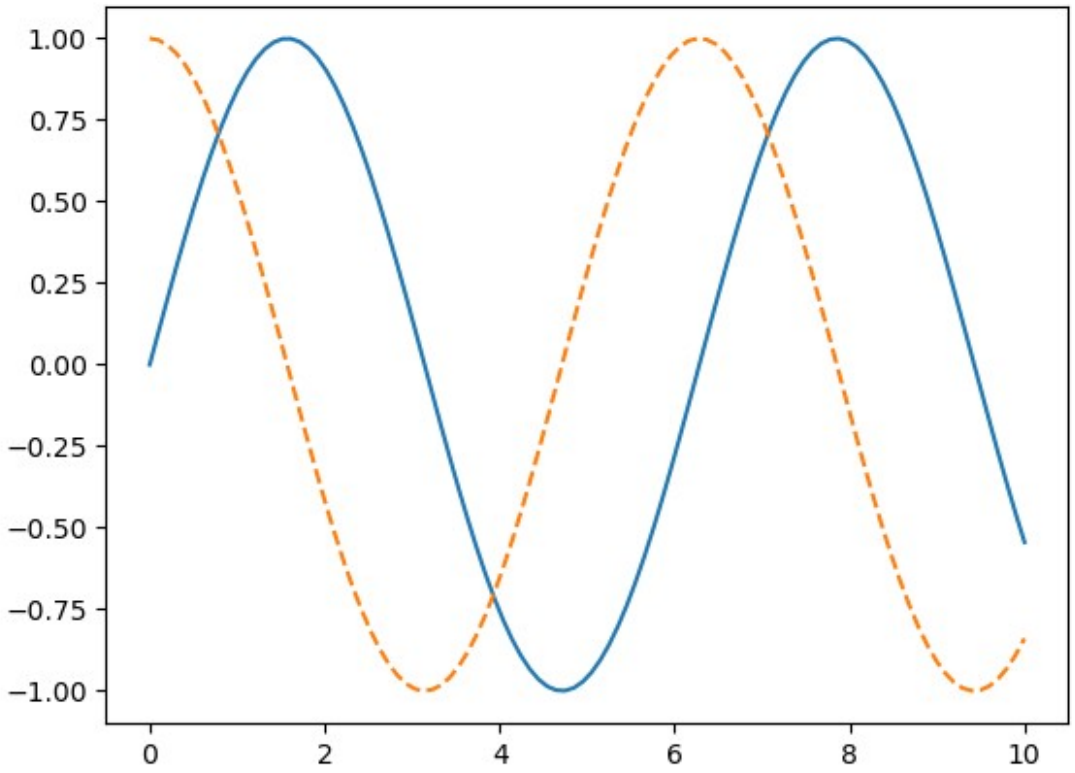
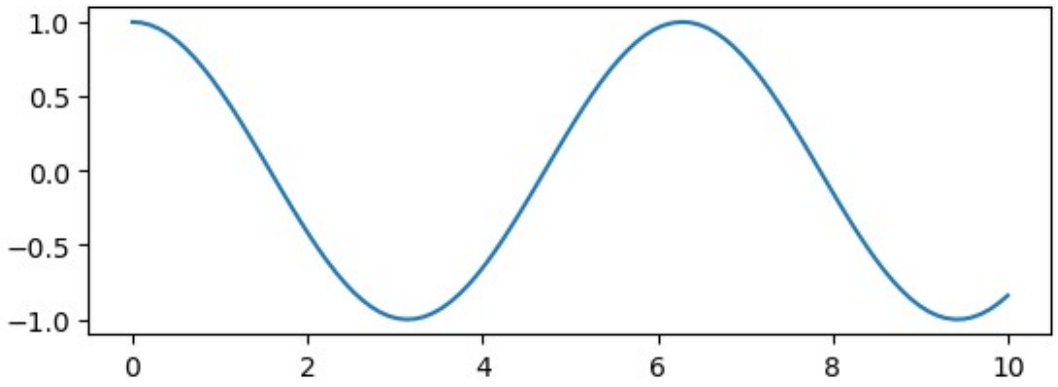
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

%matplotlib inline
x1 = np.linspace(0,10,100)

fig = plt.figure()

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



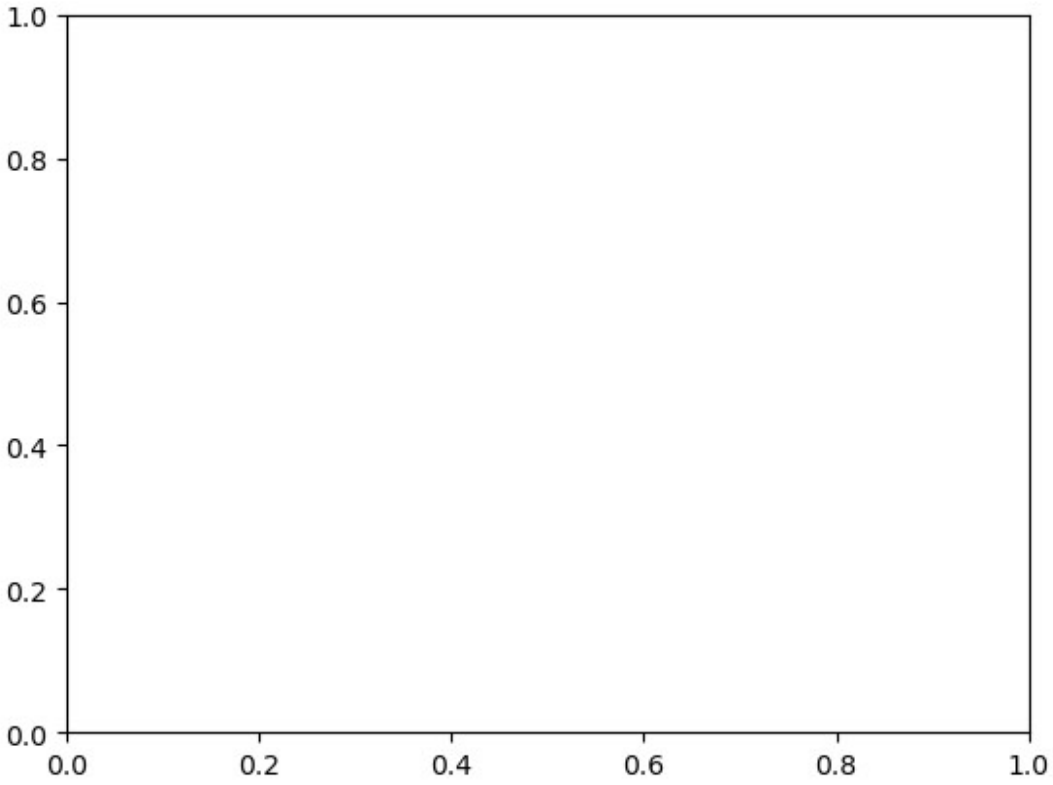

```
PyPlot API

plt.figure()

<Figure size 640x480 with 0 Axes>

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

[<matplotlib.lines.Line2D at 0xe301130>]
```

