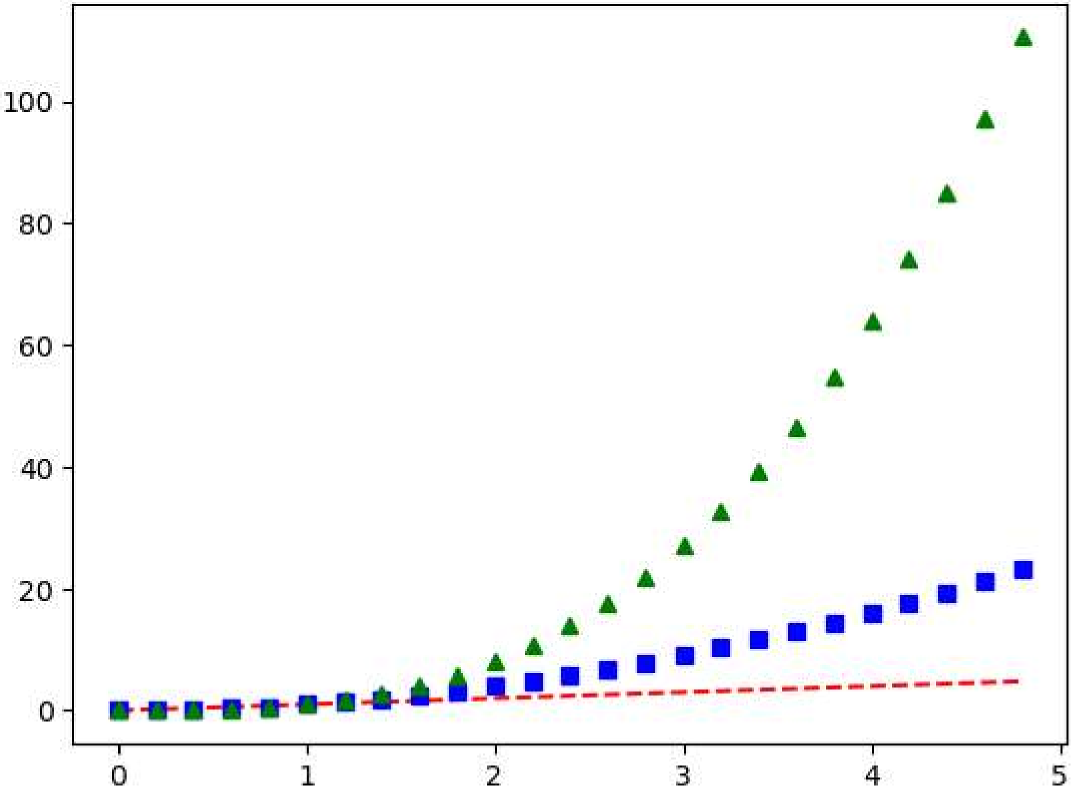
```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
plt.show()
```



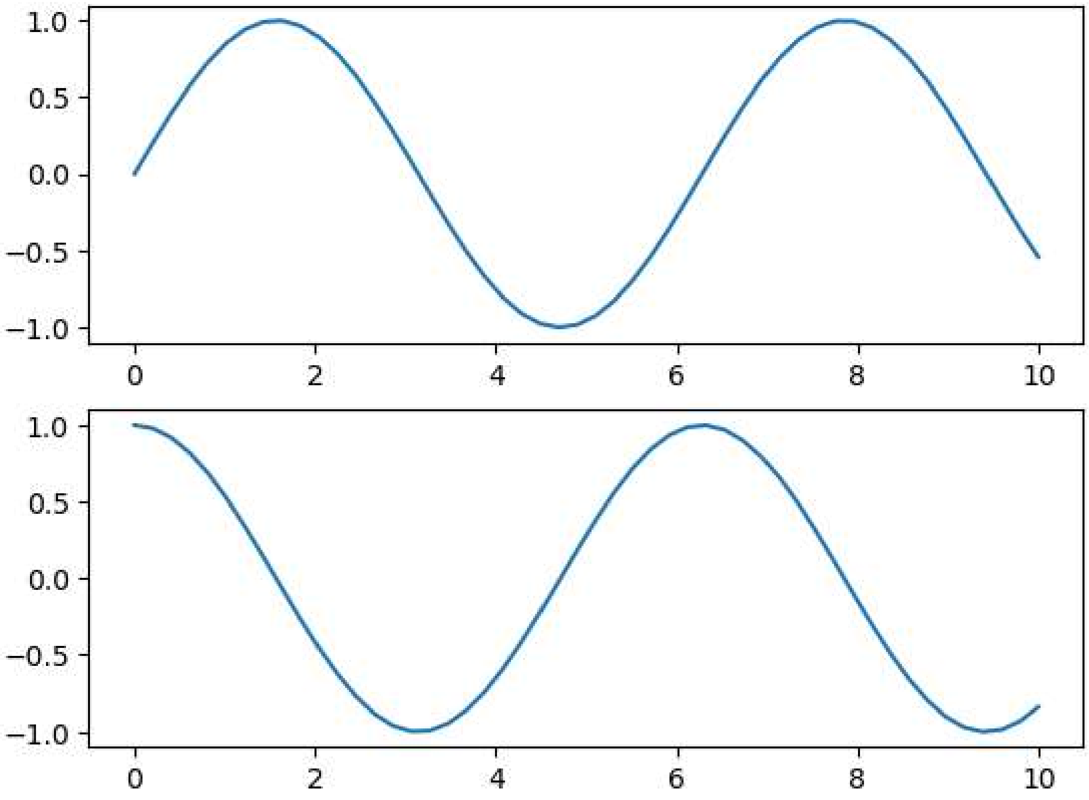
In [80]:
```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 's',c='k')
plt.axis([0, 6, 0, 20])  # [xmin, xmax, ymin, ymax]
plt.show()
```



In [84]:
```
# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)
t
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```

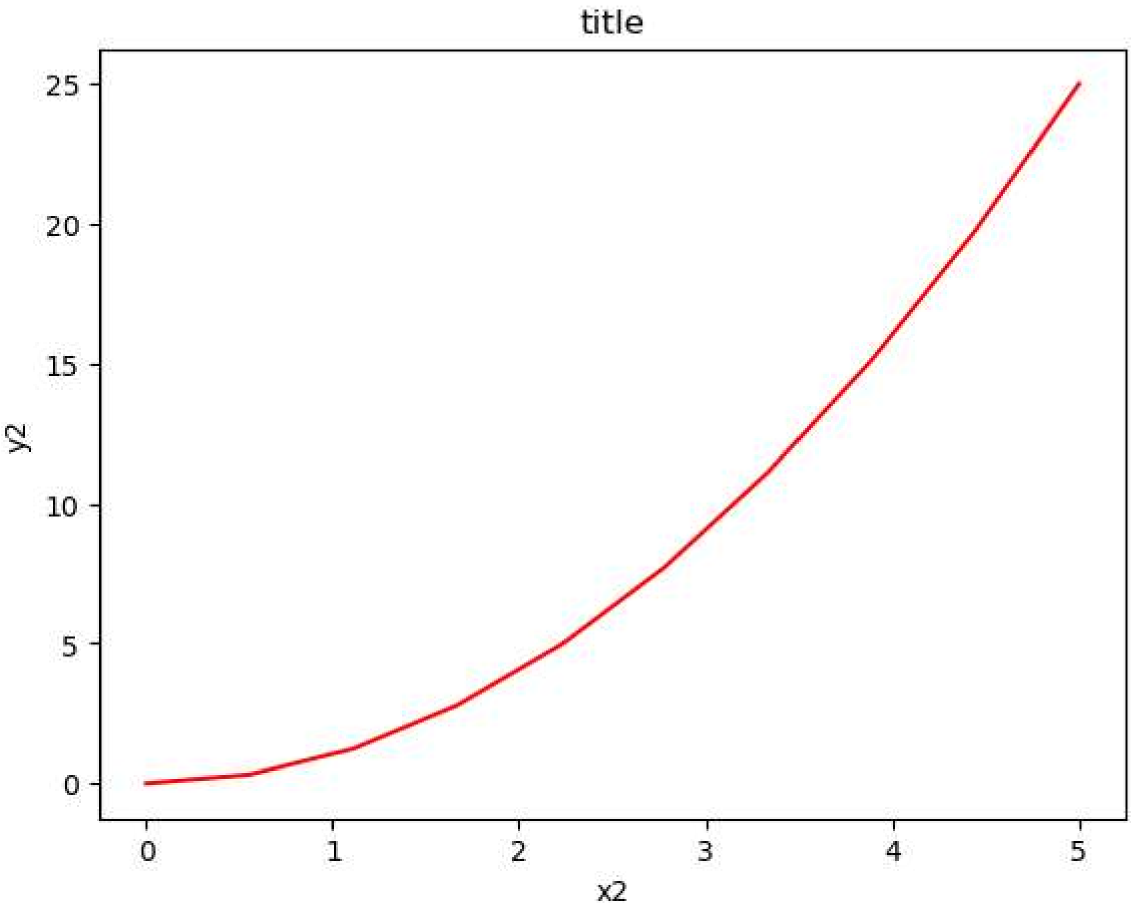


In [92]:
```
# First create a grid of plots
# ax will be an array of two Axes objects
fig, ax = plt.subplots(2)
ax[0].plot(x1, np.sin(x1), '-')
ax[1].plot(x1, np.cos(x1), '-');
plt.show()
```



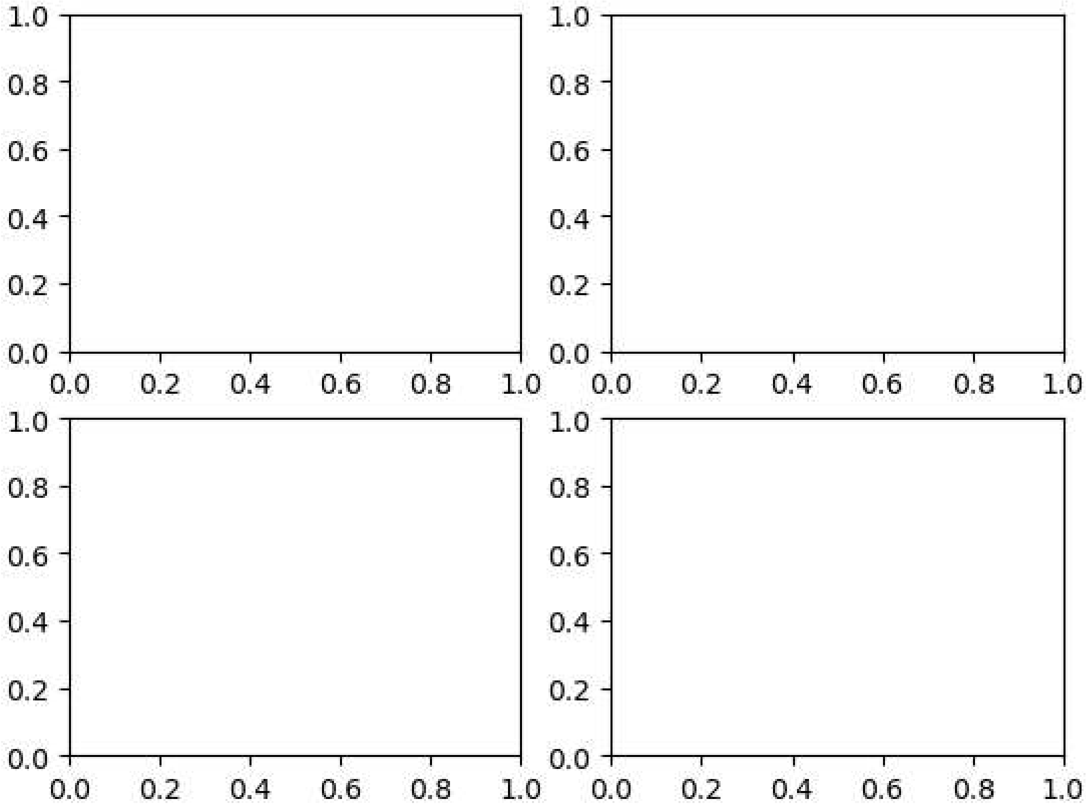
In [102…

