In [1]:

```
# Matplotlib is powerful  package in python  programming langauage for data  visuli
```

In [7]:

```
# Import dependencies
import numpy as np
import pandas as pd
import seaborn as sns
```

In [3]:

```
import matplotlib.pyplot as plt     #pyplot is a interface
```

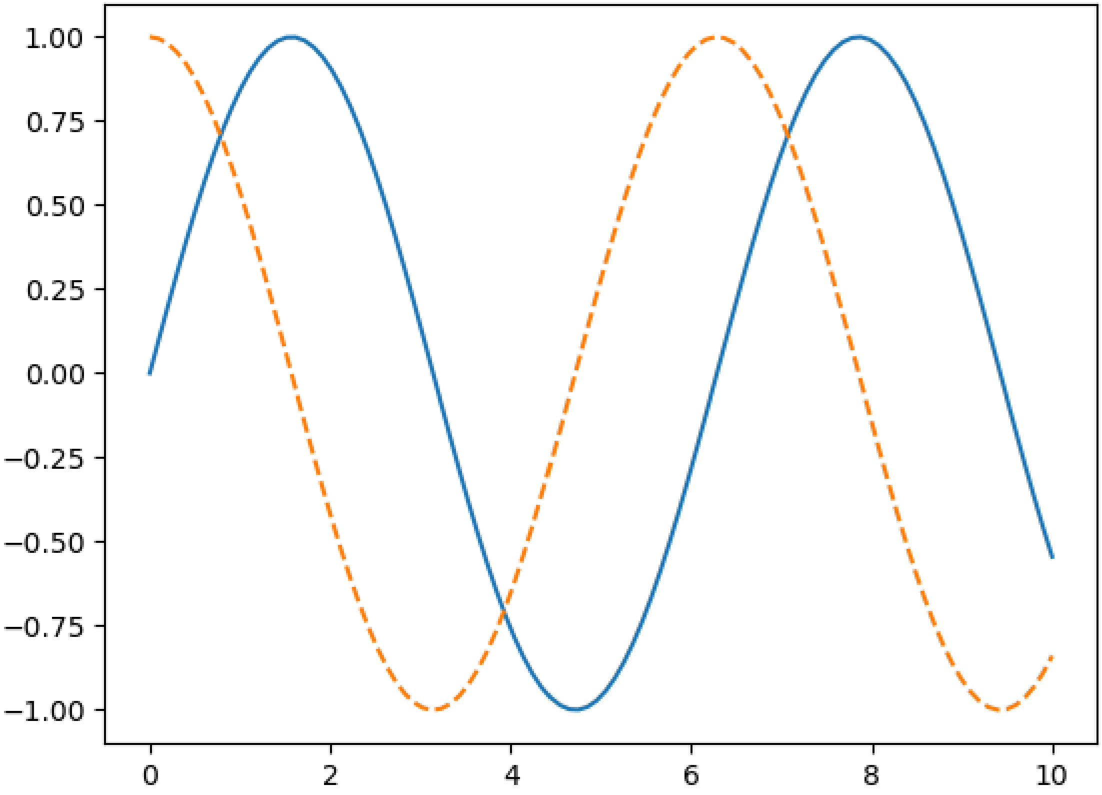
Displaying plots in Matplotlib

In [4]:

```
%matplotlib inline
#It will ouput static image of the plot embeeded in note book
x1=np.linspace(0,10,100)

#create plot figure
fig=plt.figure()

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

Out[4]:

```
[<matplotlib.lines.Line2D at 0x1807b5db750>]
```



In [5]:

```
#matplotlib use pyplot interface and pthon objectoriented interface
```

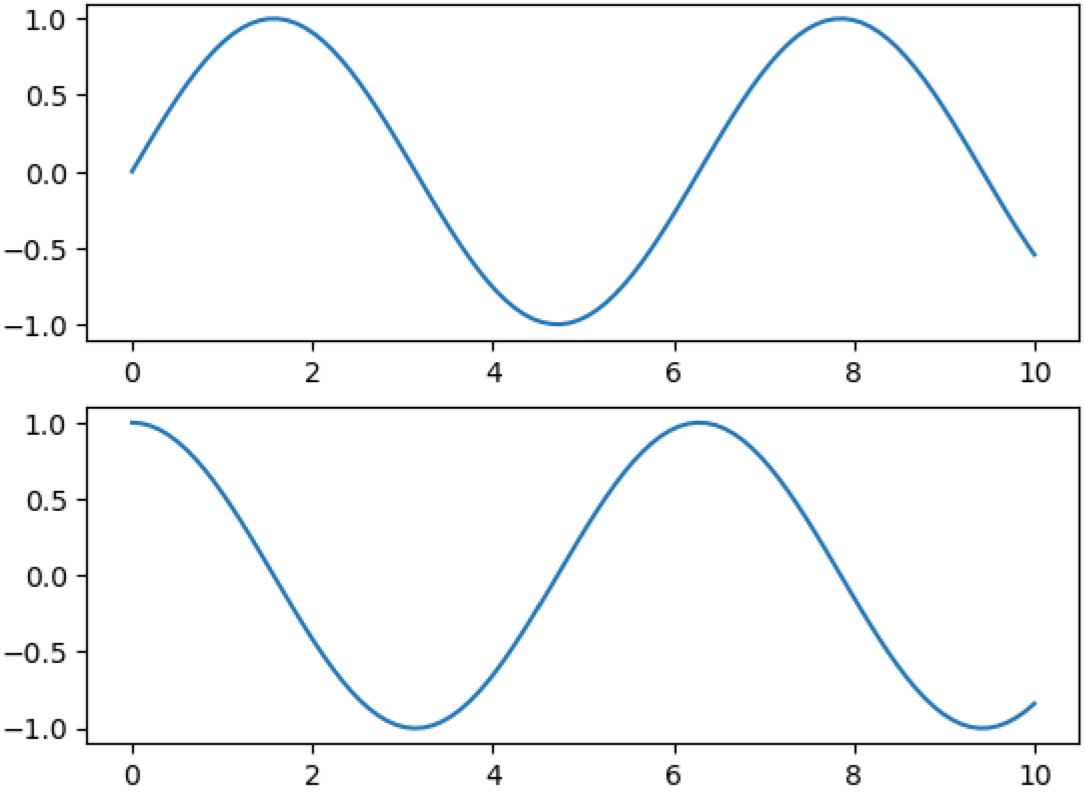
sin& cosine using pyplot API

In [6]:

```
x1=np.linspace(0,10,100)
#plt.gca()                #get current axis
plt.figure()   #create a plotf figure
plt.gcf()     #get current figure
plt.subplot(2,1,1)
plt.plot(x1,np.sin(x1))
plt.subplot(2,1,2)
plt.plot(x1,np.cos(x1))
```

Out[6]:

```
[<matplotlib.lines.Line2D at 0x1807b6dd790>]
```



In [7]:

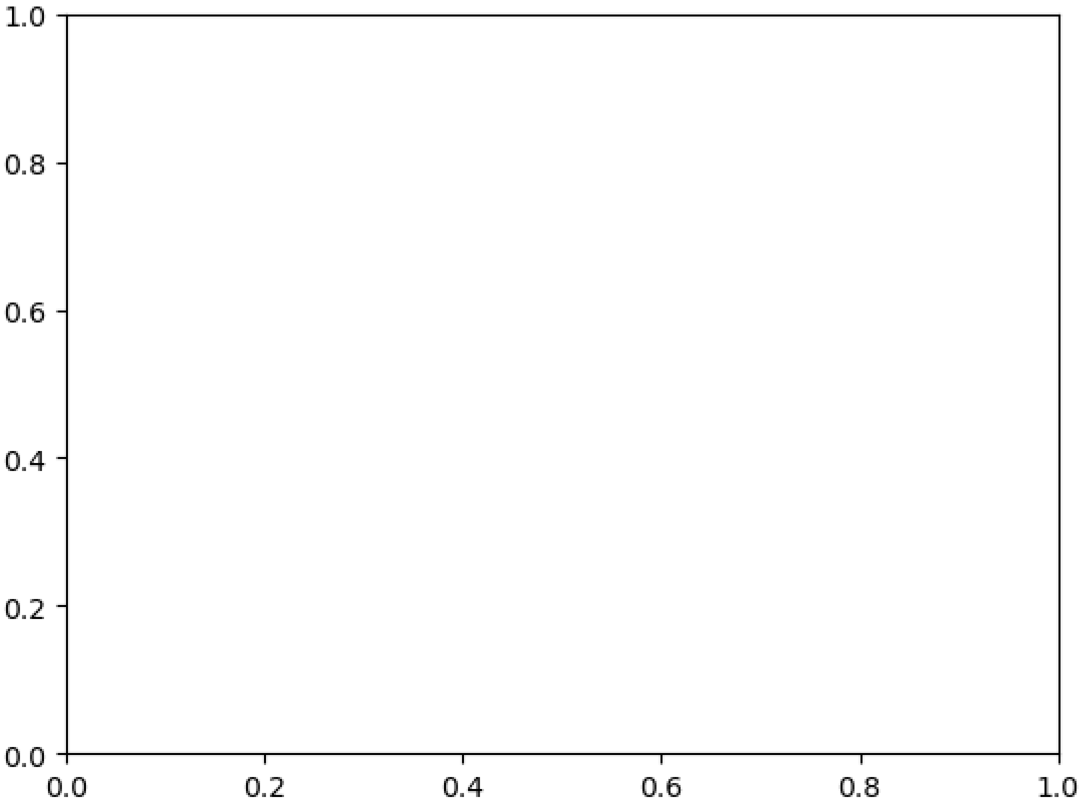
```
#getcurrent figure information
print(plt.gcf())
```

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