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



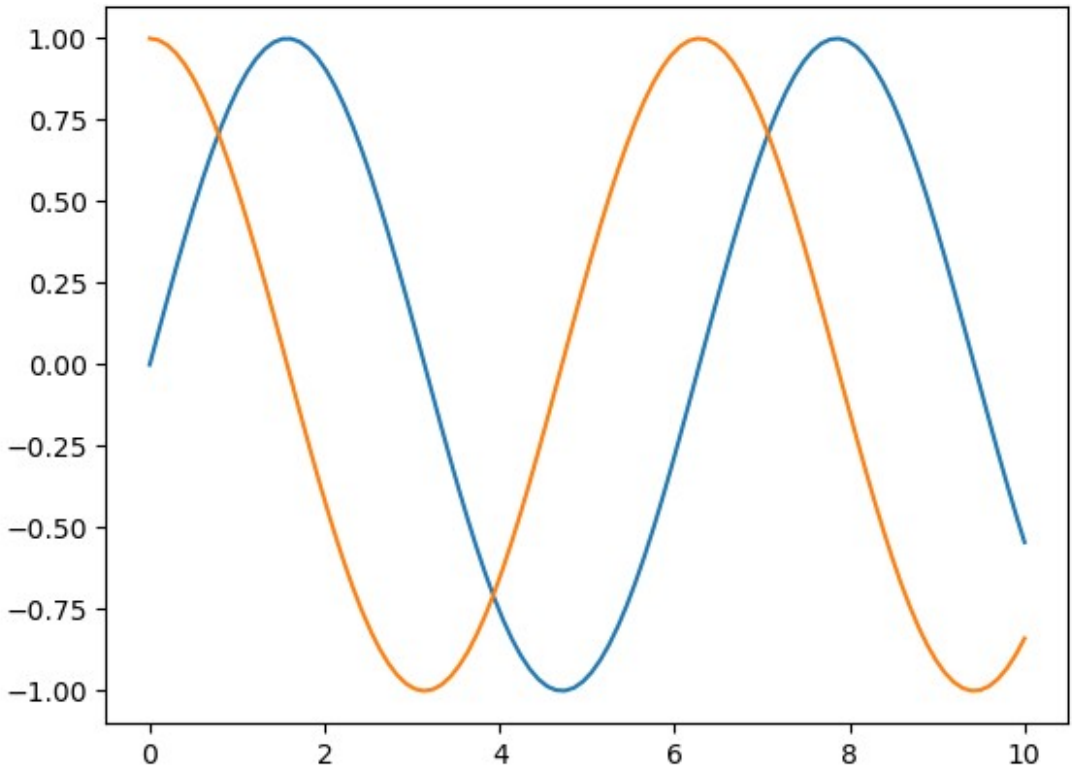

```
print(plt.gca())
plt.show()
```

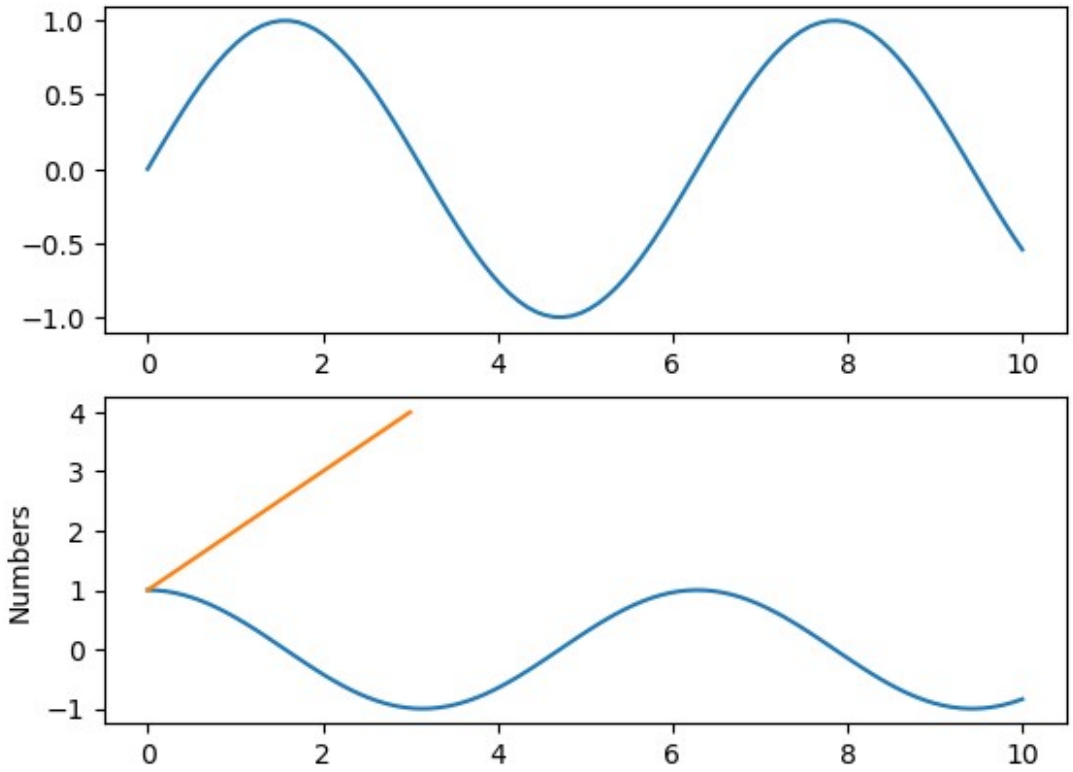
```
Axes(0.125,0.11;0.775x0.77)
```



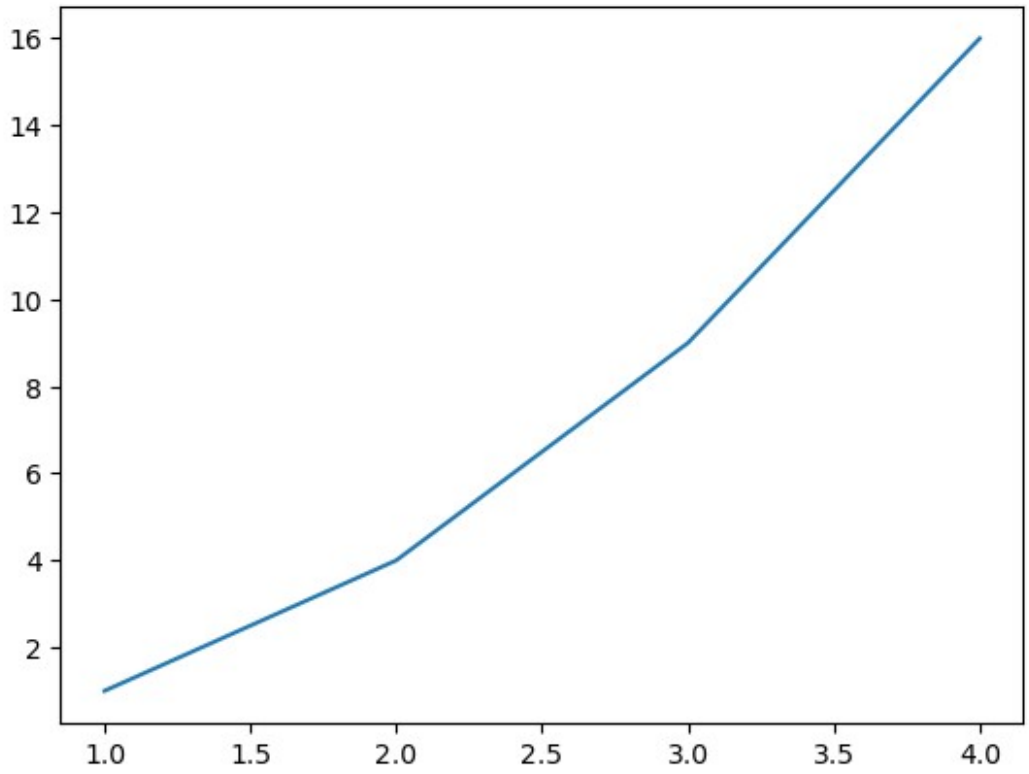

```
Visualization With Pyplot
```

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






```
plot()-A Versatile Command

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

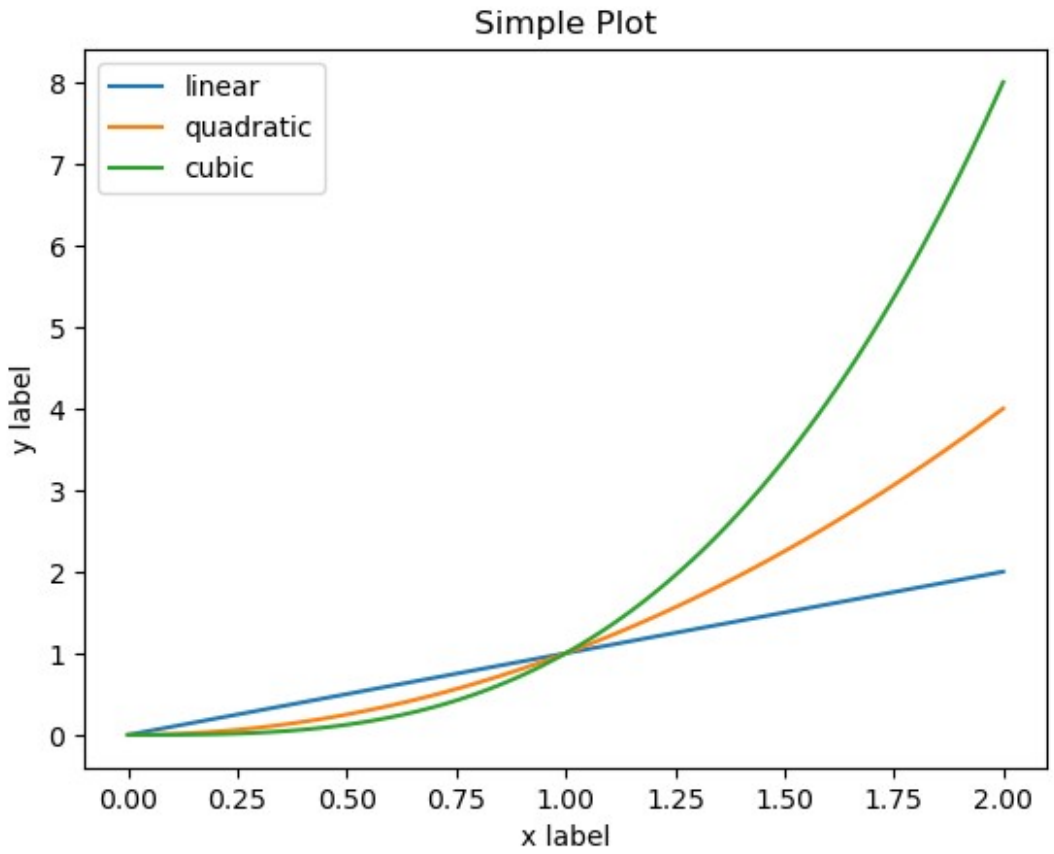


State Machine Interface

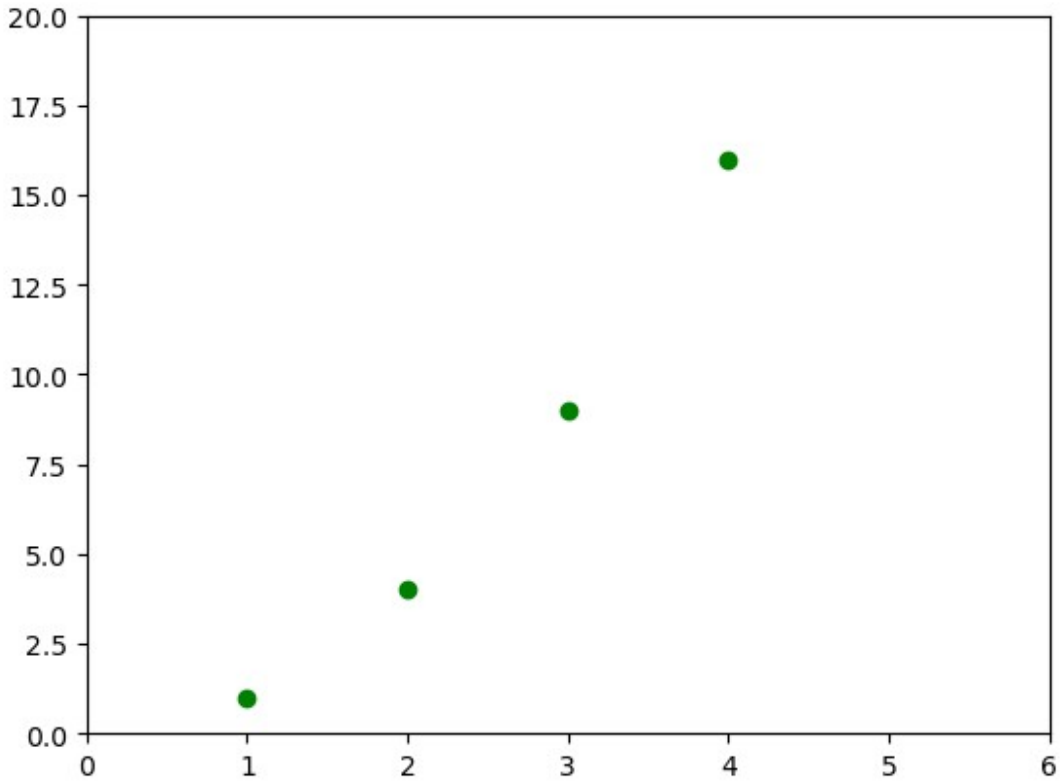
```
x = np.linspace(0,2,100)

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

```
[<matplotlib.lines.Line2D at 0x139d9a0>]
```

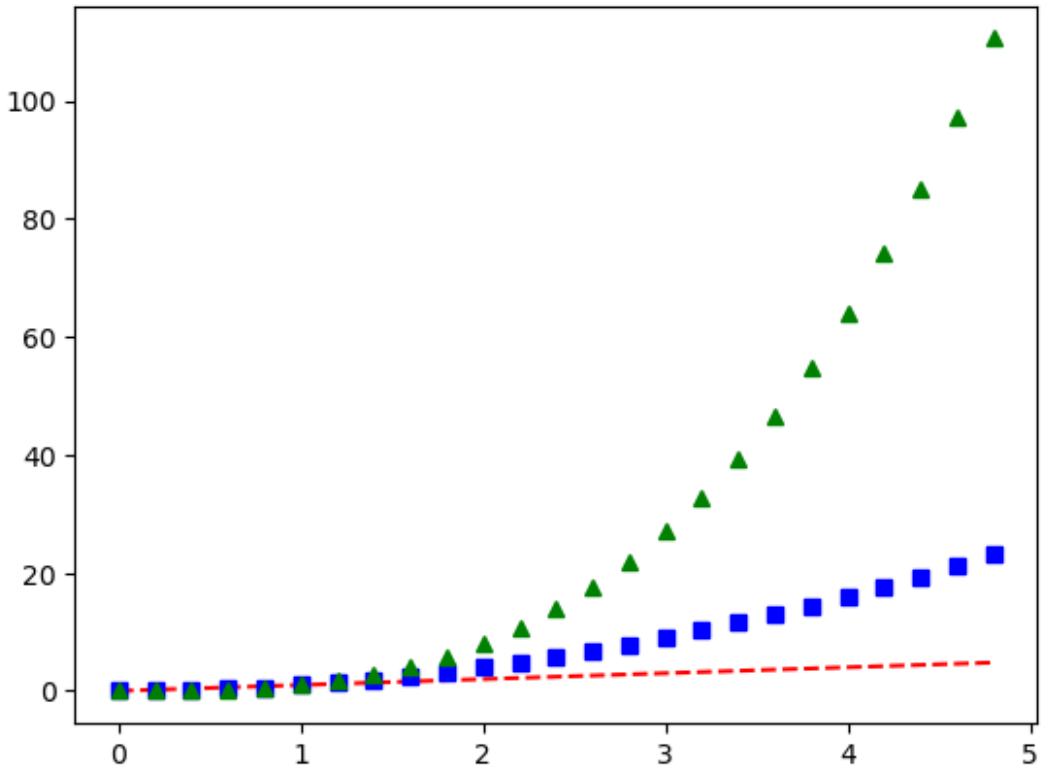
```
plt.xlabel('x label')
plt.ylabel('y label')
plt.title("Simple Plot")
plt.legend()
plt.show()
```



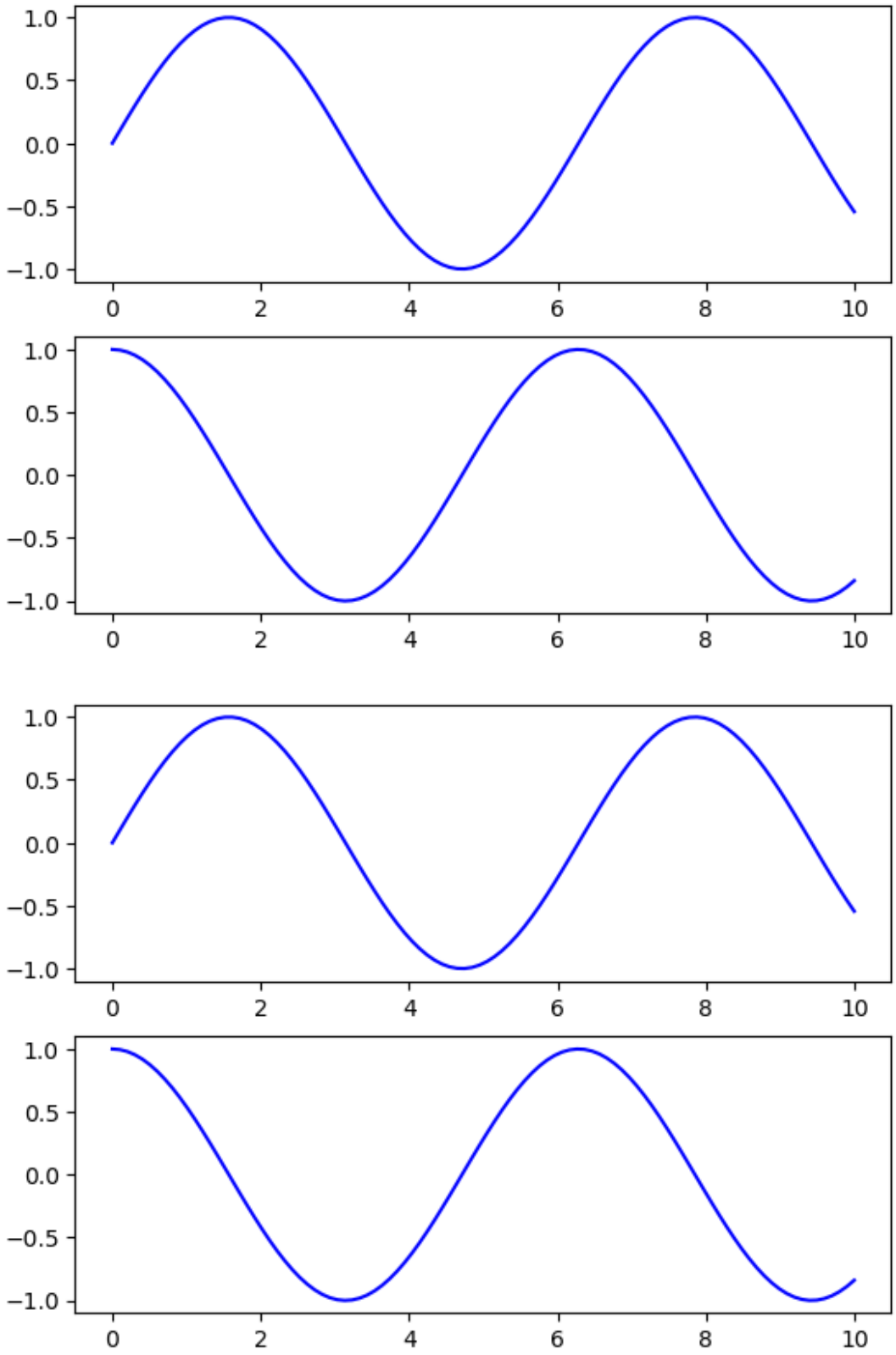

```
Formatting the style of plot

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