```
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()
```



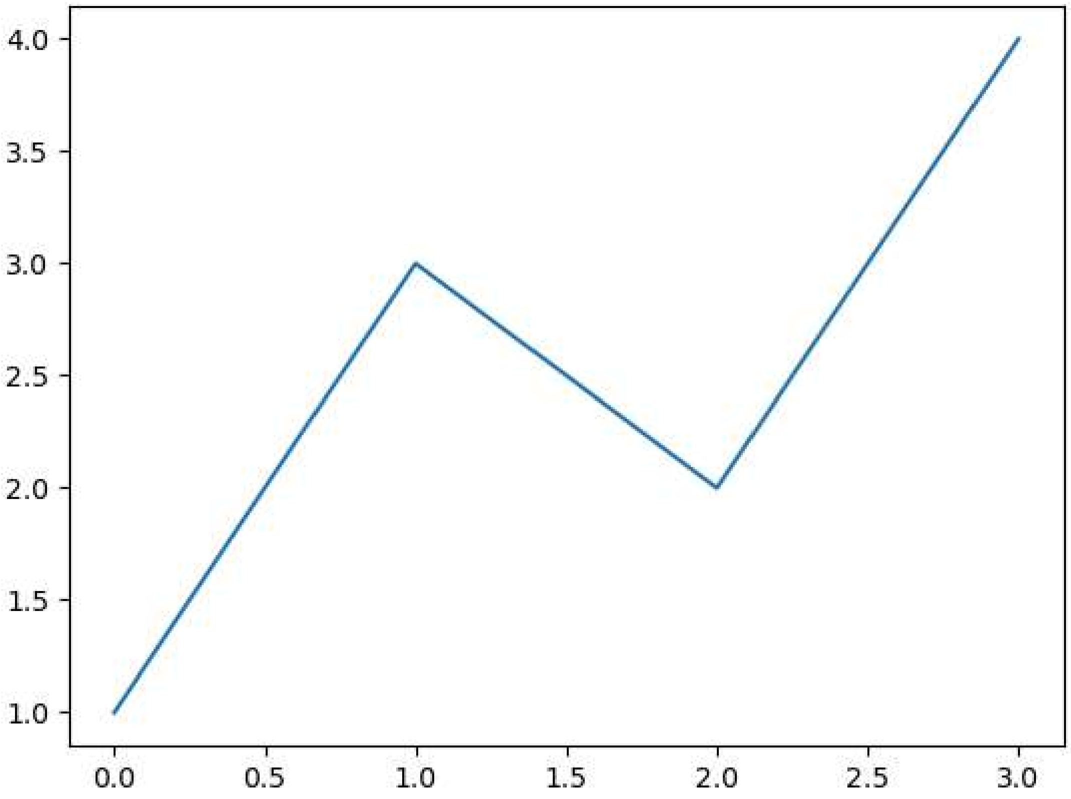
In [104…

```
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.show()
```



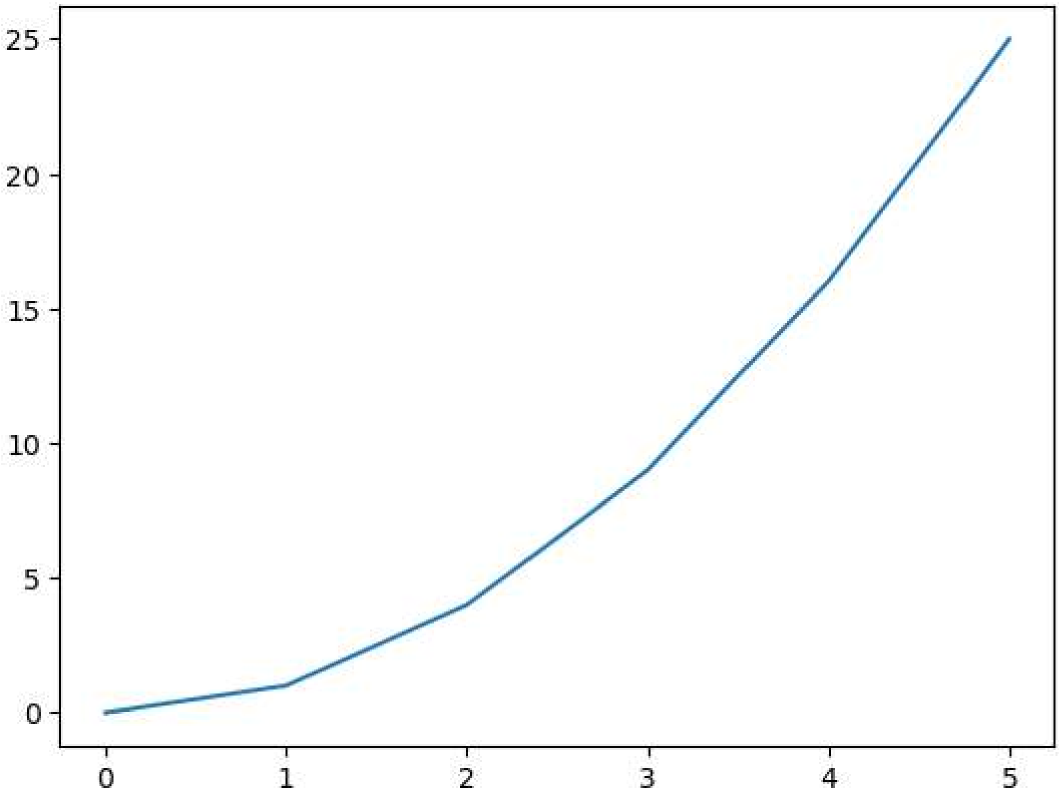
In [106…

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



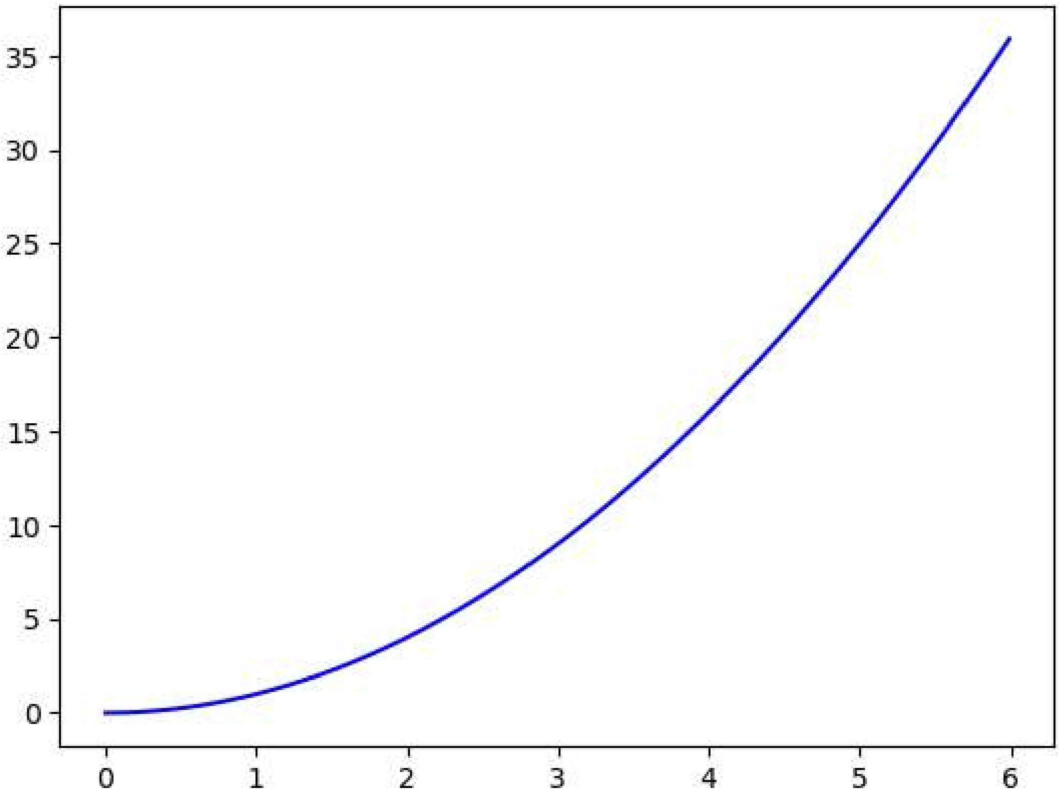
In [108…

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



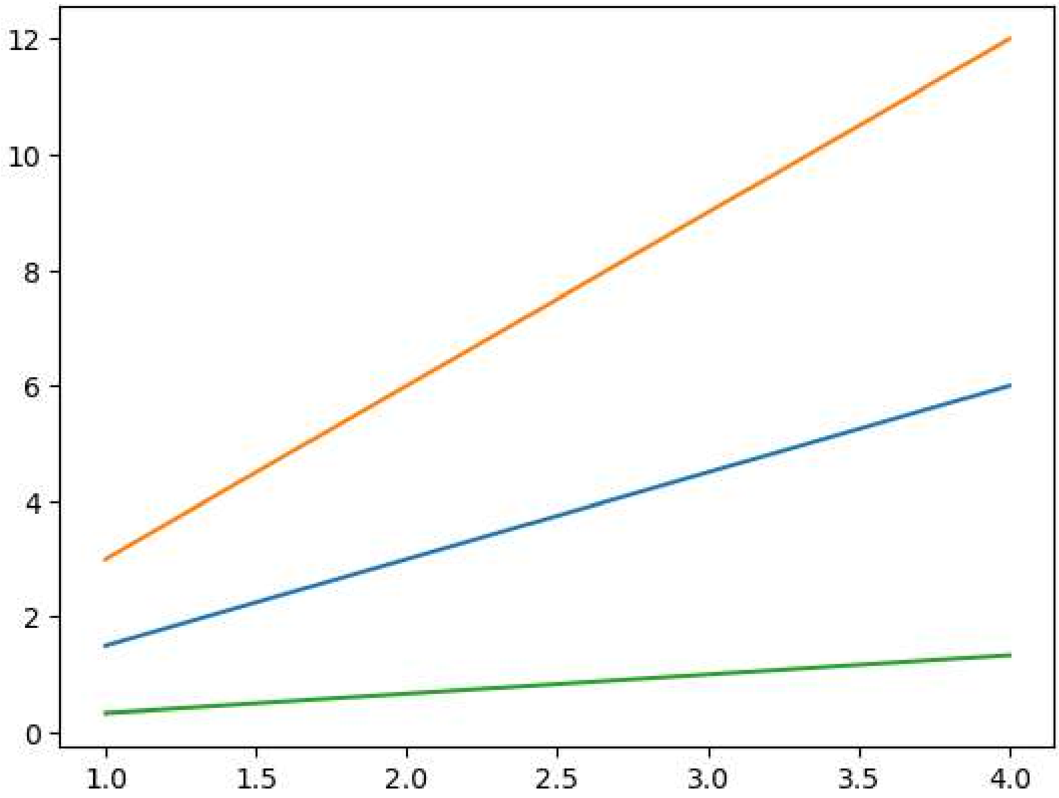
In [110…

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



In [112…