In [8]:

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

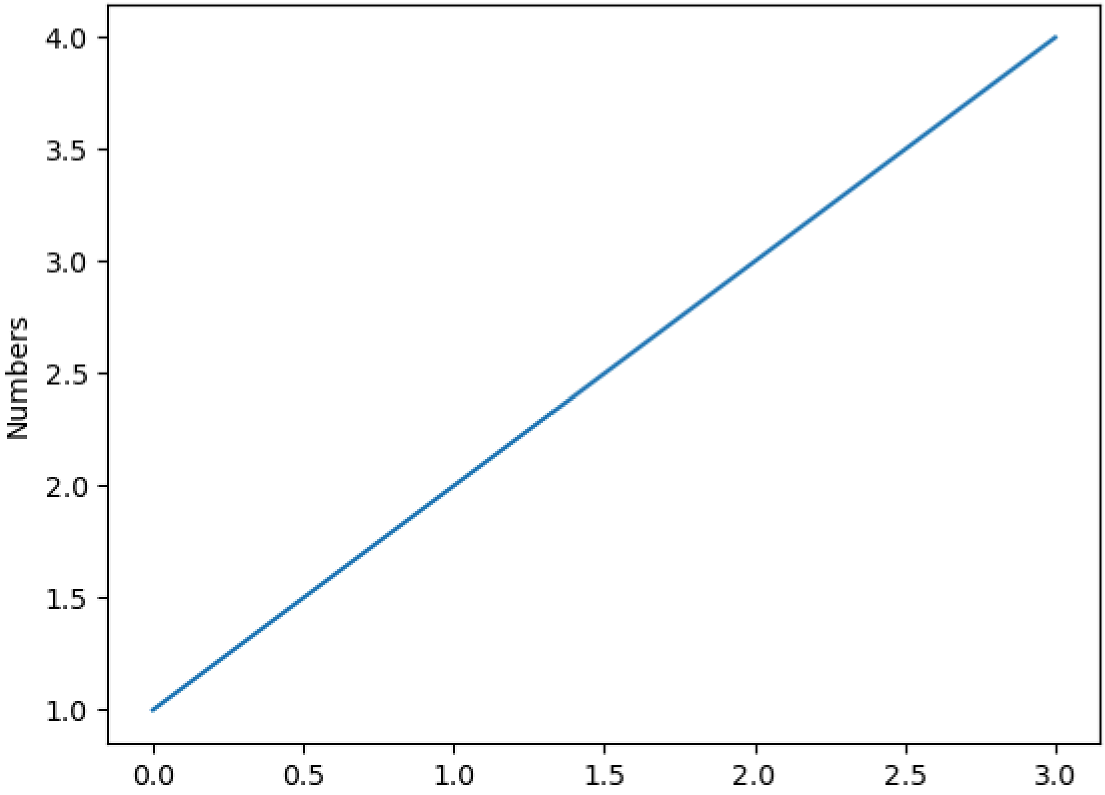
```
Axes(0.125,0.11;0.775x0.77)
```



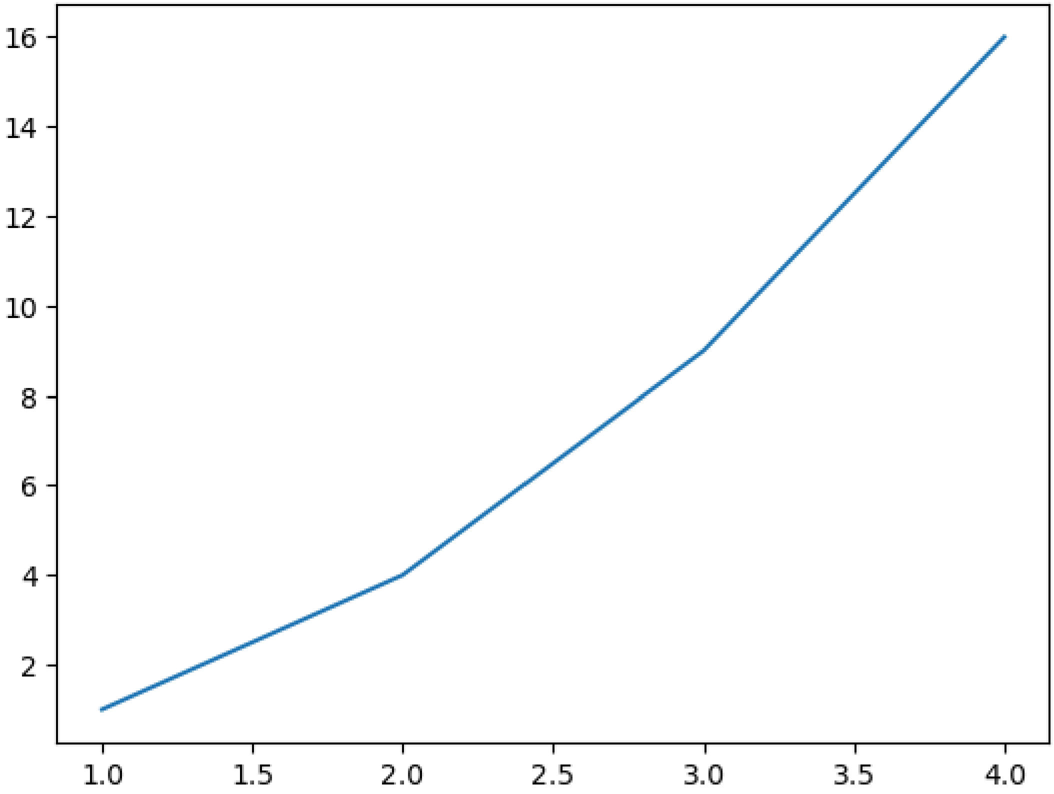
visulaization with pyplot

In [9]:

```
plt.plot([1,2,3,4])  #numbers reperesented on y axis.sequence of y values automatic
plt.ylabel('Numbers')  #default vector lenght of X same as Y x=[0,1,2,3]  y=[1,2,3,
plt.show()                                   # (start with 0)as python indeex start
```



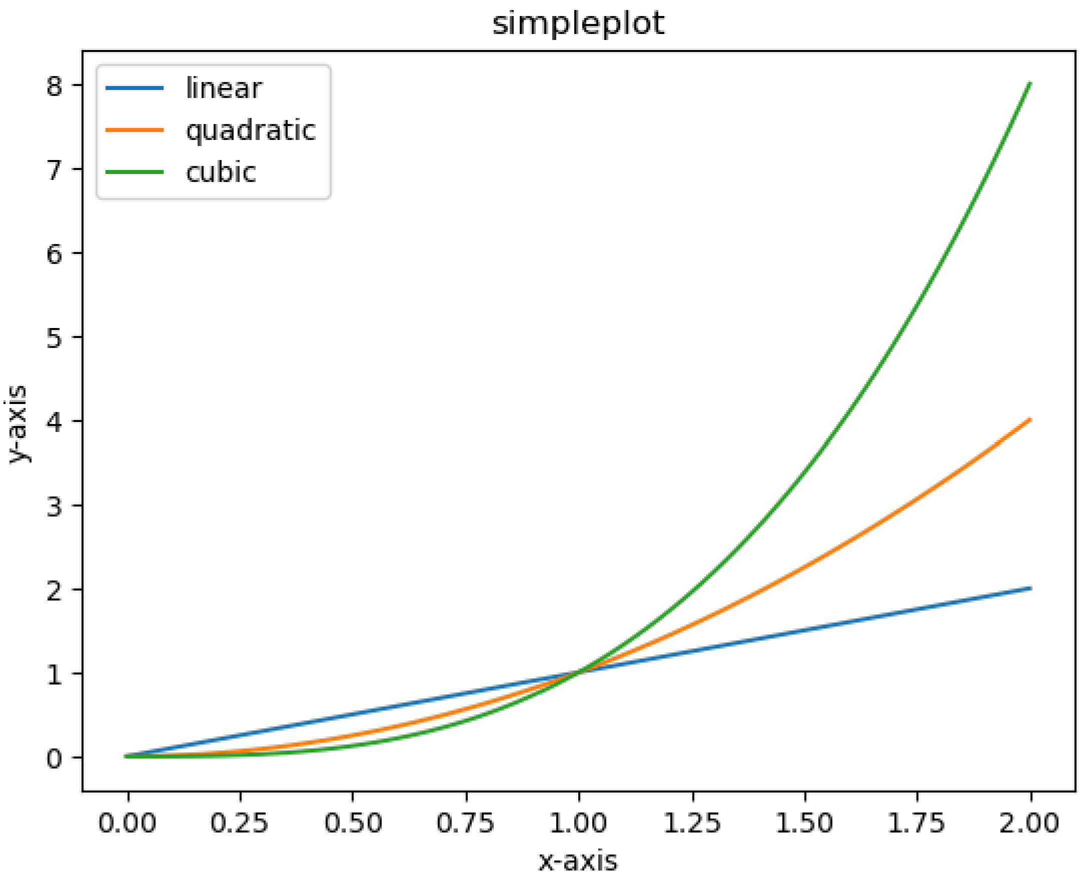
```
In [10]: plt.plot([1,2,3,4],[1,4,9,16])  #plot(x,y)  x=[1,2,3,4]  y=[1,4,9,16]
         plt.show()
```



state machine Interface

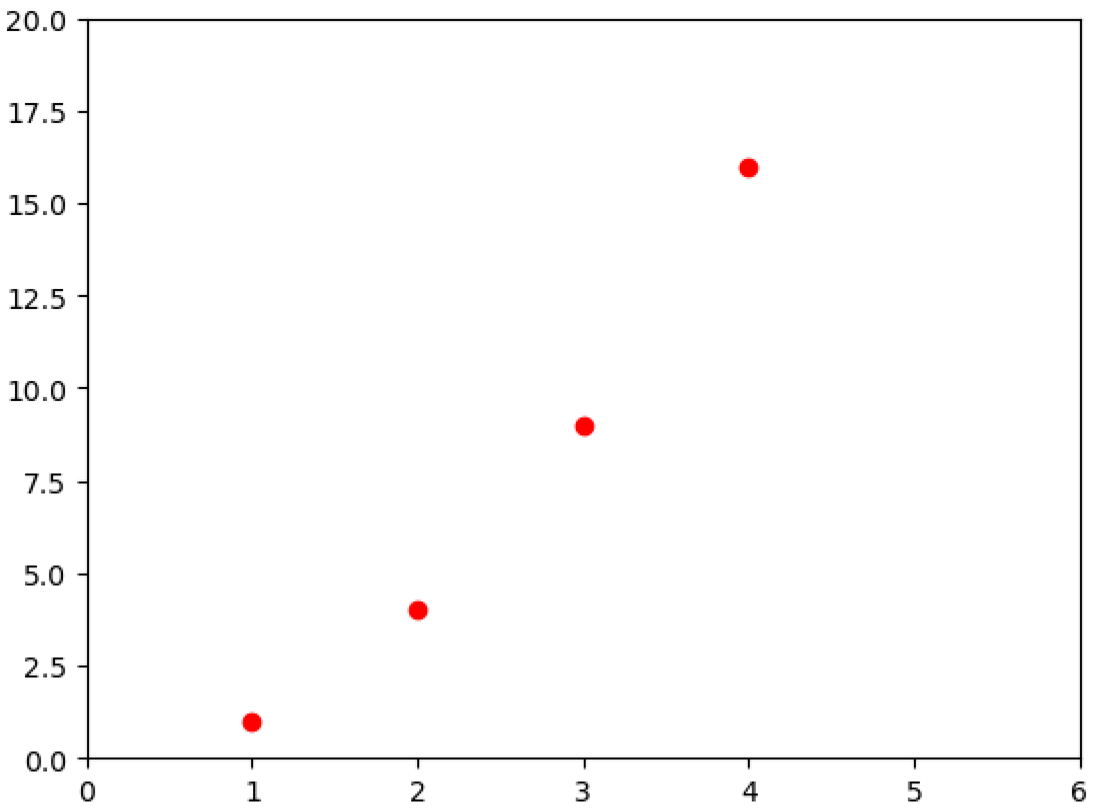
In [11]:
```
#python provides state machine interface for underlying object oriented plotting li
#It automatically creates figures anfd axes  to achieve desired plot
```

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-axis')
plt.ylabel('y-axis')
plt.title('simpleplot')
plt.legend()
plt.show()
```