```
Working with Numpy array

t = np.arange(0.,5.,0.2)
plt.plot(t, t, 'r--',t,t**2, 'bs', t, t**3, 'g^')
plt.show()
```

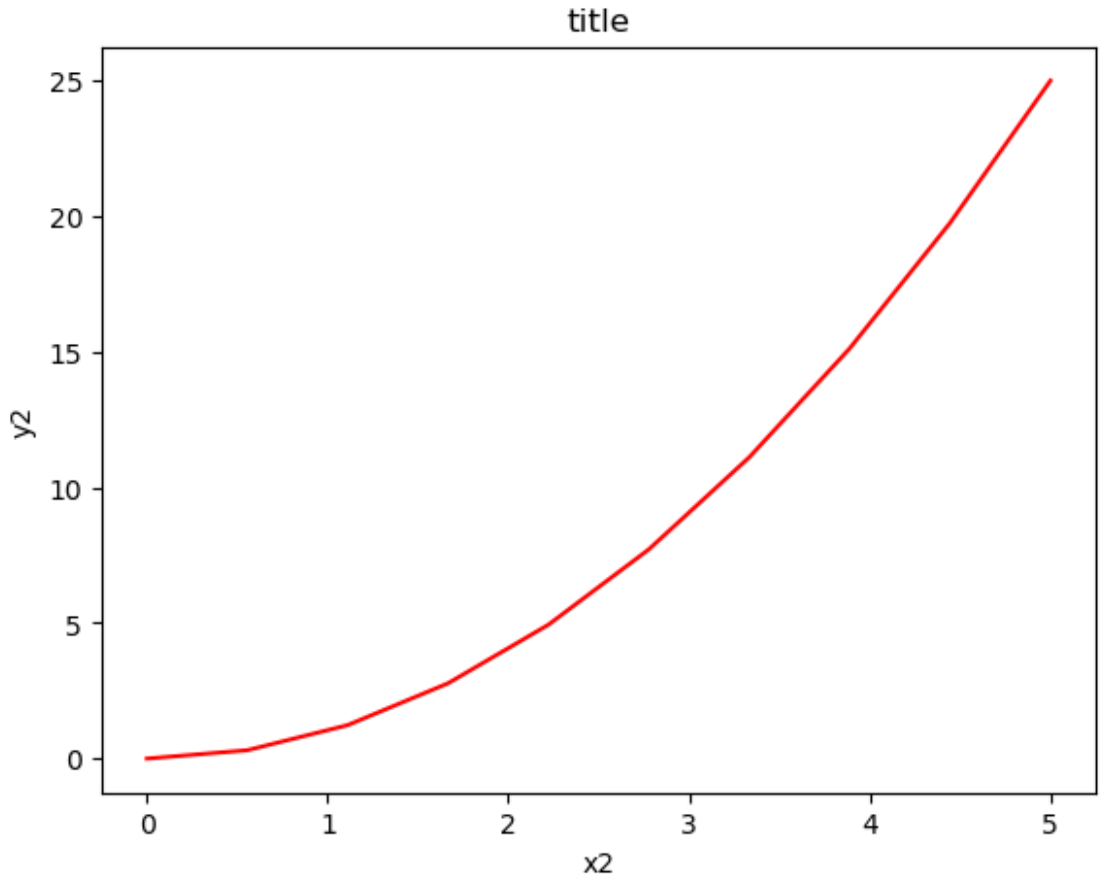


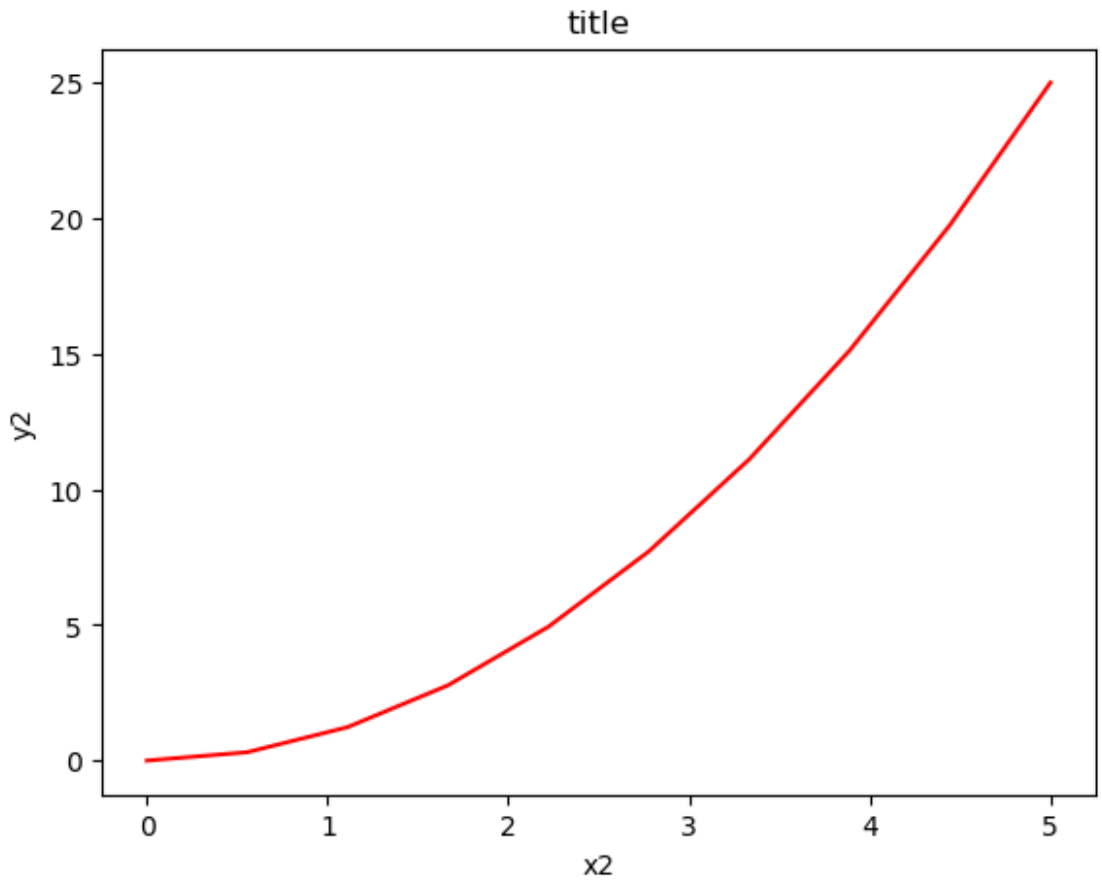

```
Object-Oriented API

fig, ax = plt.subplots(2)

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



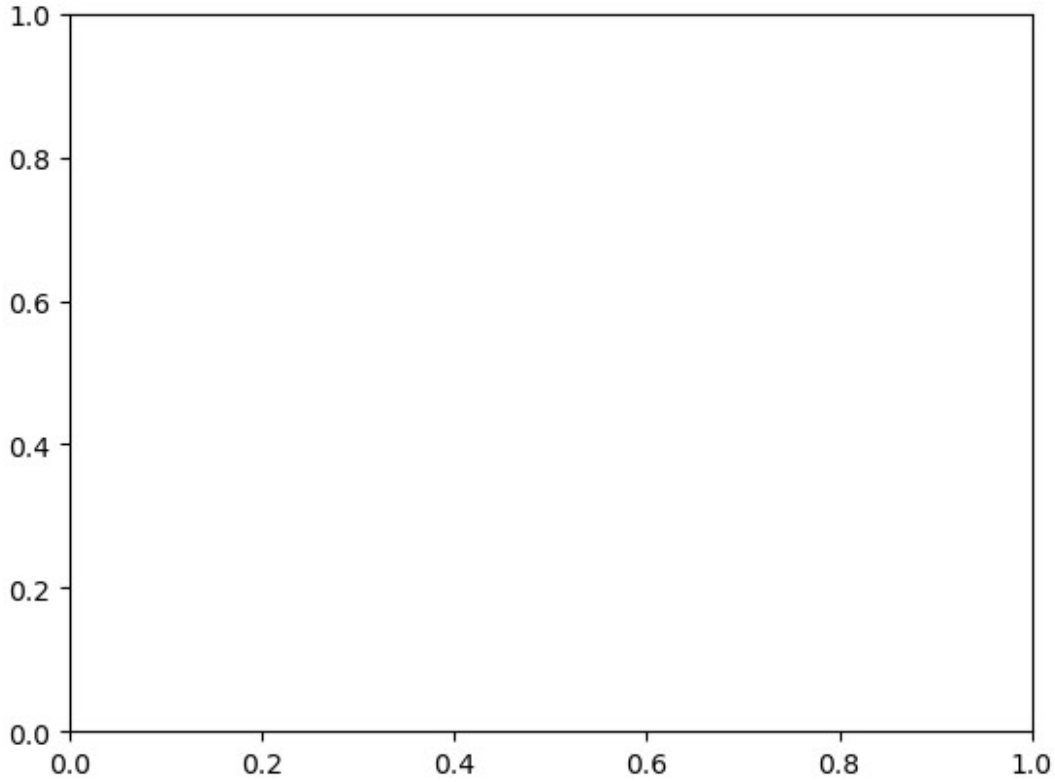

```
Objects And Reference
```

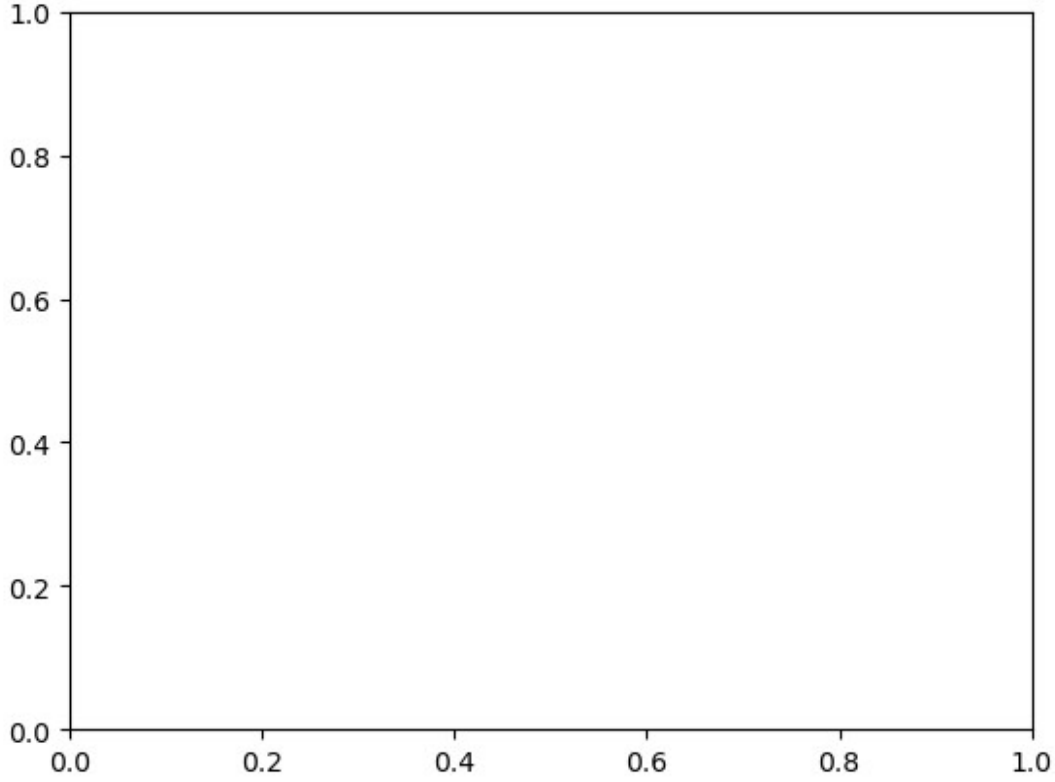
```
fig = plt.figure()
x2 = np.linspace(0,5,10)
y2 = x2 ** 2
axes = fig.add_axes([0.1,0.1,0.8,0.8])
axes.plot(x2, y2, 'r')

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



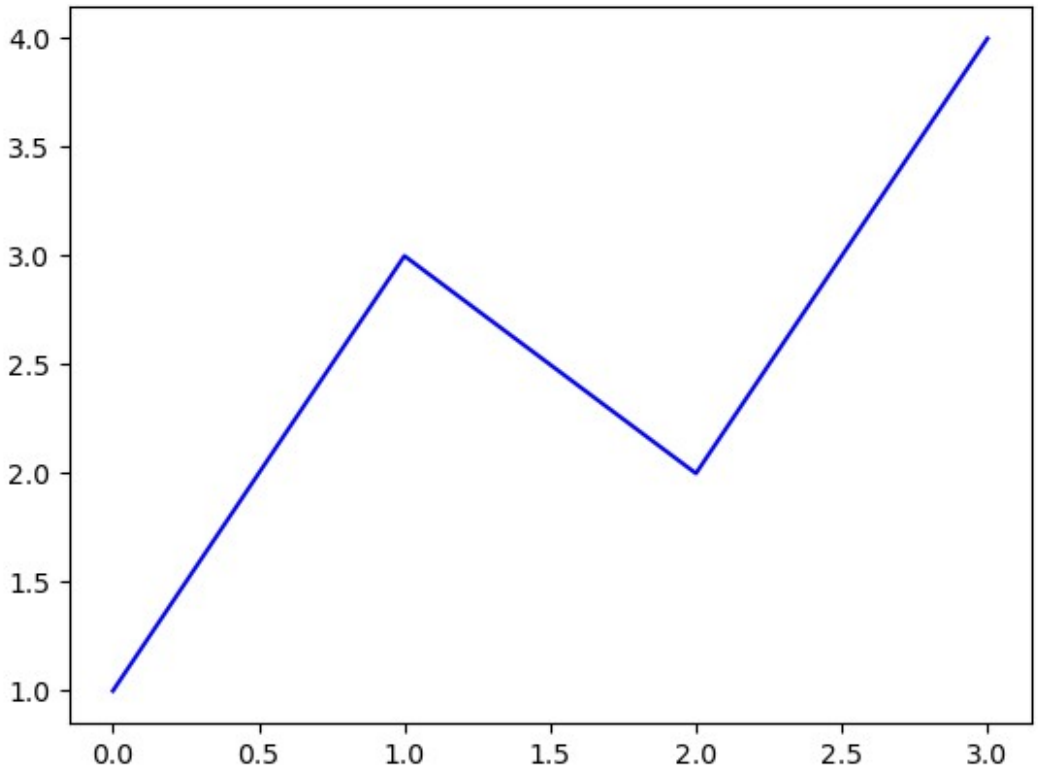