```
# multiline plot
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()
```



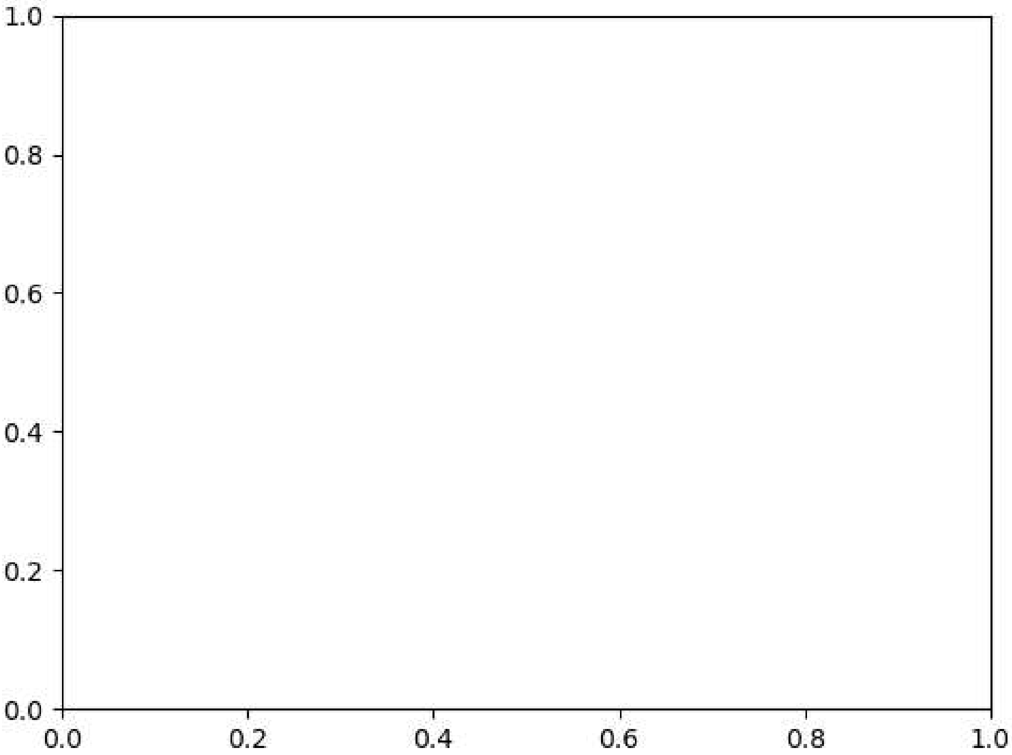
In [138…

```
fig = plt.figure()
x1 = fig.add_subplot(1, 1, 1)
```

In [140…

```
# Saving the figure
fig.savefig('plot1.png')
# Explore the contents of figure
from IPython.display import Image
Image('plot1.png')
```

Out[140…



In [142…

```
# Explore supported file formats
fig.canvas.get_supported_filetypes()
```

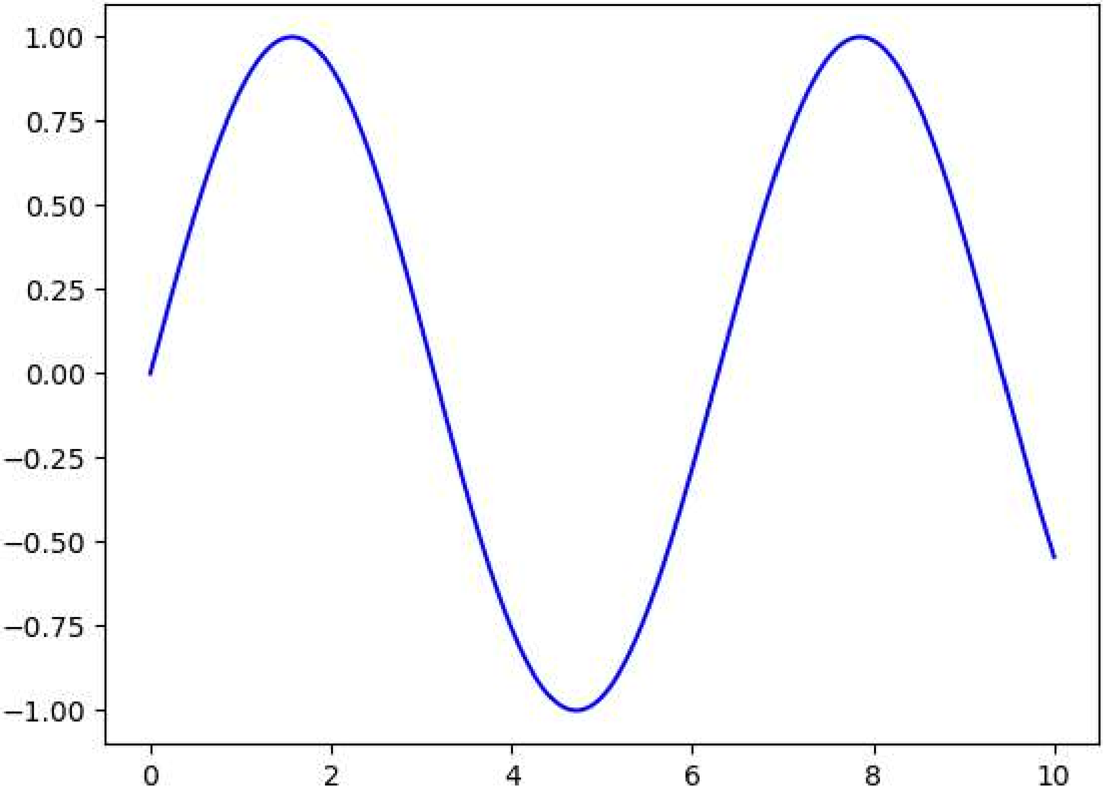
Out[142…