In [13]:
```
#Formatting style of plot
```

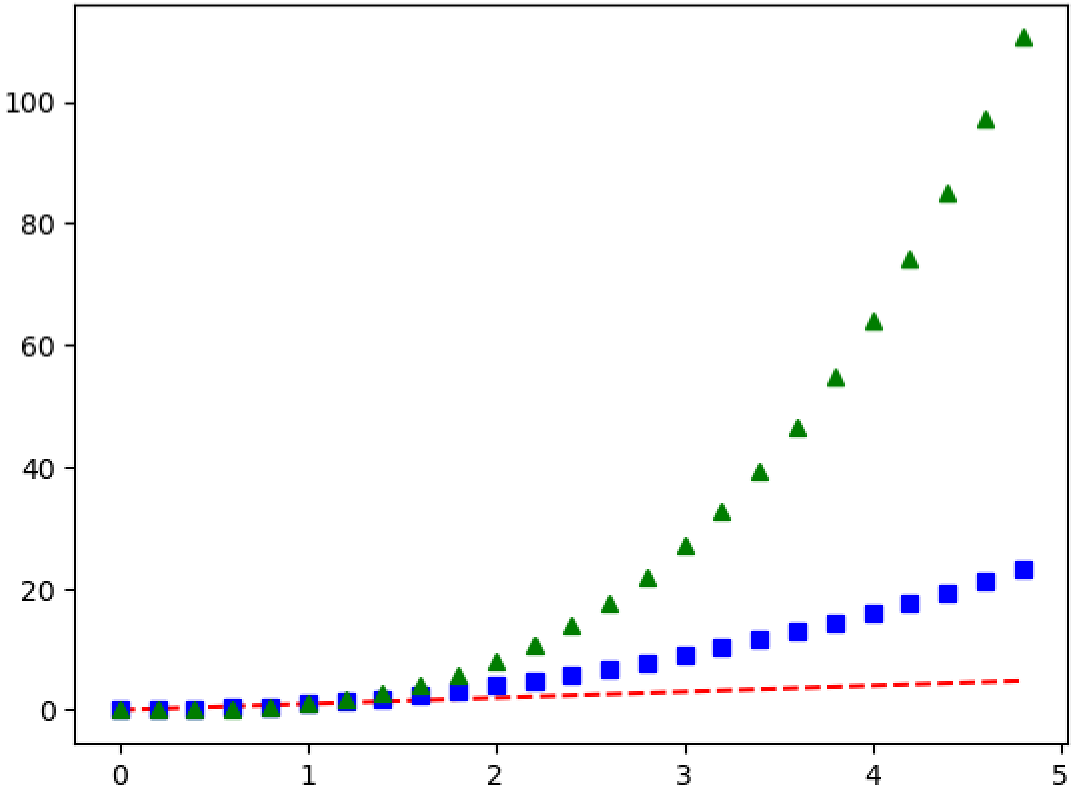
In [14]:
```
plt.plot([1,2,3,4],[1,4,9,16],'ro')
plt.axis([0,6,0,20])   #axis() used to taxke list of [xmin,xmax,ymin,ymax]
plt.show()
```