```
Figure And Axes

fig=plt.figure()
ax=plt.axes()
plt.show()
```

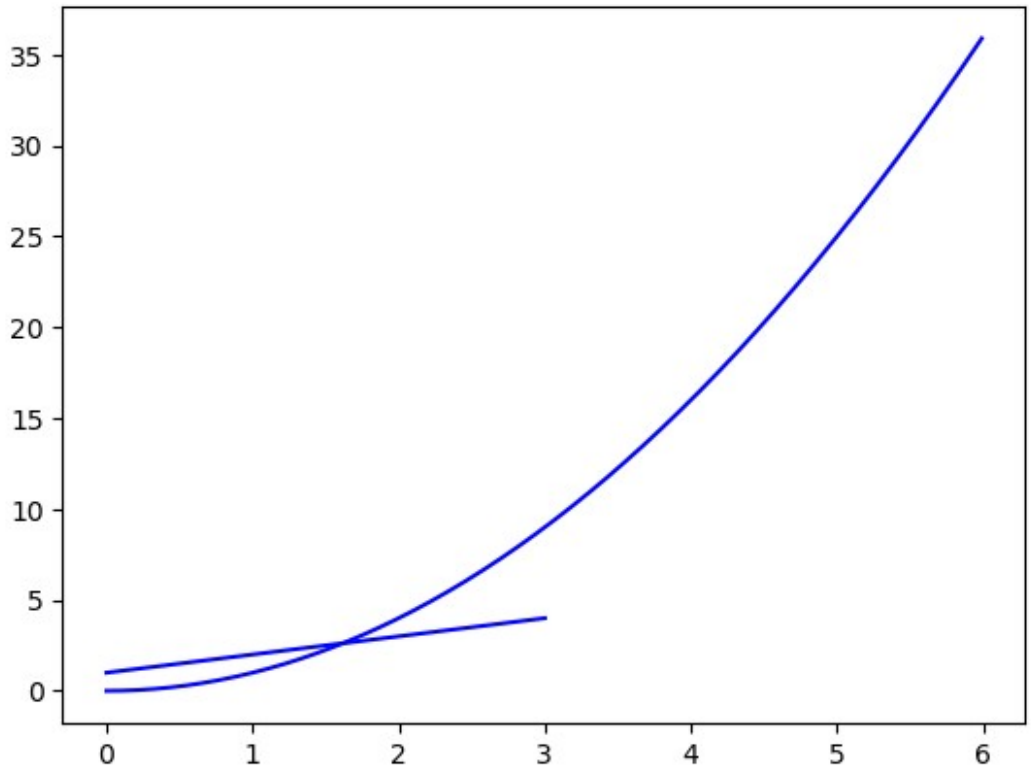

1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

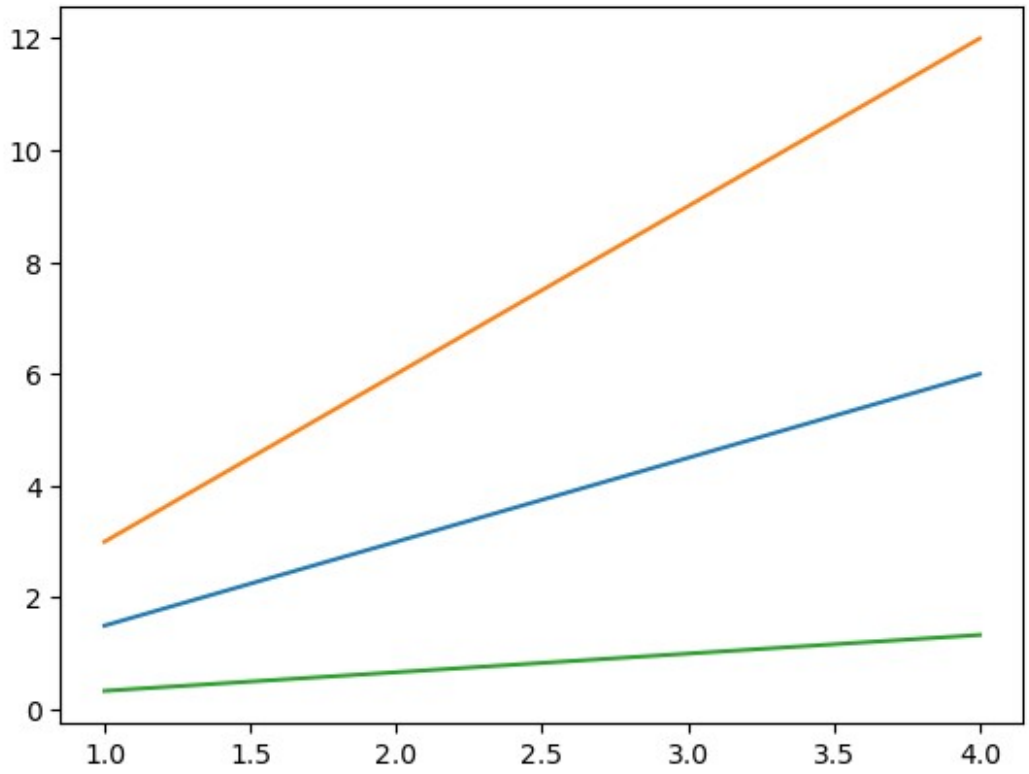
```
First plot with Matplotlib

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




```
plt.plot([1,2,3,4],'b-')
x3 = np.arange(0.0, 6.0, 0.01)
plt.plot(x3, [xi**2 for xi in x3], 'b-')
plt.show()
```

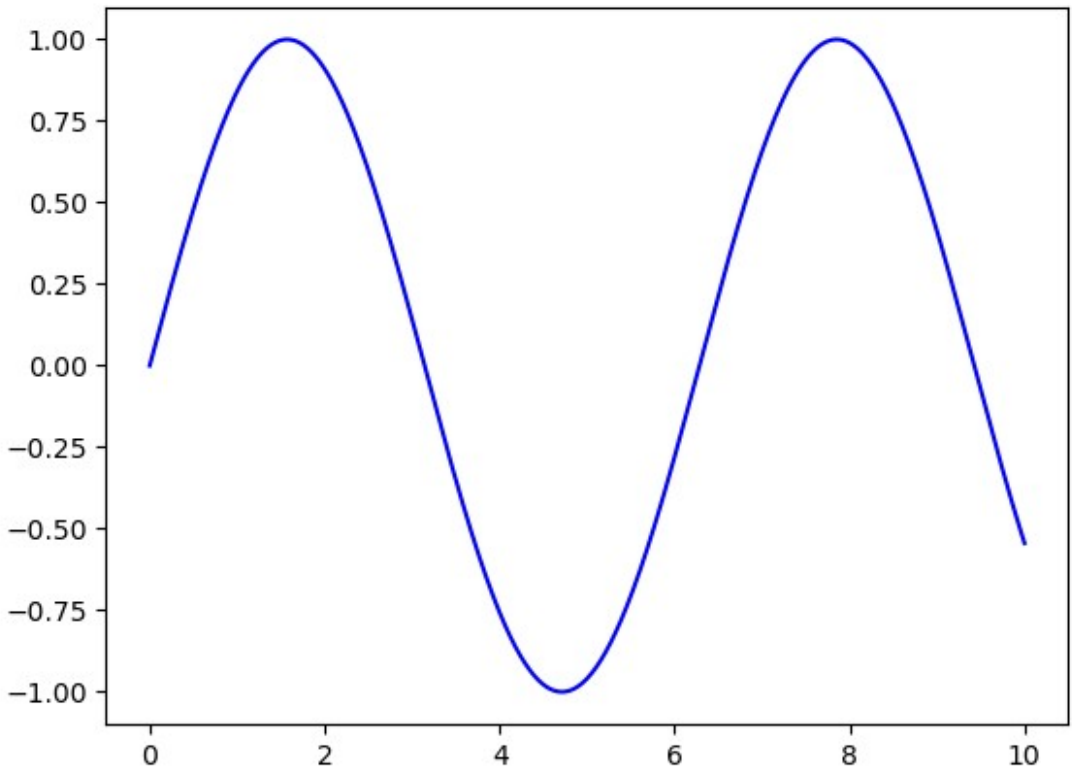


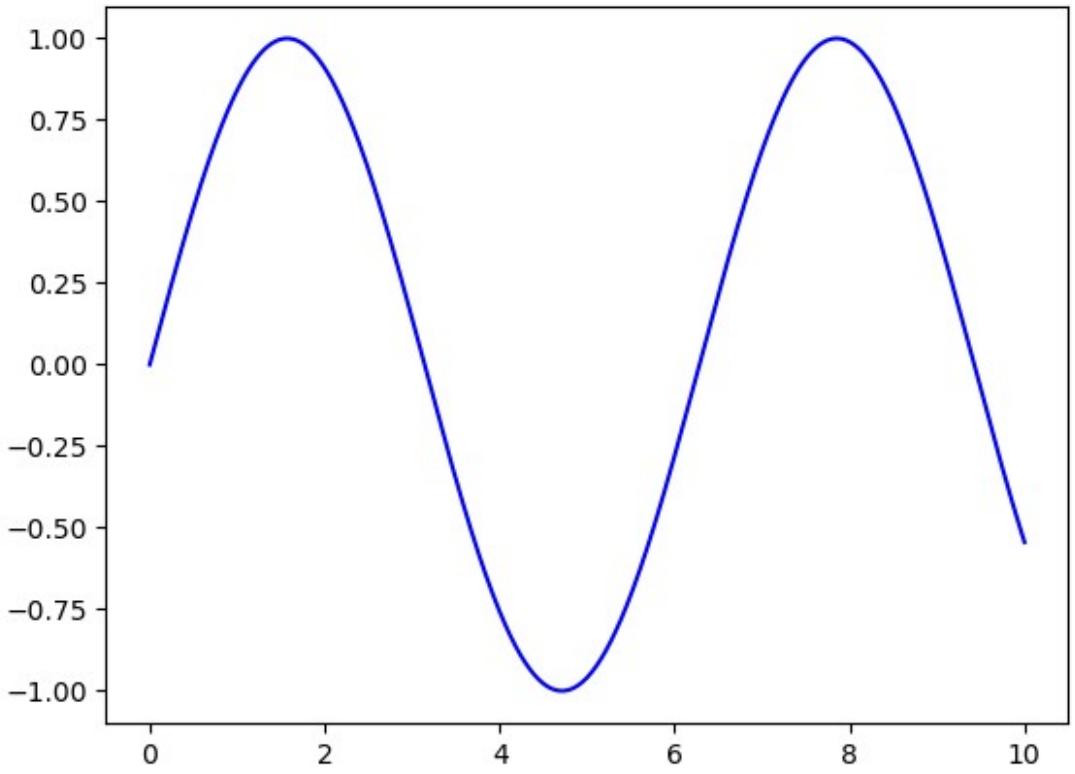

```
Multiline  plots

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()
```




```
Saving the plot

fig.savefig('plot1.png')
from Ipython.display import Image
Image('plot1.png')

---------------------------------------------------------------------------
-----
ModuleNotFoundError                       Traceback (most recent call
last)
Cell In[67], line 2
      1 fig.savefig('plot1.png')
----> 2 from Ipython.display import Image
      3 Image('plot1.png')

ModuleNotFoundError: No module named 'Ipython'

fig.canvas.get_supported_filetypes()

{'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',
```