```
{'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'}
```

LinePlot

In [149…

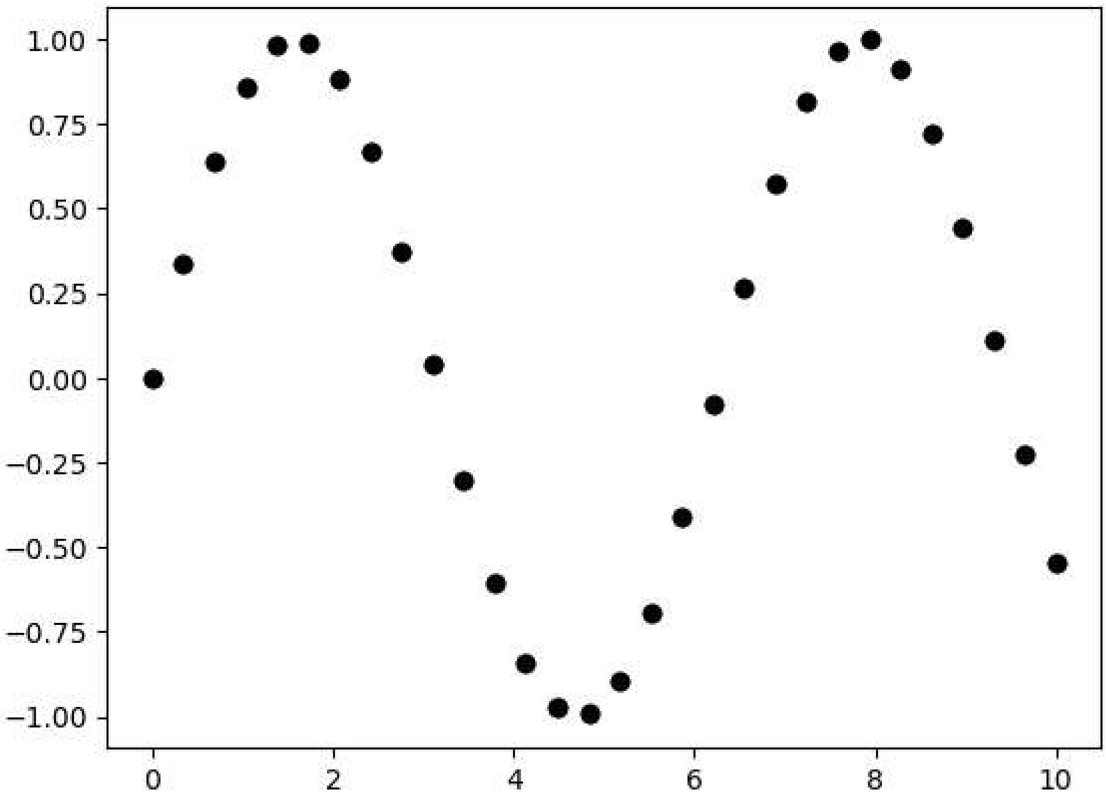
```
# Create figure and axes first
fig = plt.figure()
ax = plt.axes()
# Declare a variable x5
x5 = np.linspace(0, 10, 1000)
# print(x5)
# Plot the sinusoid function
ax.plot(x5, np.sin(x5), 'b-');
plt.show()
```



Scatter Plot

In [154…

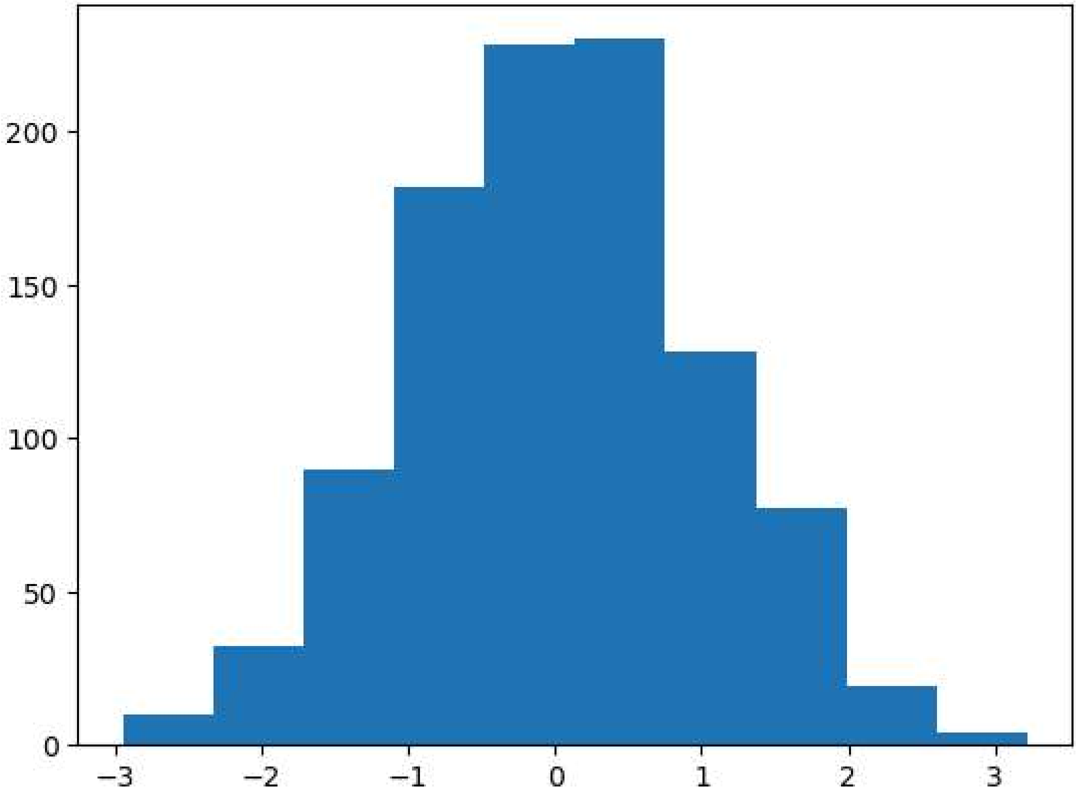
```
x7 = np.linspace(0, 10, 30)
y7 = np.sin(x7)
plt.plot(x7, y7, 'o', color = 'black');
plt.show()
```



Histogram

In [163…

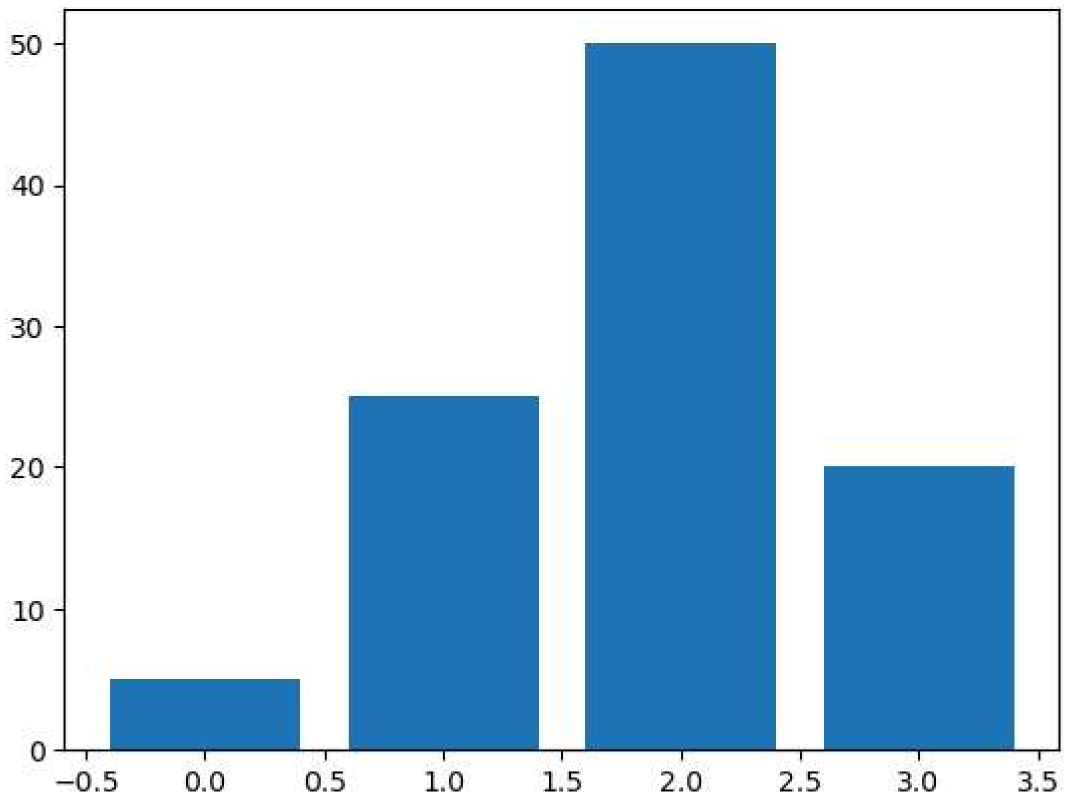
```
data1 = np.random.randn(1000)
plt.hist(data1);
plt.show()
```



Bar Chart

In [173…

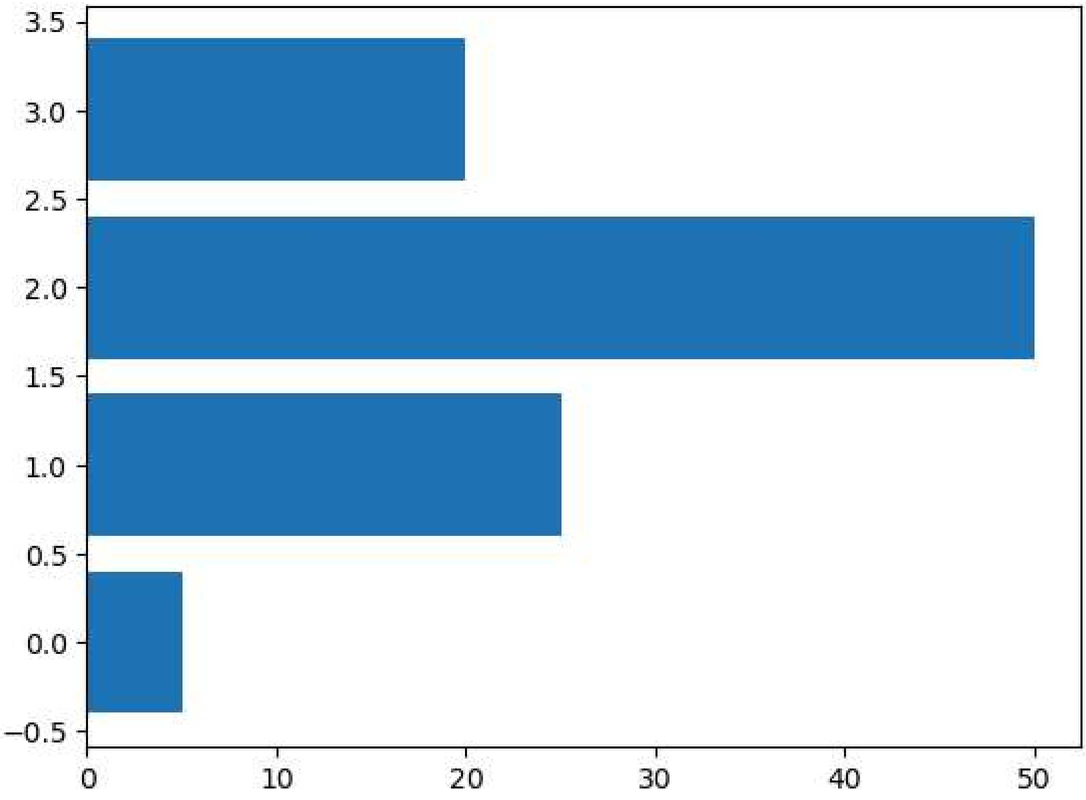
```
data2 = [5. , 25. , 50. , 20.]
plt.bar(range(len(data2)), data2)
plt.show()
```



Horizontal Bar Chart

In [176…

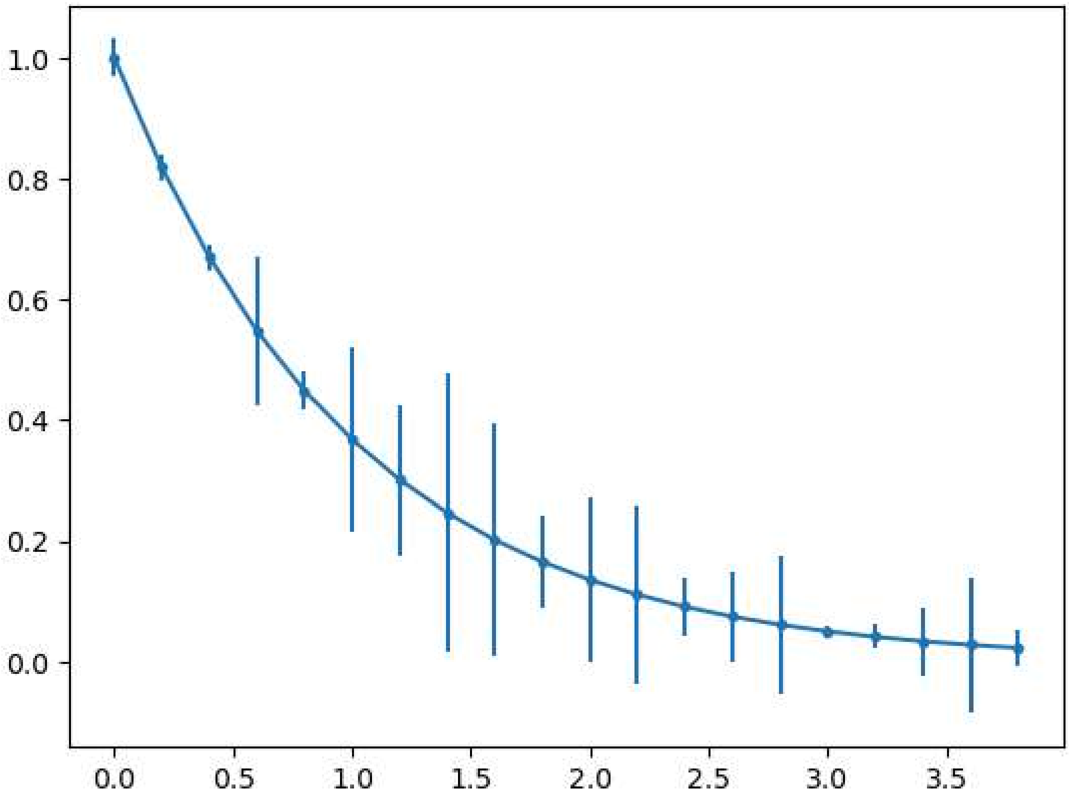
```
data2 = [5. , 25. , 50. , 20.]
plt.barh(range(len(data2)), data2)
plt.show()
```



Error Bar Chart

In [179…

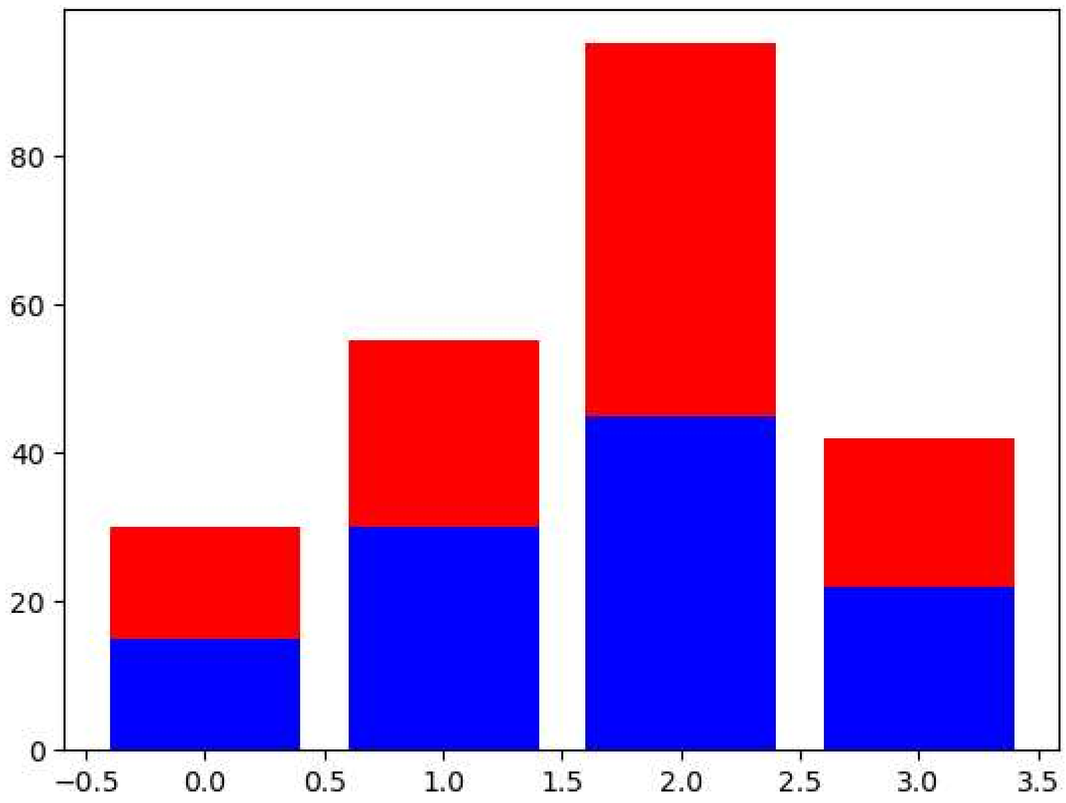
```
x9 = np.arange(0, 4, 0.2)
y9 = np.exp(-x9)
e1 = 0.1 * np.abs(np.random.randn(len(y9)))
plt.errorbar(x9, y9, yerr = e1, fmt = '.-')
plt.show();
```



Stacked Bar Chart

In [182…

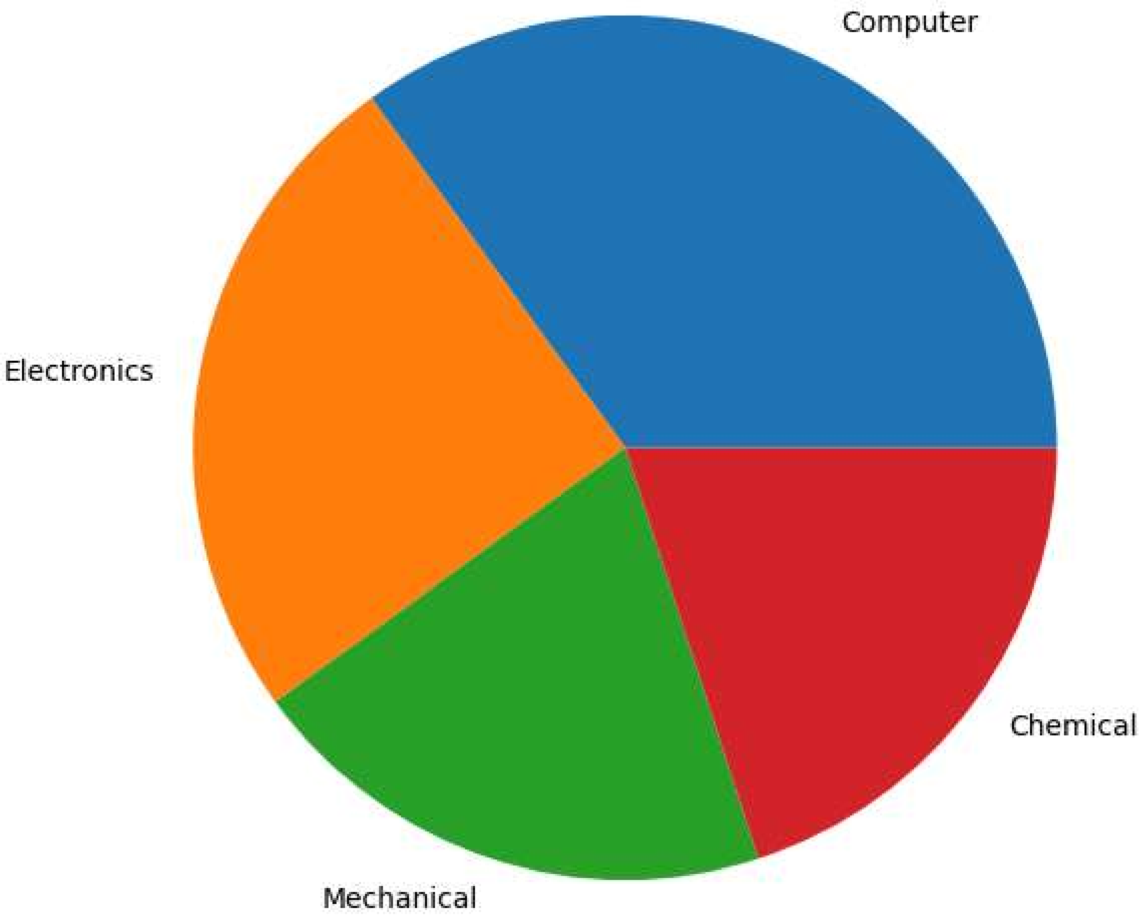
```
A = [15., 30., 45., 22.]
B = [15., 25., 50., 20.]
z2 = range(4)
plt.bar(z2, A, color = 'b')
plt.bar(z2, B, color = 'r', bottom = A)
plt.show()
```



Pie Chart

In [185…

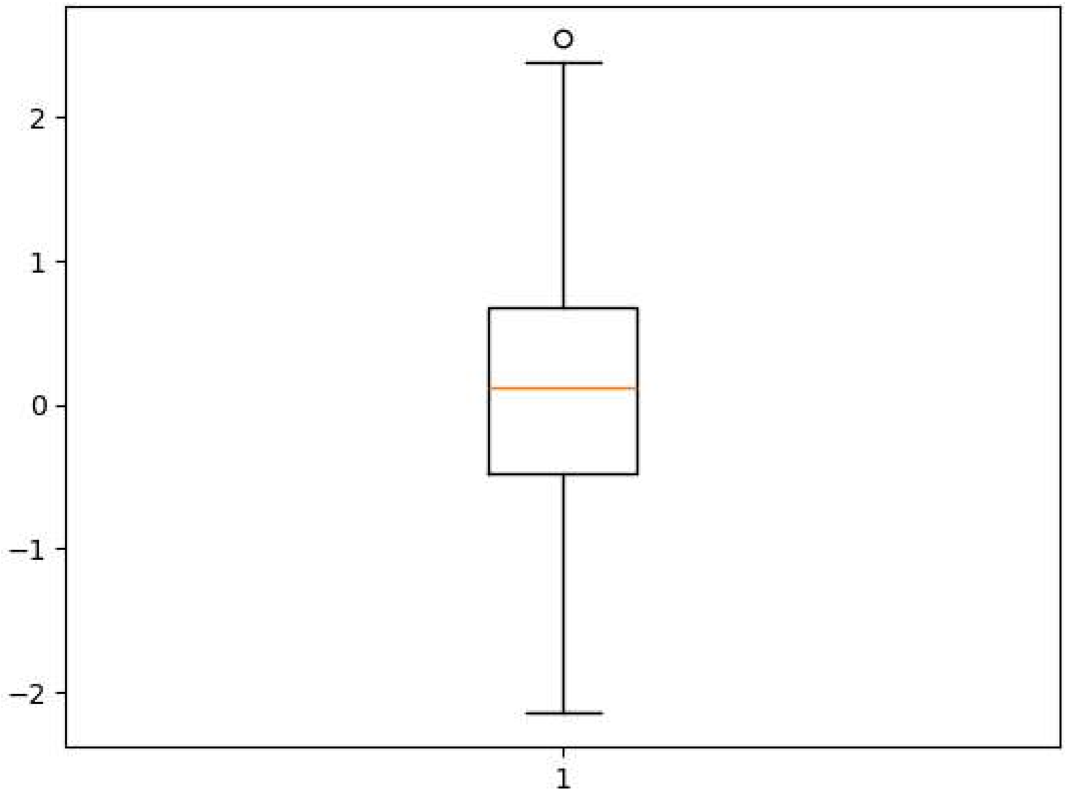
```
plt.figure(figsize=(7,7))
x10 = [35, 25, 20, 20]
labels = ['Computer', 'Electronics', 'Mechanical', 'Chemical']
plt.pie(x10, labels=labels);
plt.show()
```



Box Plot

In [188…

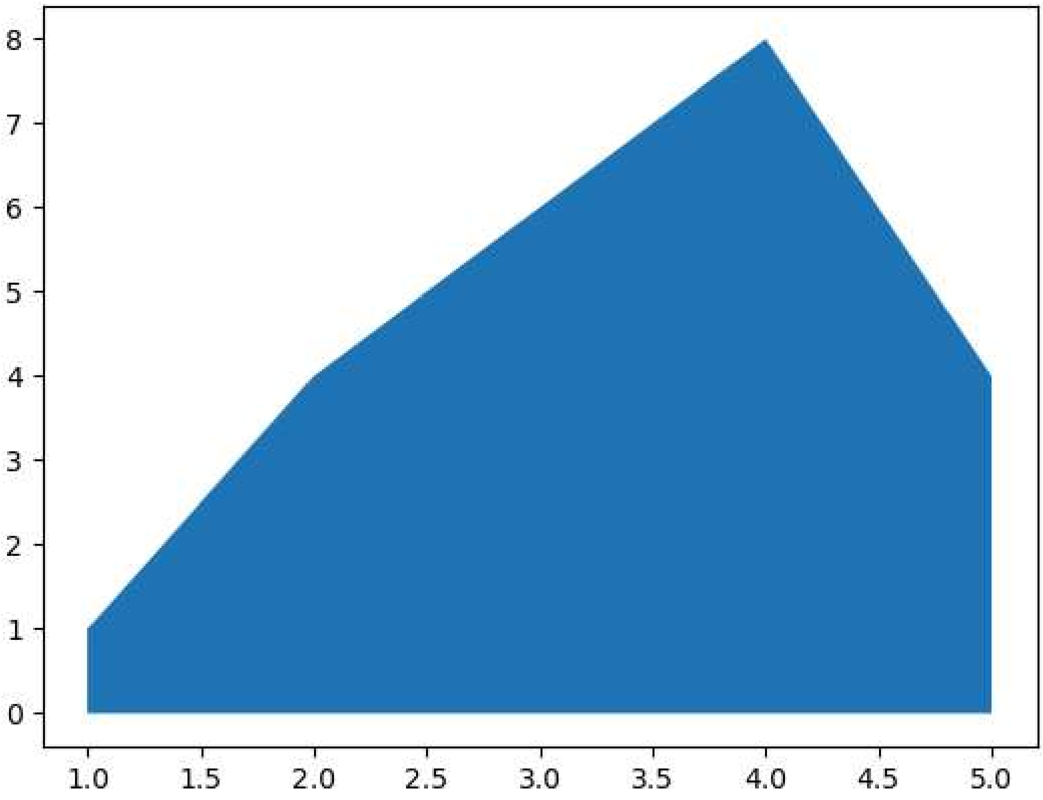
```
data3 = np.random.randn(100)
plt.boxplot(data3)
plt.show();
```



Area Chart

In [191…

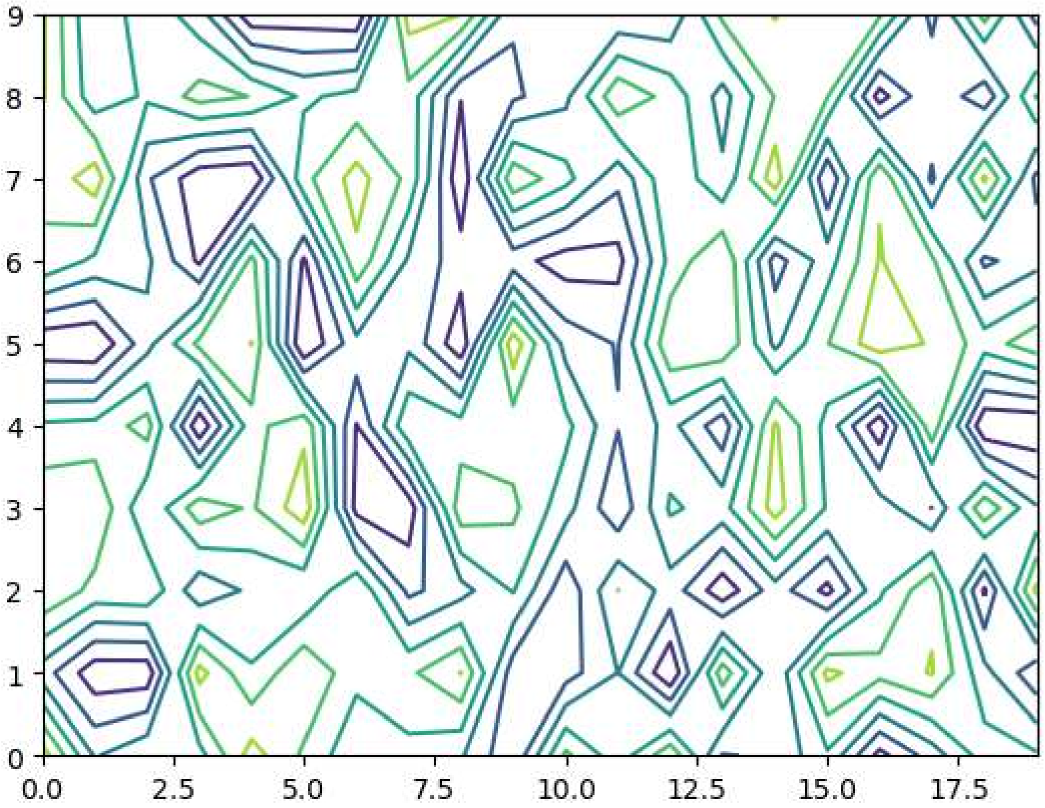
```
# Create some data
x12 = range(1, 6)
y12 = [1, 4, 6, 8, 4]
# Area plot
plt.fill_between(x12, y12)
plt.show()
```



Contour Plot

In [194…

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



Styles with Matplotlib Plots

In [263…