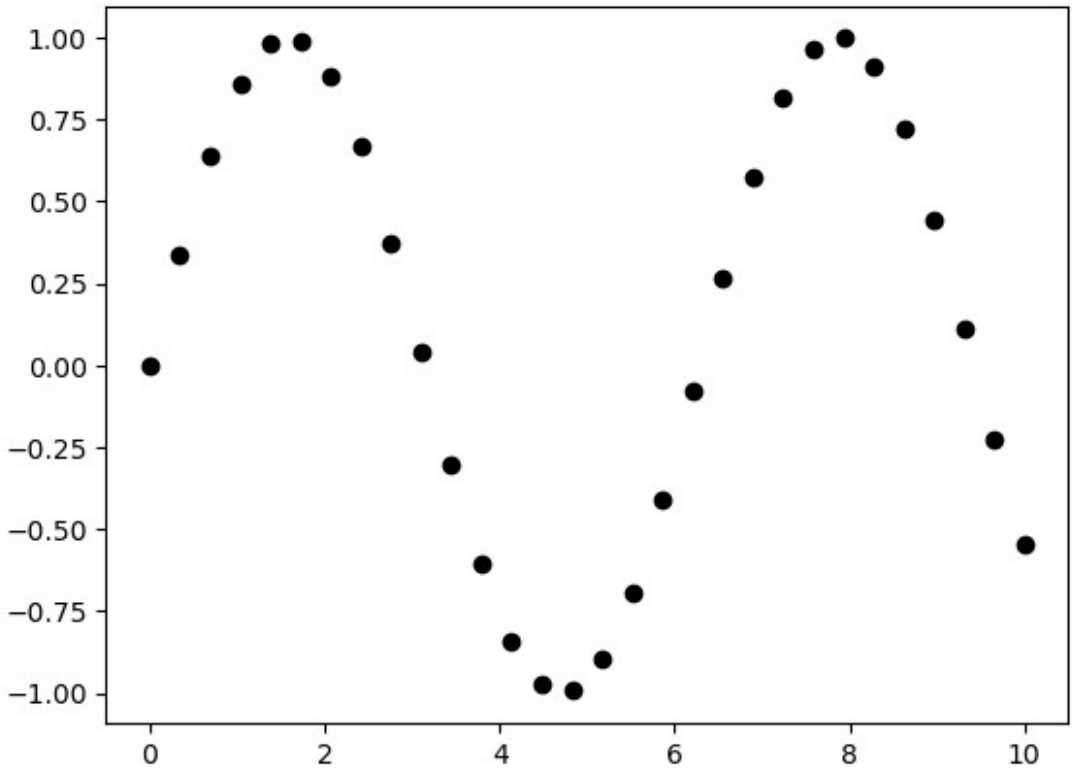
```
 'raw': 'Raw RGBA bitmap',
 'rgba': 'Raw RGBA bitmap',
 'svg': 'Scalable Vector Graphics',
 'svgz': 'Scalable Vector Graphics',
 'tif': 'Tagged Image File Format',
 'tiff': 'Tagged Image File Format',
 'webp': 'WebP Image Format'}

Line plot

fig = plt.figure()

ax = plt.axes()

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



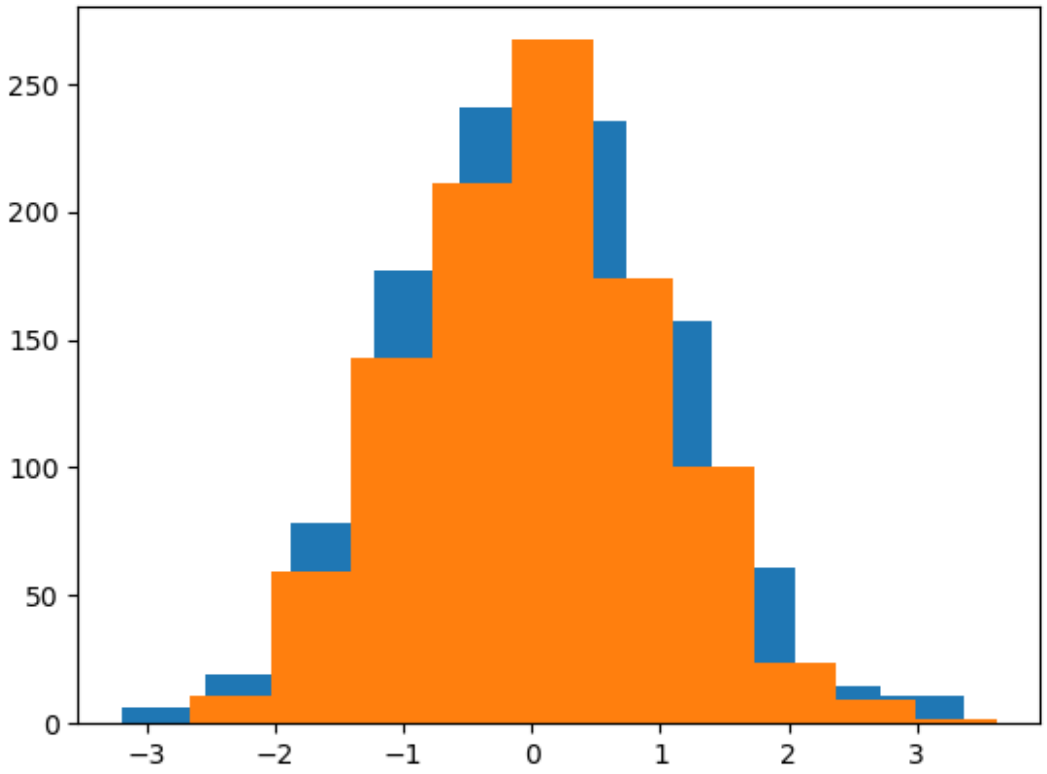


Scatter Plot

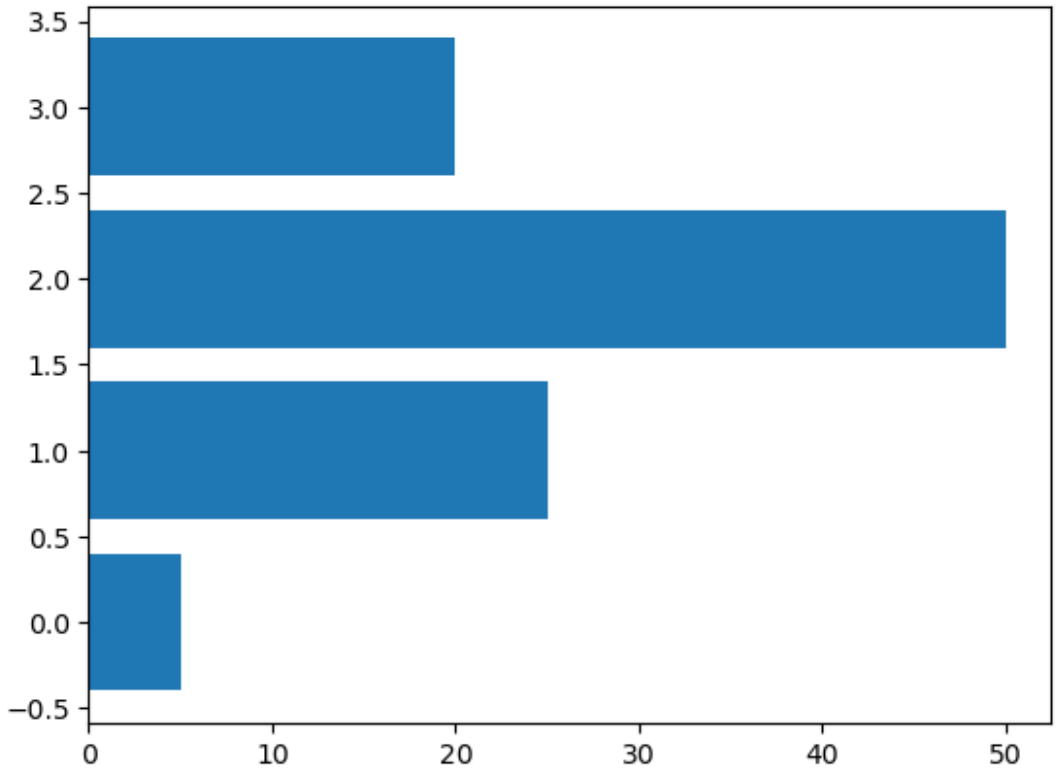
```
x7 = np.linspace(0, 10, 30)
y7 = np.sin(x7)
plt.plot(x7, y7, 'o', color = 'black');
plt.show()
```



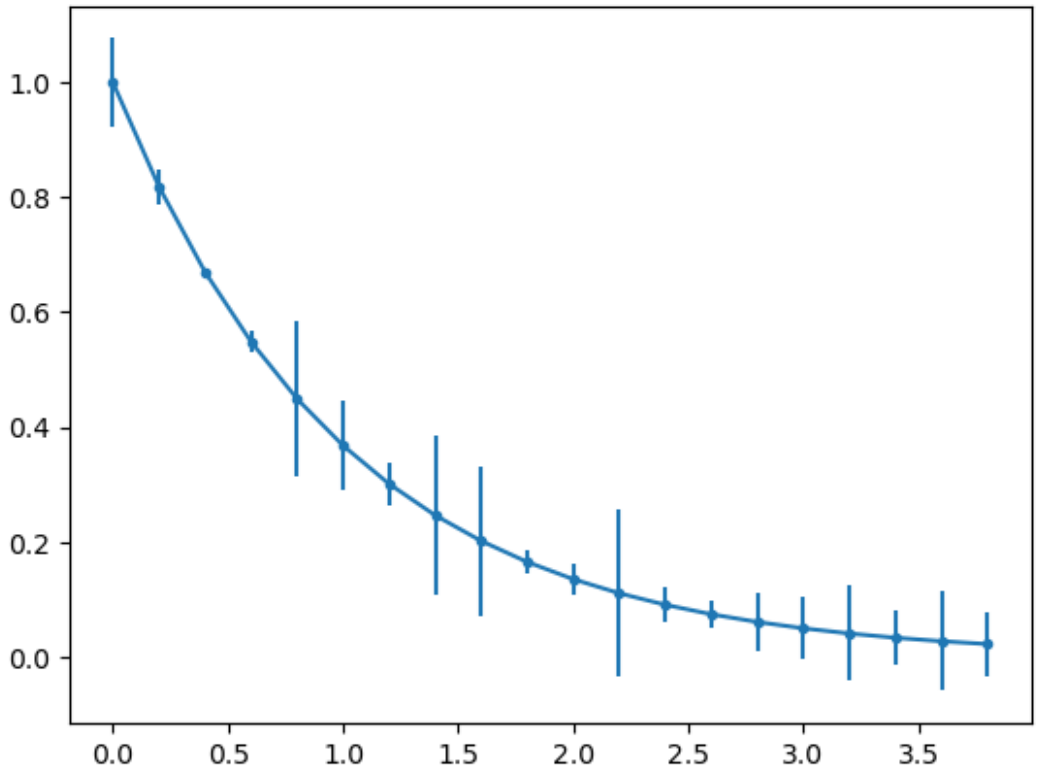