```
# View list of all available styles
print(plt.style.available)
```

```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'b
mh', 'classic', 'dark_background', 'fast', 'fivethirtyeight', 'ggplot', 'grayscale',
'seaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblind', 'seaborn-v0_8-dar
k', 'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-deep', 'seab
orn-v0_8-muted', 'seaborn-v0_8-notebook', 'seaborn-v0_8-paper', 'seaborn-v0_8-paste
l', 'seaborn-v0_8-poster', 'seaborn-v0_8-talk', 'seaborn-v0_8-ticks', 'seaborn-v0_8-
white', 'seaborn-v0_8-whitegrid', 'tableau-colorblind10']
```

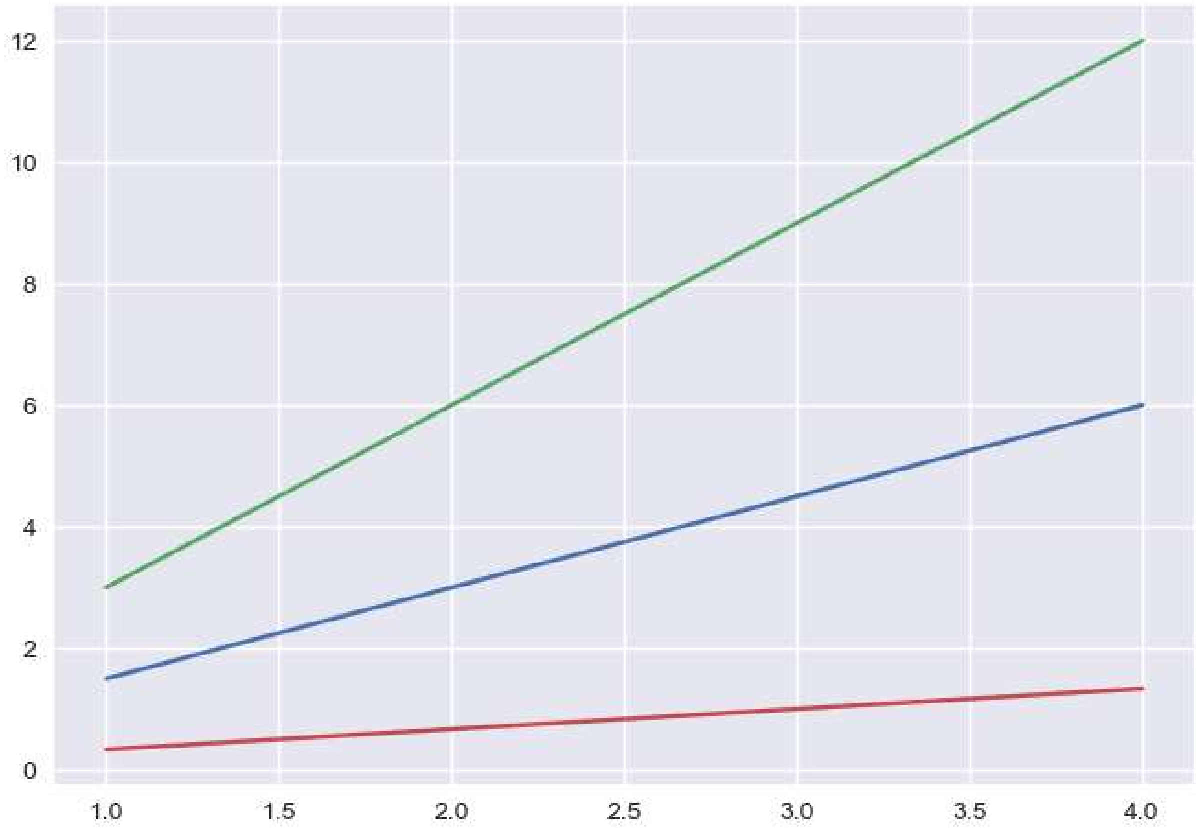
In [265…

```
# Set styles for plots
plt.style.use('seaborn-v0_8')
```

Adding a grid

In [272…

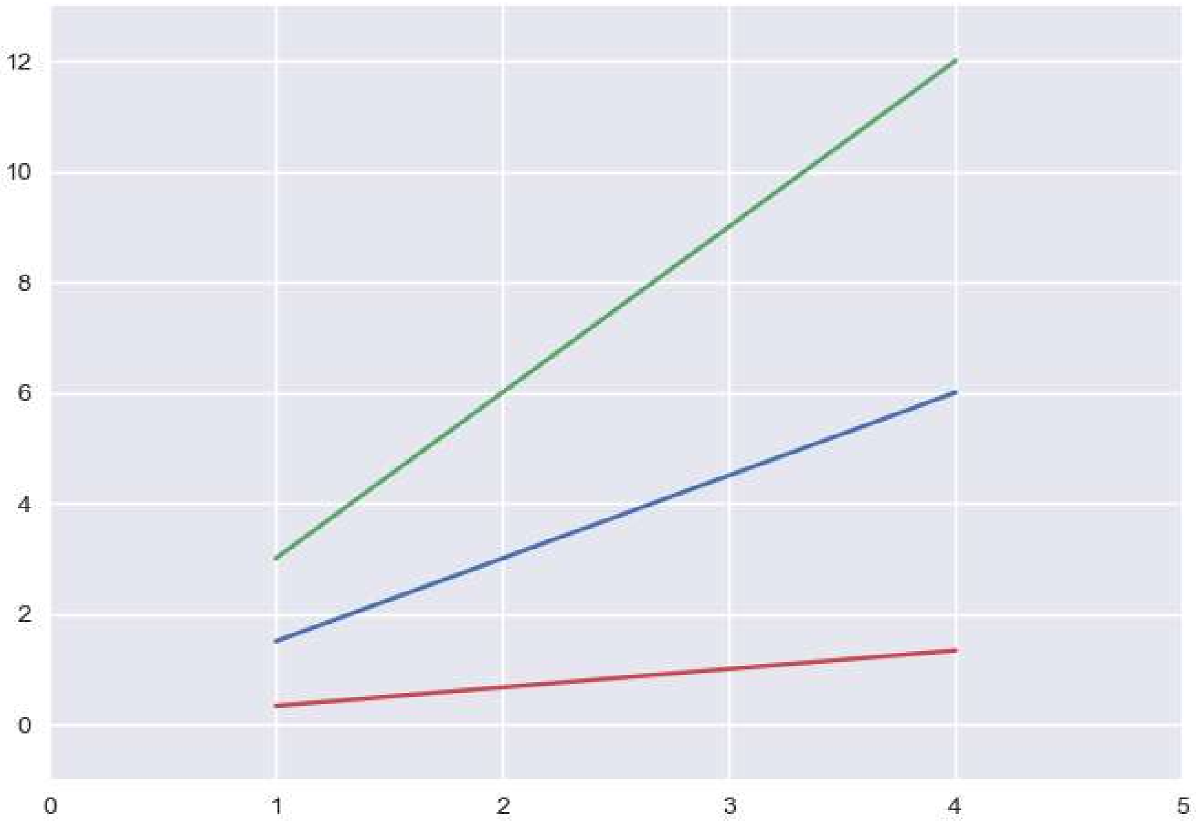
```
x15 = np.arange(1, 5)
plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
plt.grid(True)
plt.show()
```



Handling axes

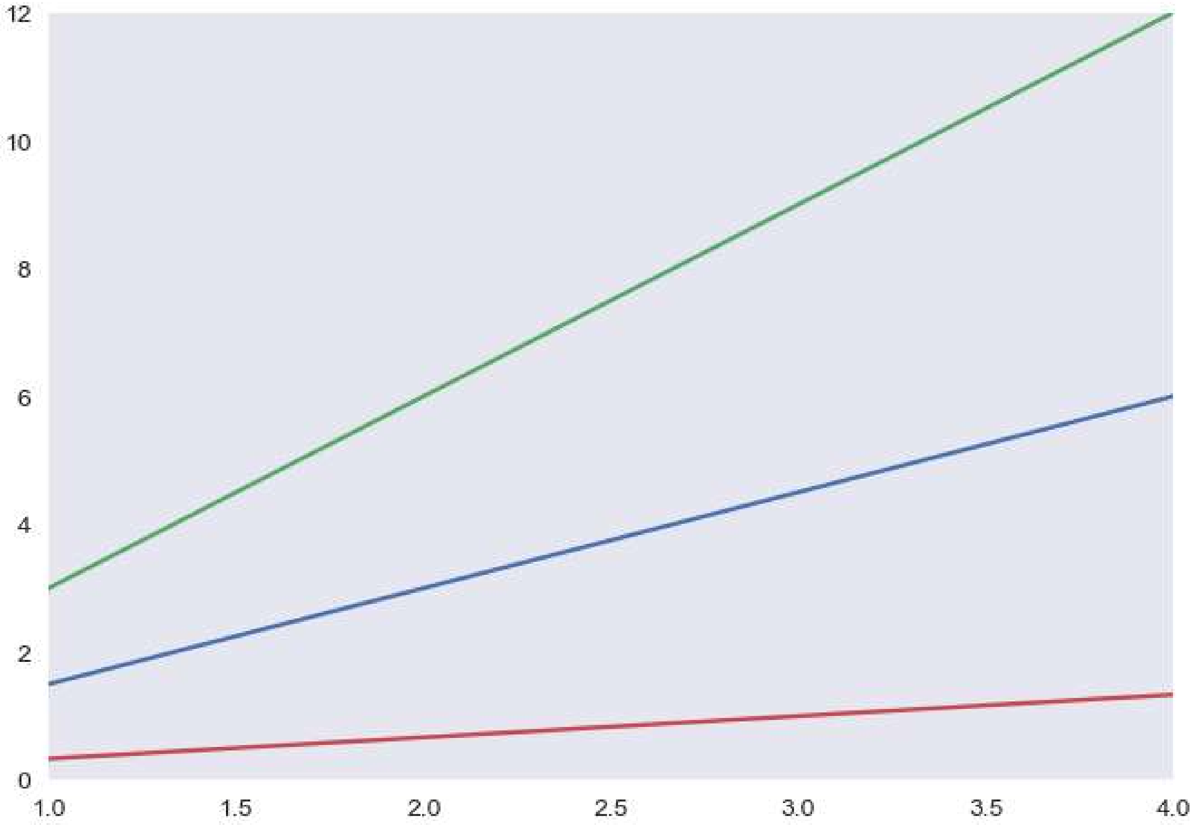
In [275…

```
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 [277…

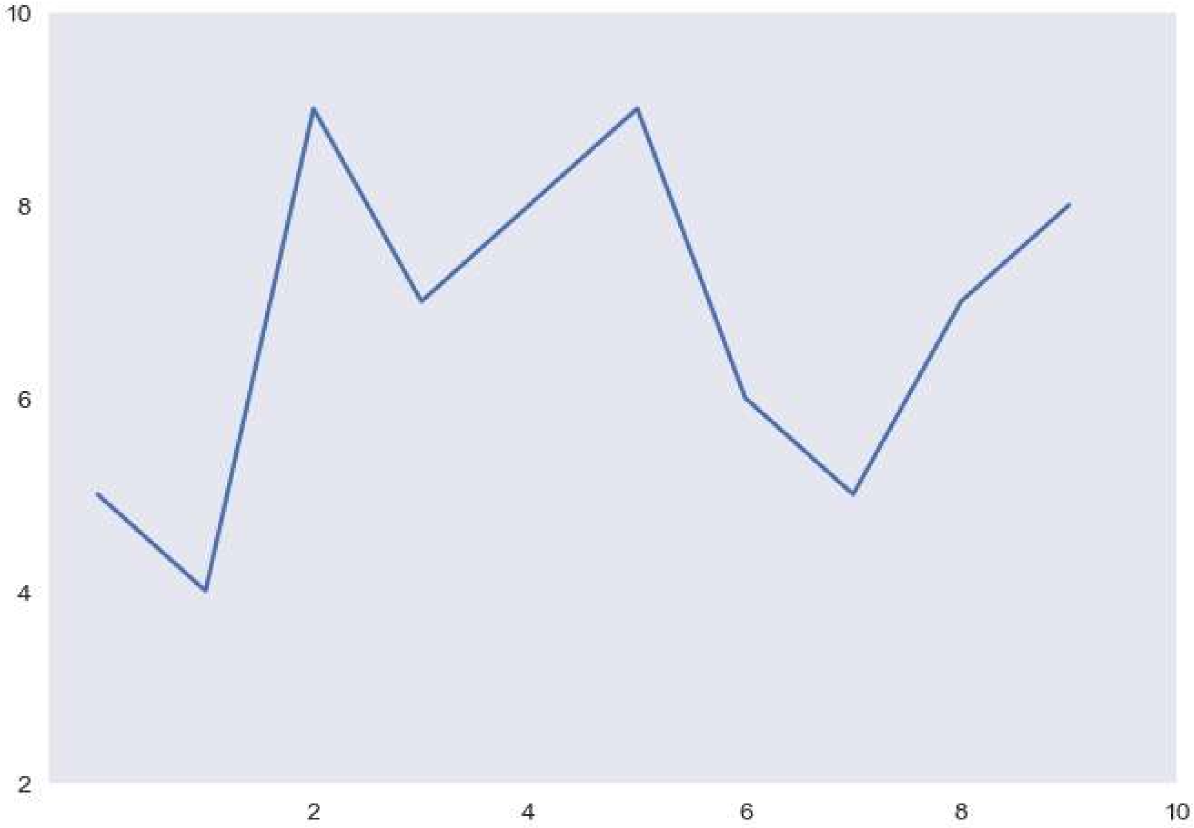
```
x15 = np.arange(1, 5)
plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
plt.xlim([1.0, 4.0])
plt.ylim([0.0, 12.0])
plt.grid(False)
plt.show()
```



Handling X and Y ticks

In [280…

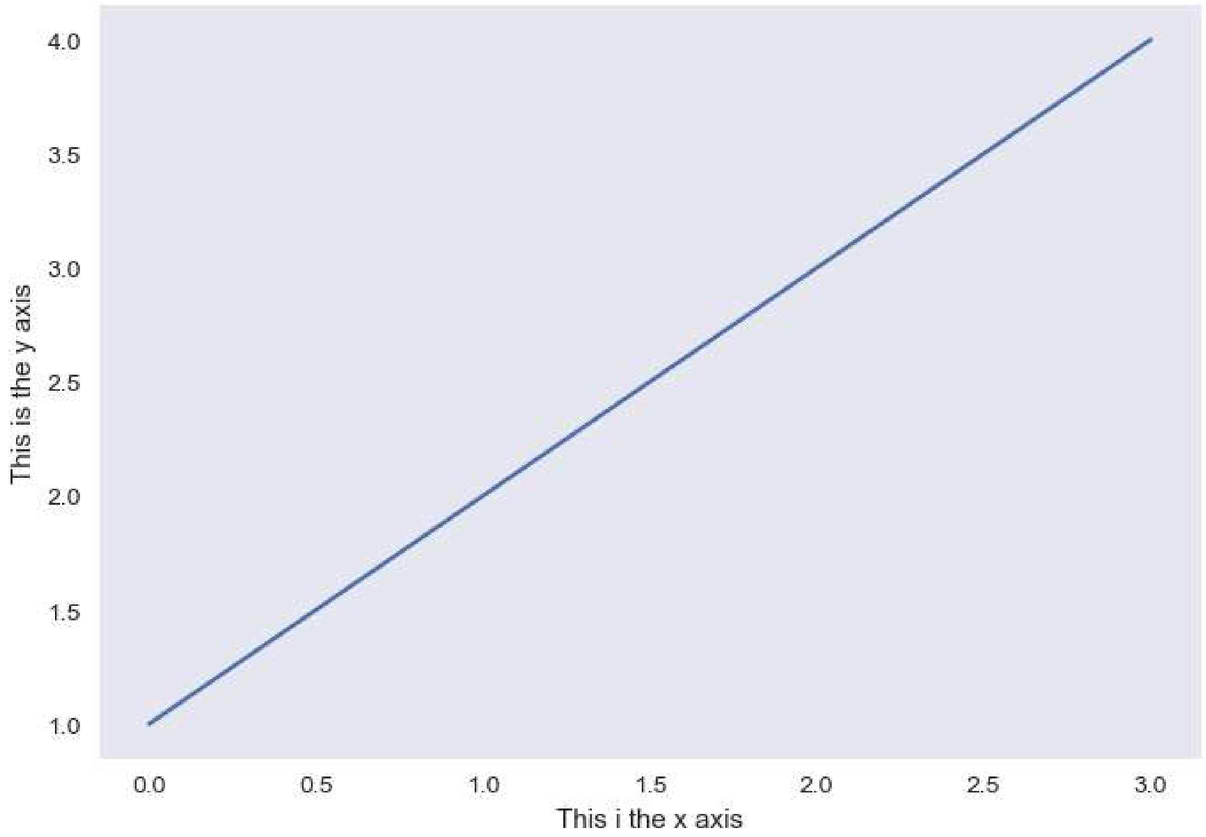
```
u = [5, 4, 9, 7, 8, 9, 6, 5, 7, 8]
plt.plot(u)
plt.xticks([2, 4, 6, 8, 10])
plt.yticks([2, 4, 6, 8, 10])
plt.grid(False)
plt.show()
```



Adding labels

In [283…

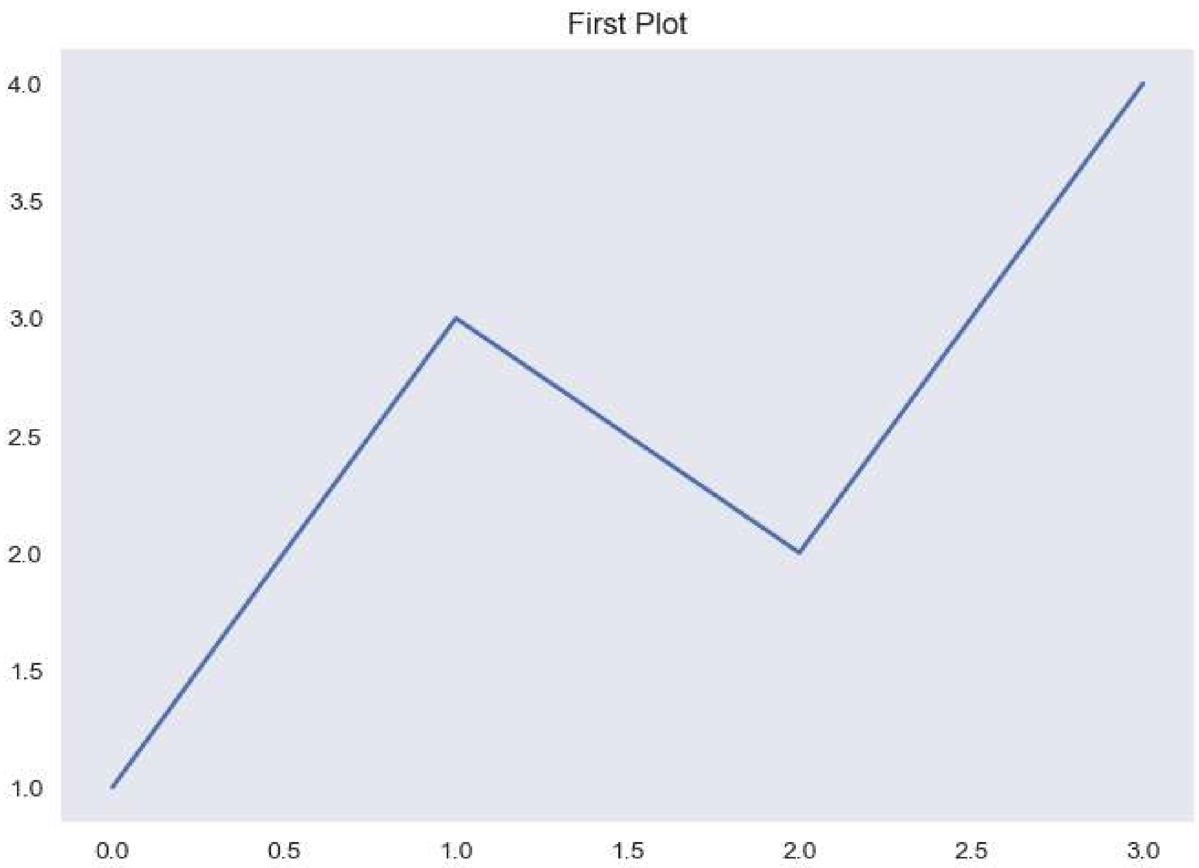
```
plt.plot([1,2,3,4])
plt.xlabel('This i the x axis')
plt.ylabel('This is the y axis')
plt.grid()
plt.show()
```



Adding title

In [286…

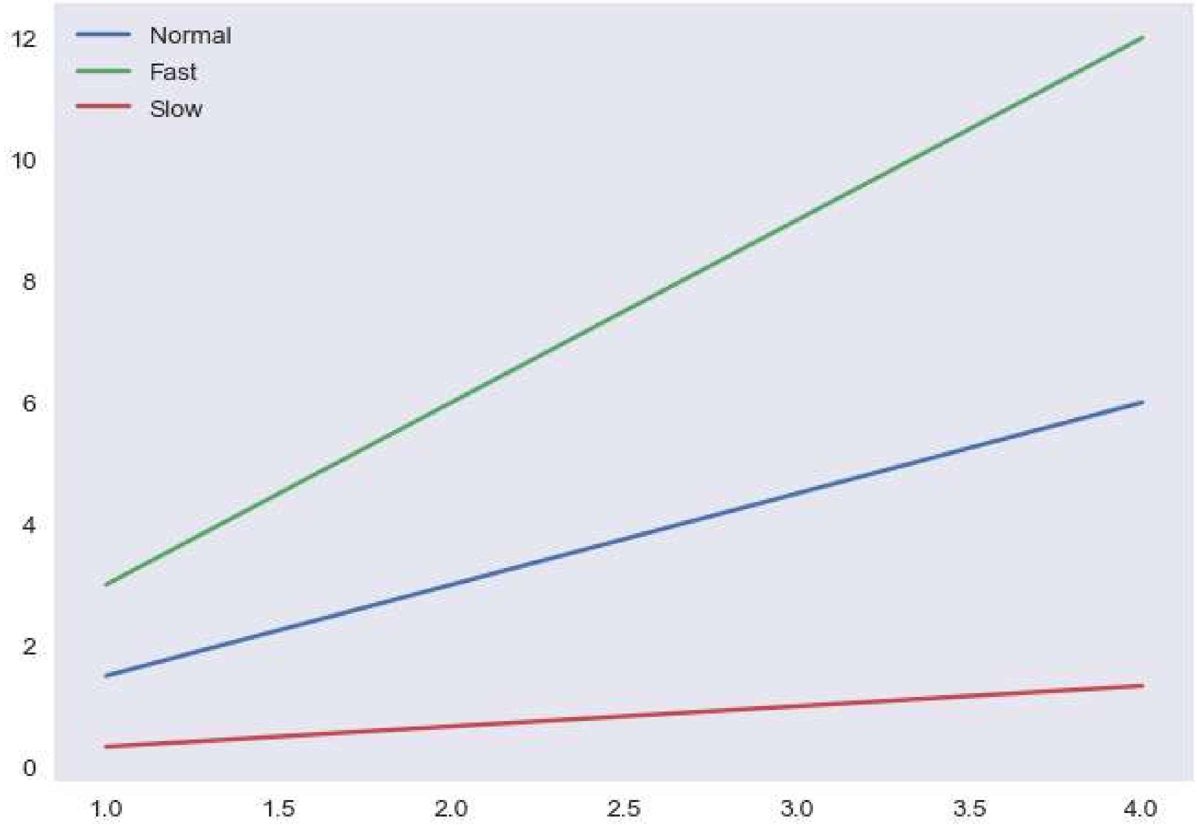
```
plt.plot([1, 3, 2, 4])
plt.title('First Plot')
plt.grid(None)
plt.show()
```