```
Histogram

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



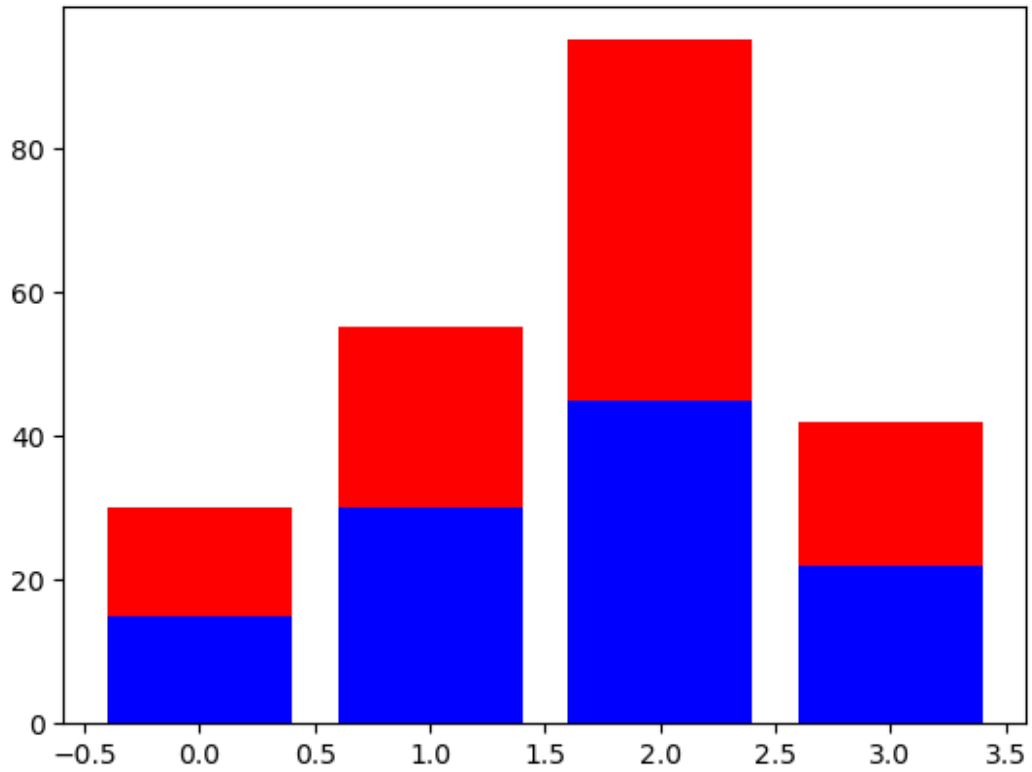

```
Horizontal bar chart

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



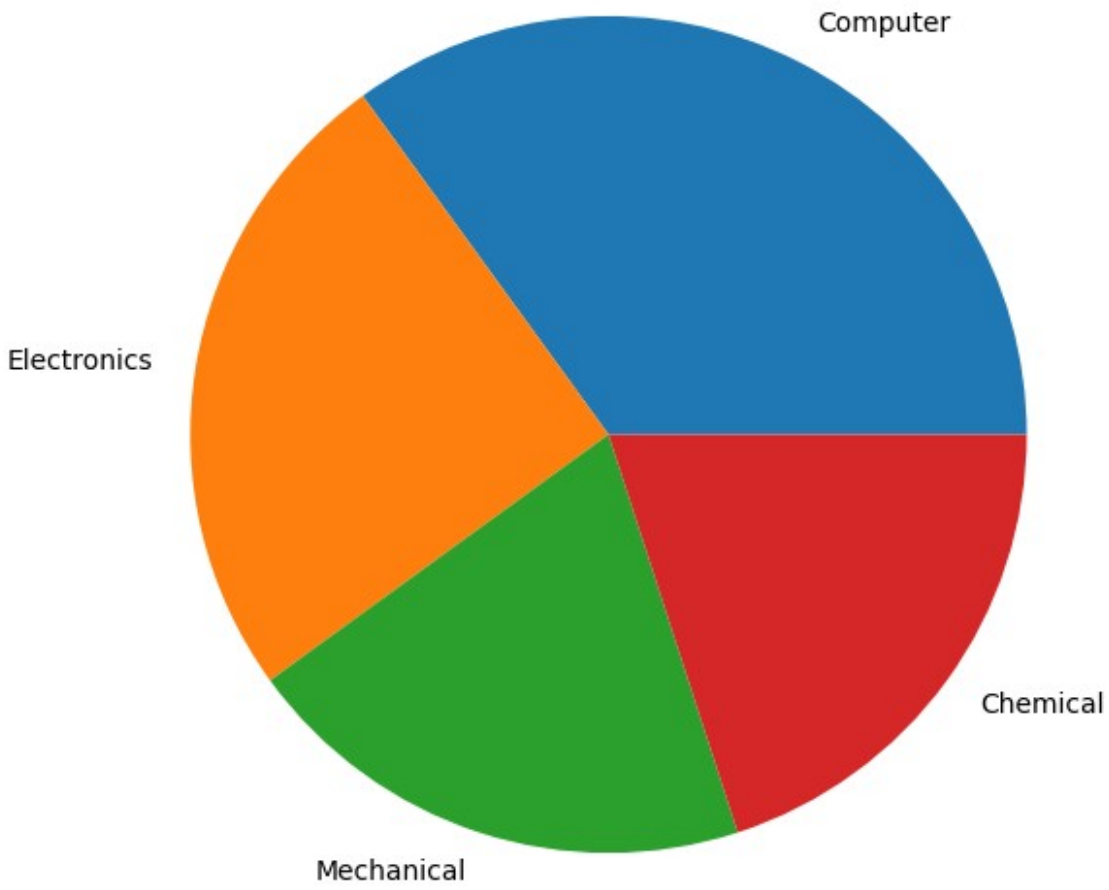

```
Error bar chart

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();
```




```
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()
```

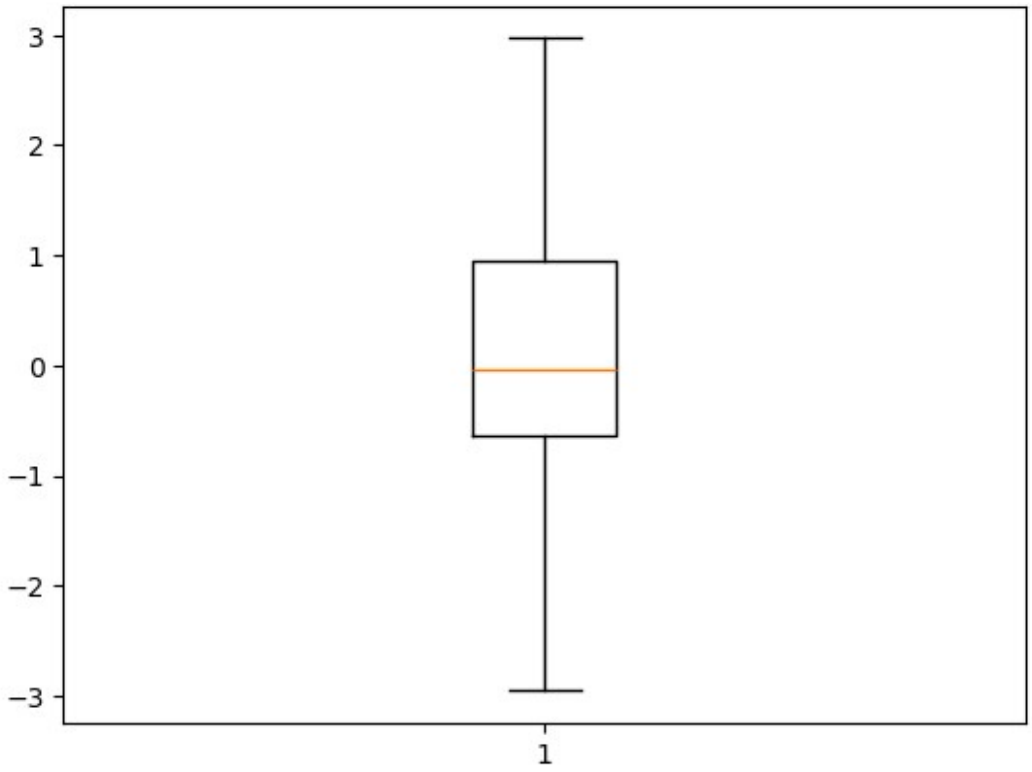



```
plt.figure(figsize=(7,7))
x10 = [35,25,20,20]
labels = ['Computer', 'Electronics', 'Mechanical', 'Chemical']
plt.pie(x10, labels=labels);
plt.show()
```

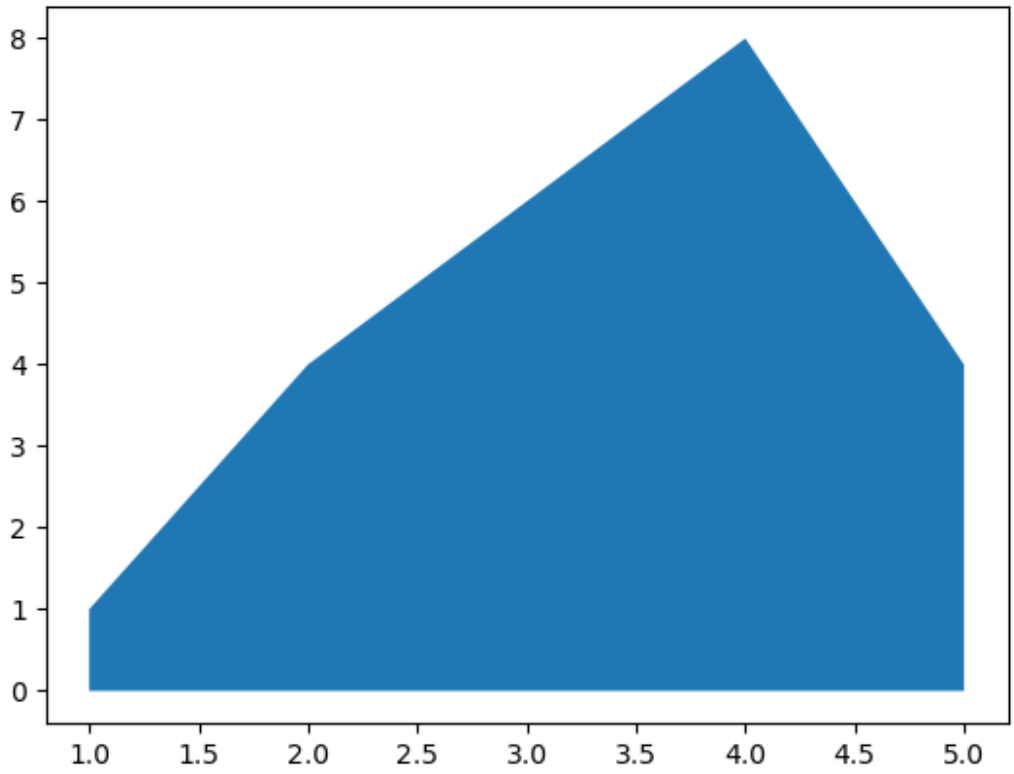


Boxplot

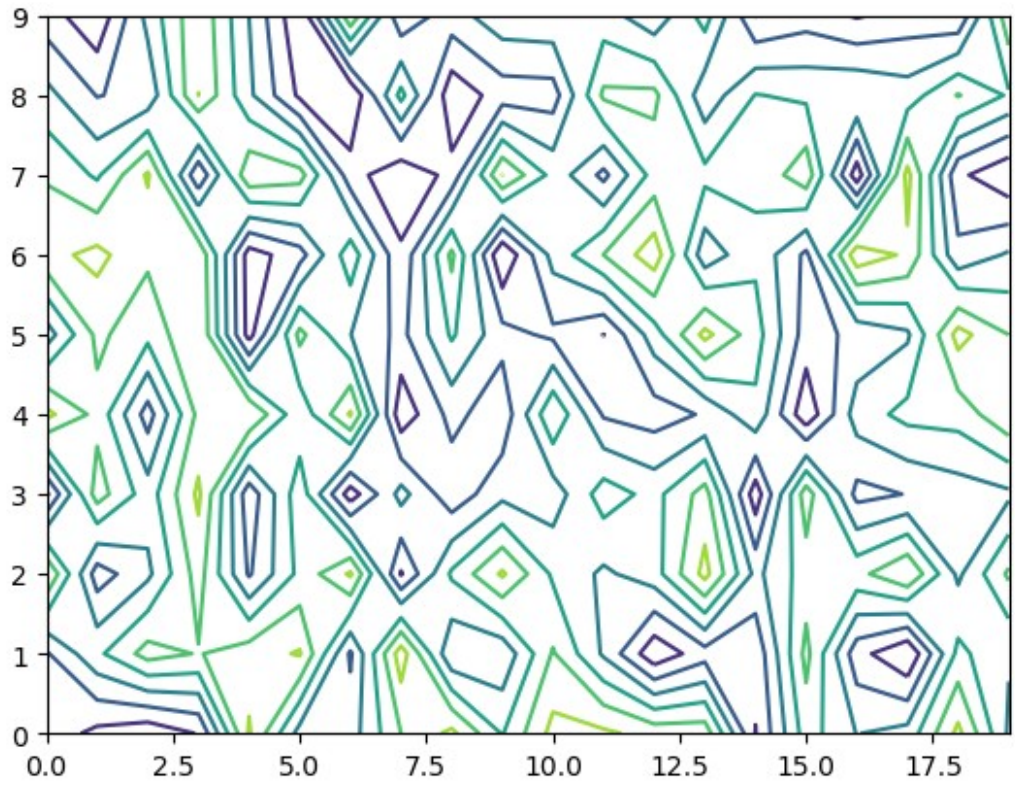
```
data3 = np.random.randn(100)
plt.boxplot(data3)
plt.show();
```