Adding legends

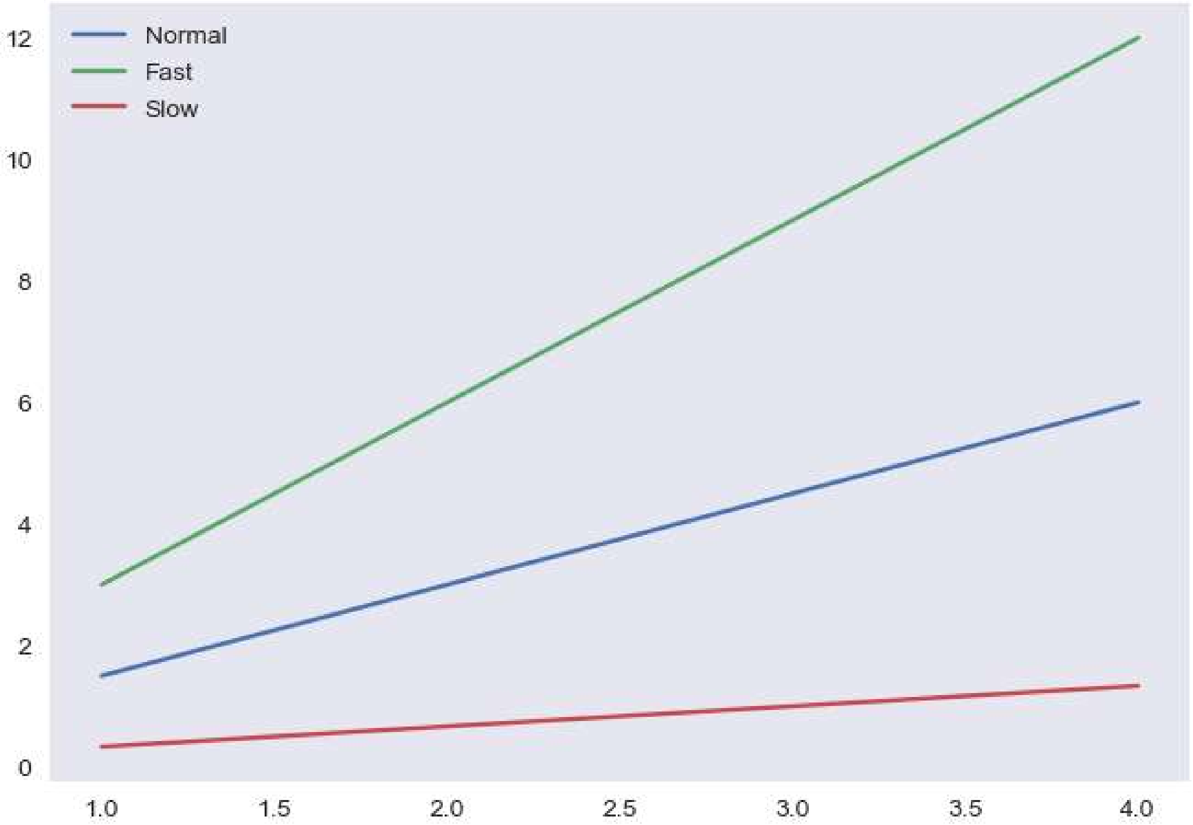
In [289…

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



In [291…

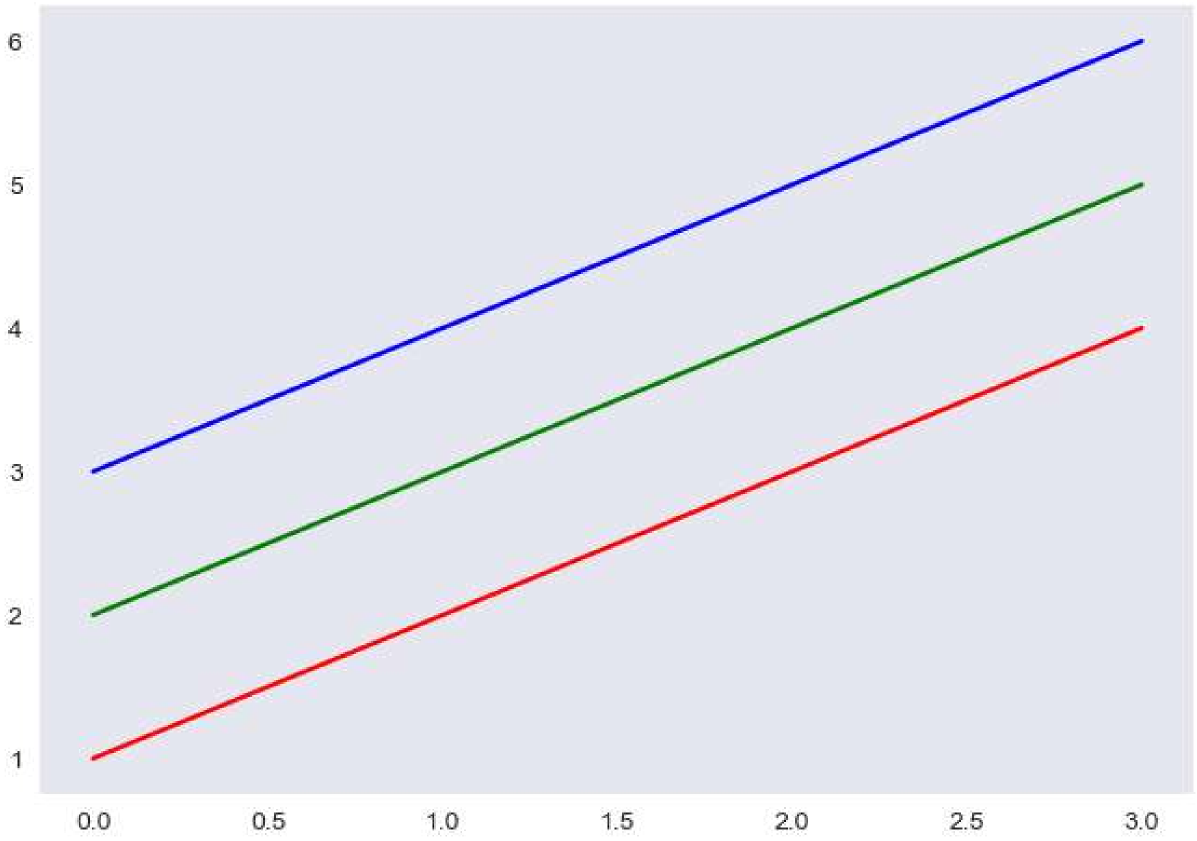
```
x15 = np.arange(1, 5)
fig, ax = plt.subplots()
ax.plot(x15, x15*1.5, label='Normal')
ax.plot(x15, x15*3.0, label='Fast')
ax.plot(x15, x15/3.0, label='Slow')
ax.legend();
plt.grid()
plt.show()
```



Control colours

In [294…

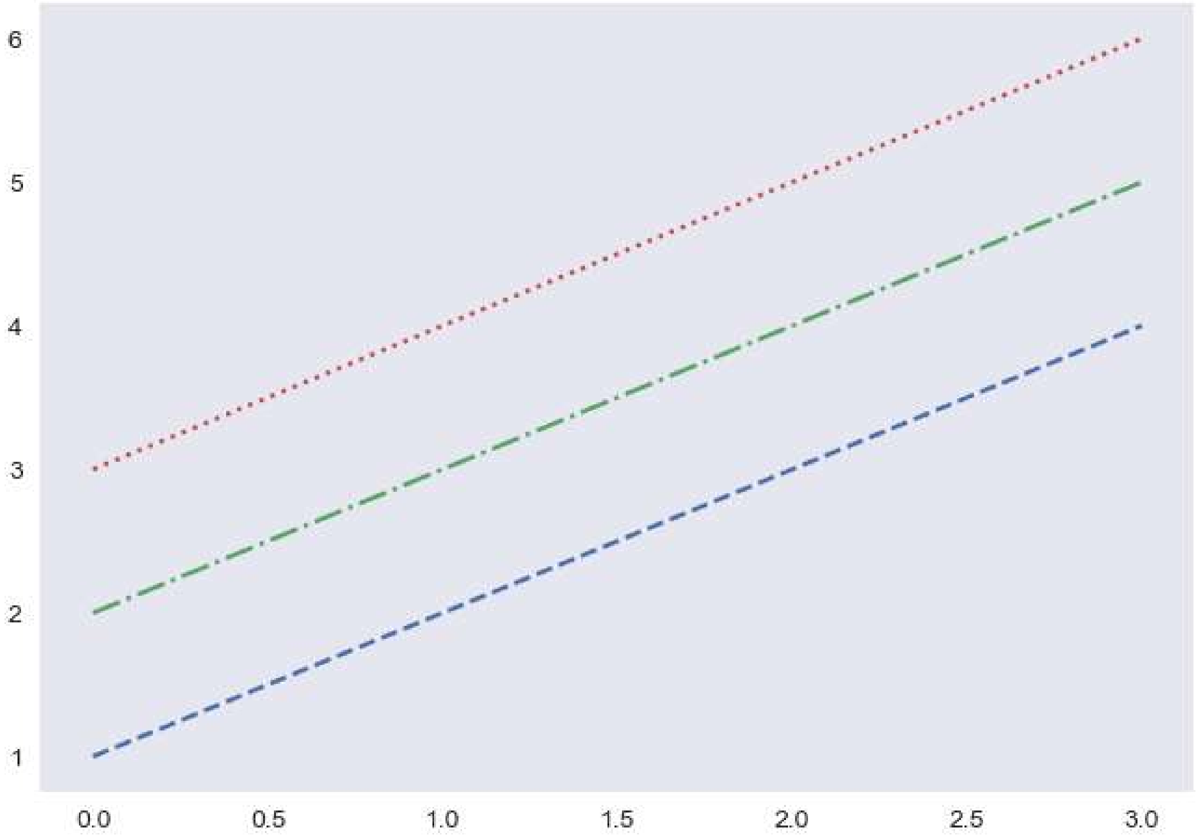
```
x16 = np.arange(1, 5)
plt.plot(x16, 'r')
plt.plot(x16+1, 'g')
plt.plot(x16+2, 'b')
plt.grid()
plt.show()
```



Control line styles

In [297…

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


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