```
Area Chart

x12 = range(1,6)
y12 = [1, 4, 6, 8, 4]
plt.fill_between(x12, y12)
plt.show()
```



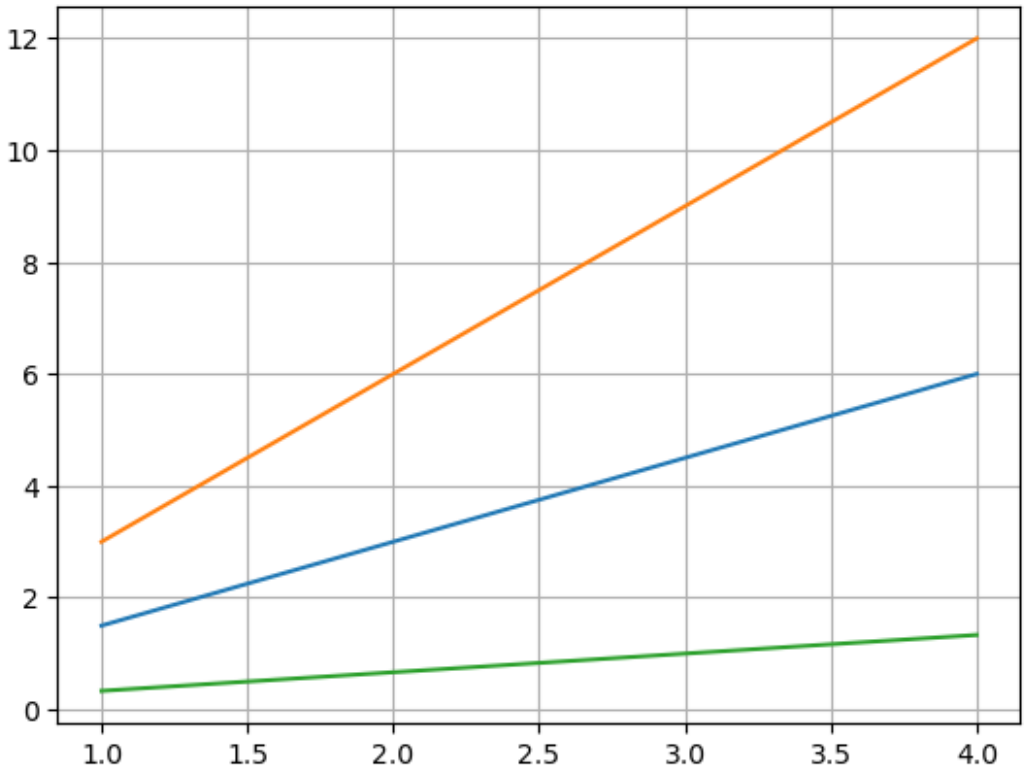

```
Counter plot

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



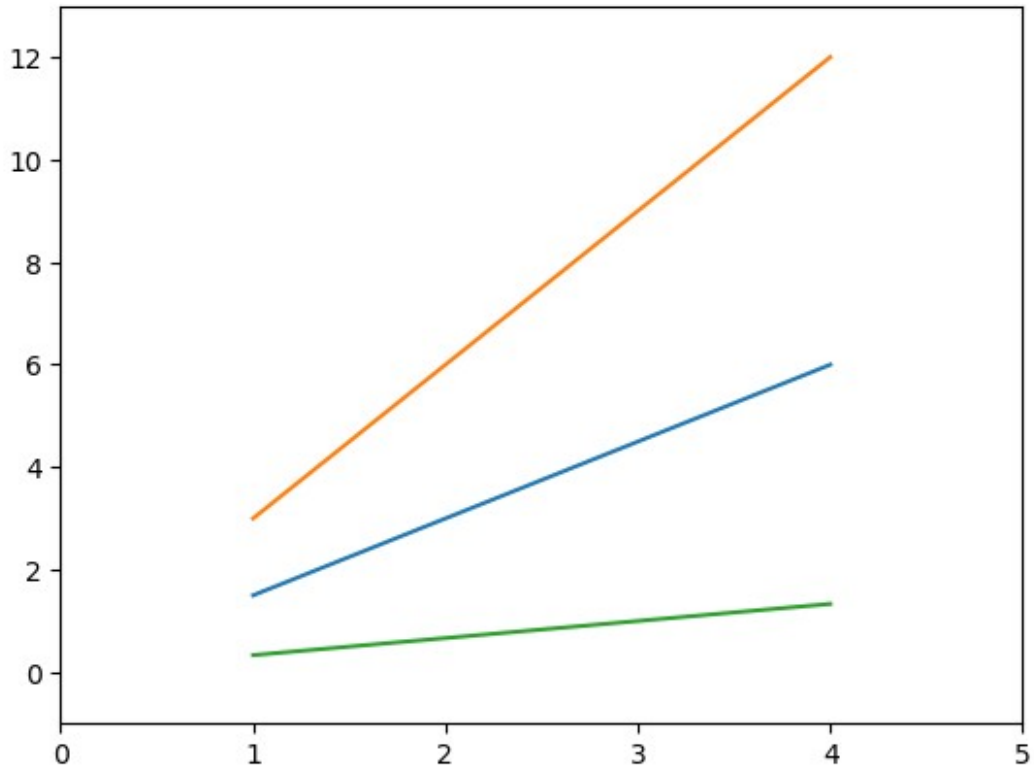

```
Styles with matplotlib plot

print(plt.style.available)
```

```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-
gallery-nogrid', 'bmh', 'classic', 'dark_background', 'fast',
'fivethirtyeight', 'ggplot', 'grayscale', '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']
```

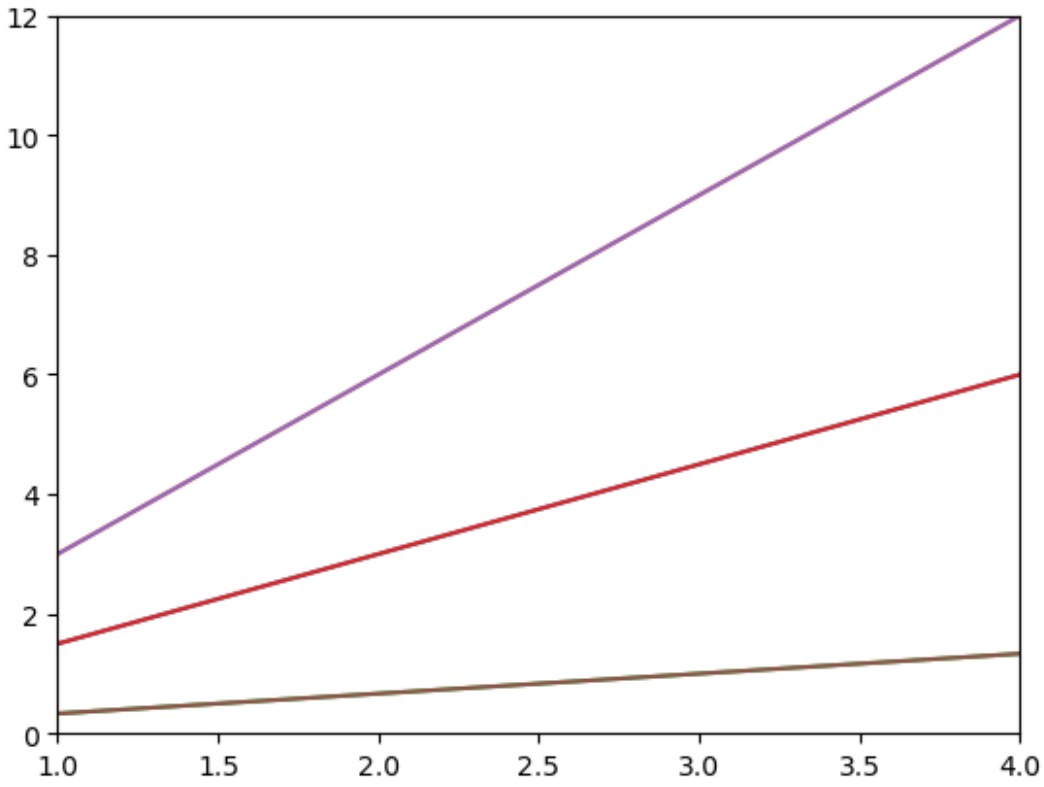
```
Set styles for plots

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

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()
```



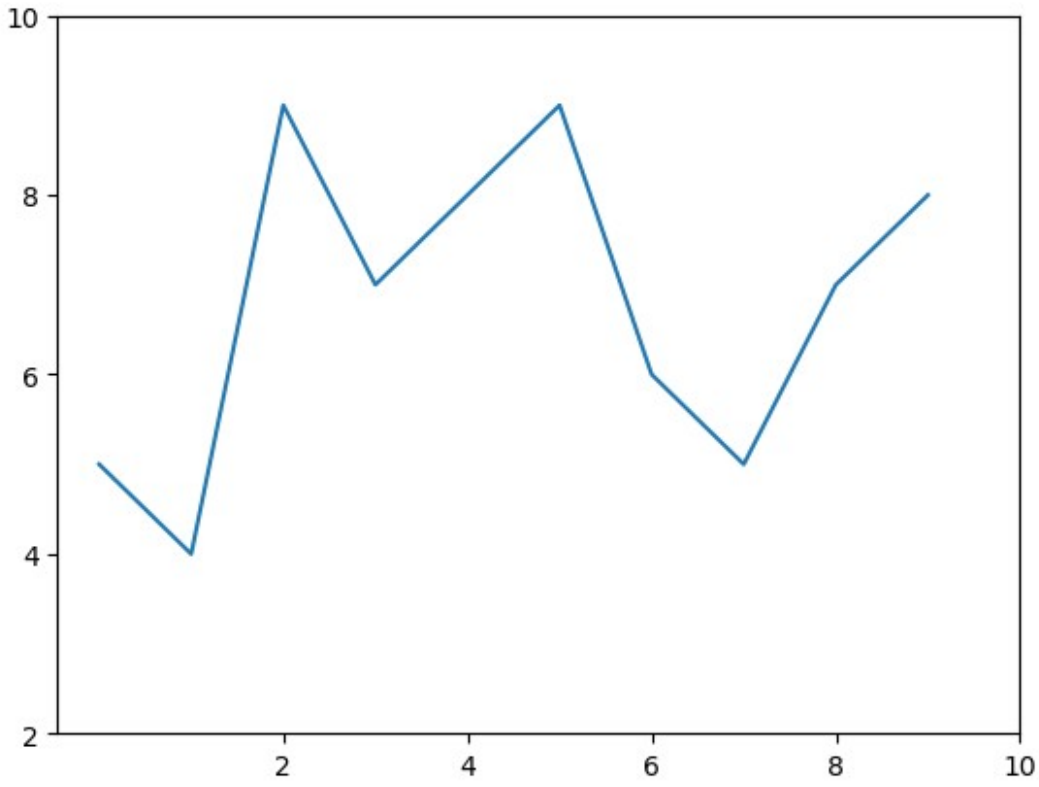

```
Handling axes

x15 = np.arange(1, 5)
plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
plt.axis()
plt.axis([0, 5, -1, 13])
plt.show()
```



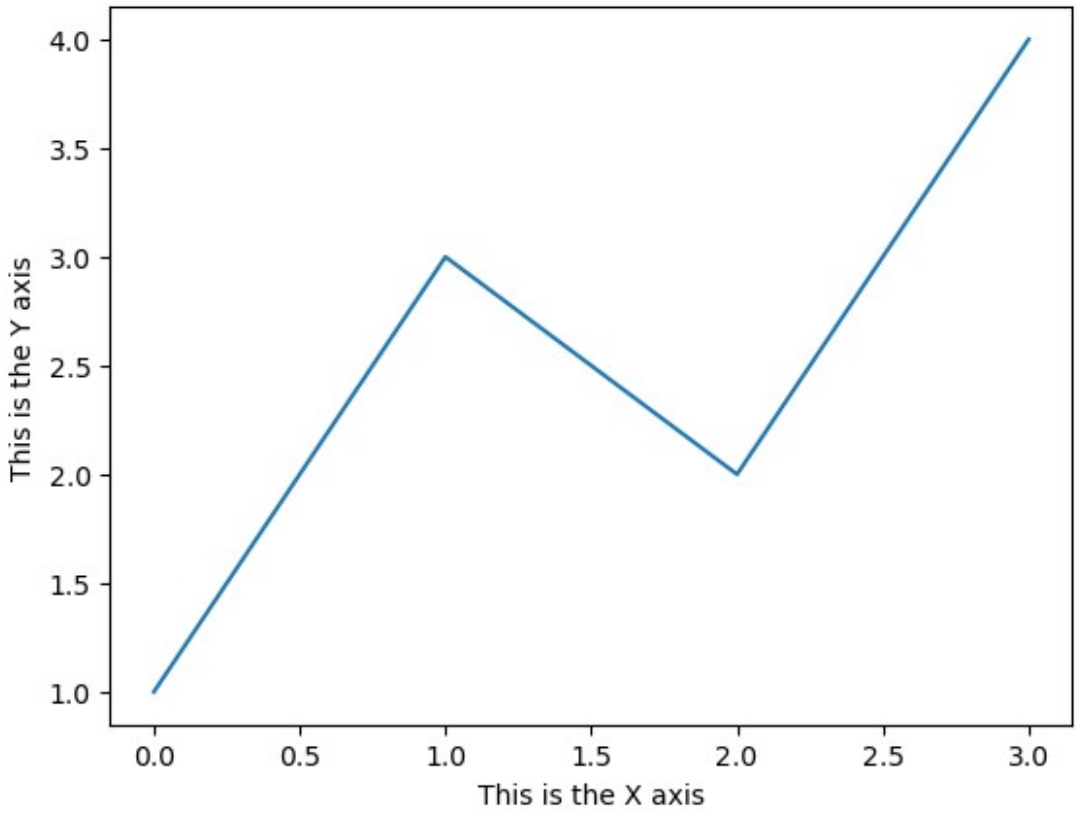

```
x15 = np.arange(1, 5)
plt.plot(x15, x15*1.5, x15, x15*3.0, x15, x15/3.0)
plt.xlim([1.0, 4.0])
plt.ylim([0.0, 12.0])
(0.0, 12.0)
plt.show()
```



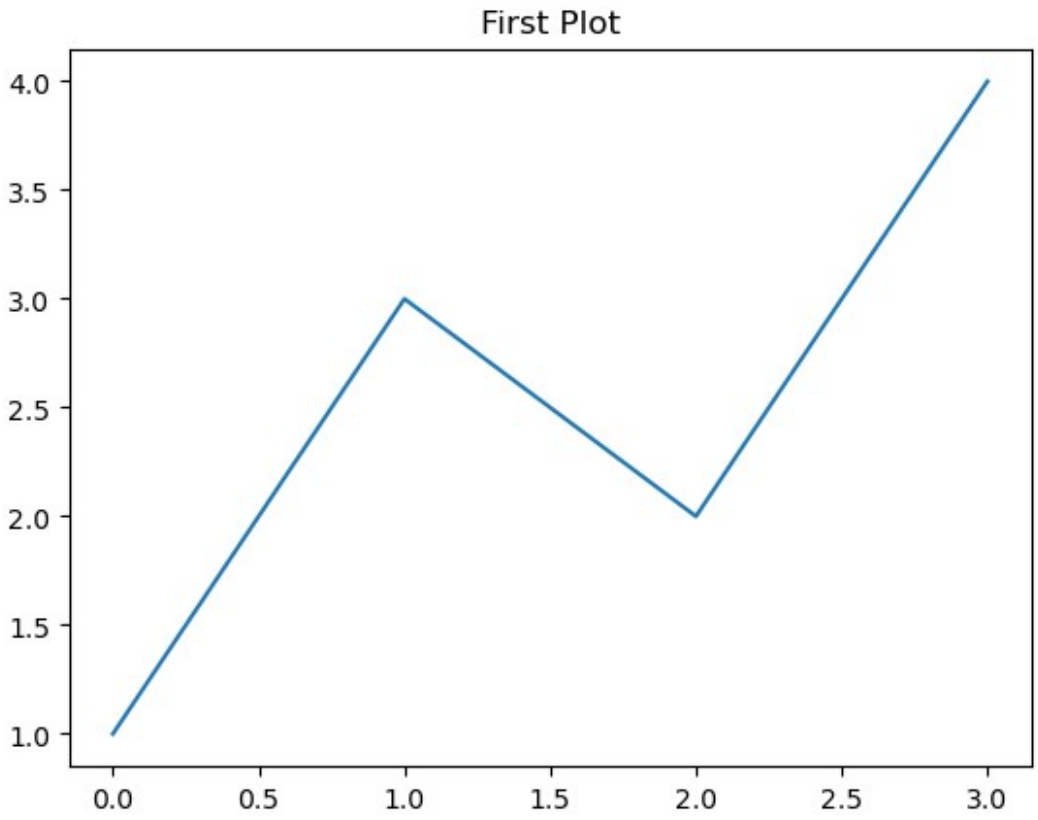

```
u = [5, 4, 9, 7, 8, 9, 6, 5, 7, 8]
plt.plot(u)
plt.xticks([2, 4, 6, 8, 10])
plt.yticks([2, 4, 6, 8, 10])
plt.show()
```



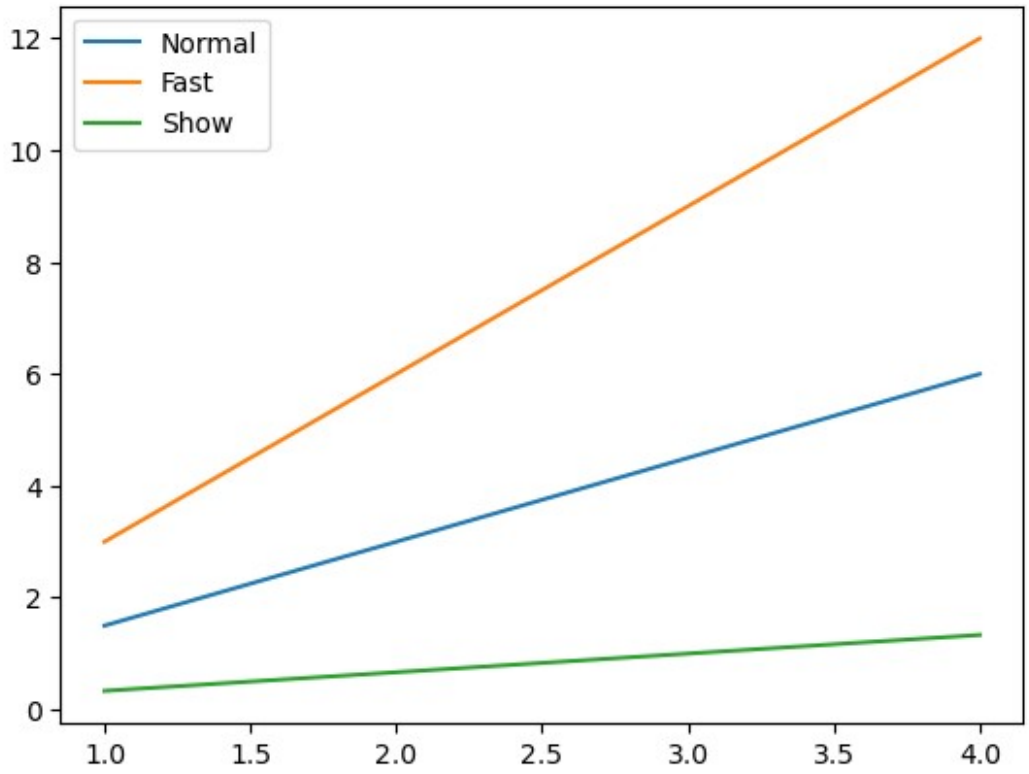

```
Adding labels

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

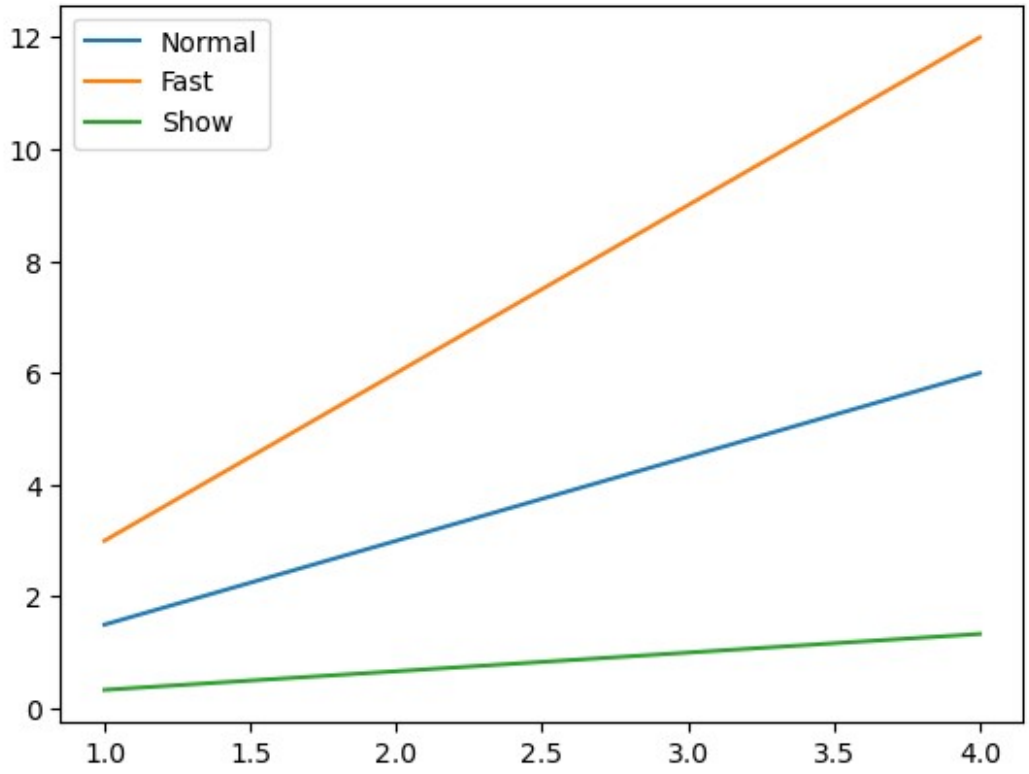



```
Adding a title

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



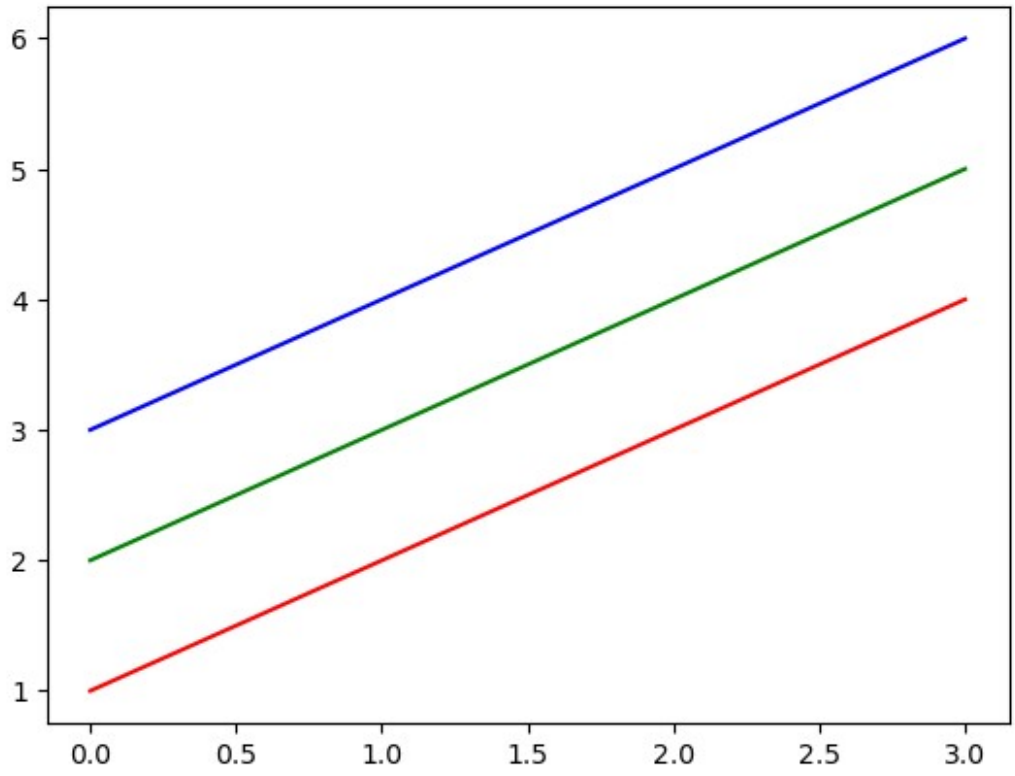

```
Adding a legend

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', 'Show']);
plt.show()
```



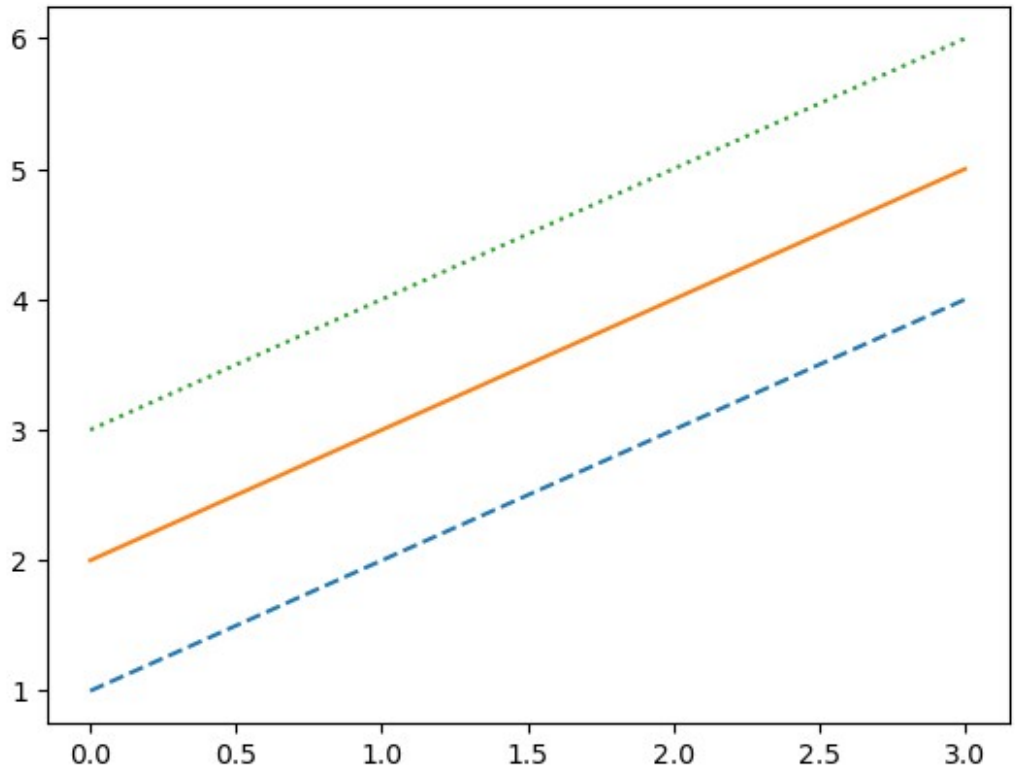



```
Control colour
```

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




```
Control 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