working with Numpy arrays

In [15]:

```
#evenly sampled time at 200ms interval
t=np.arange(0.0,5.,0.2)
#red dashes,bluesquares,green triangles
plt.plot(t,t,'r--',t,t**2,'bs',t,t**3,'g^')
plt.show()
```



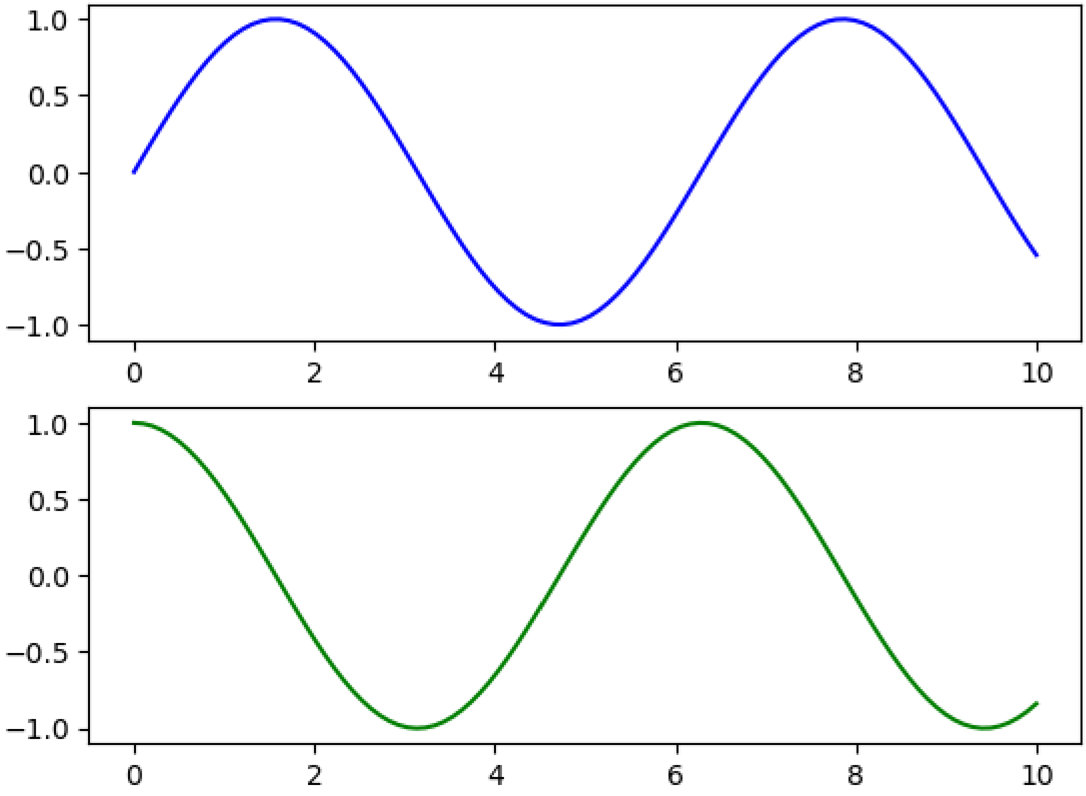
sin &cosine curves using object oriented API

In [16]:

```
#first create grid of plots
#ax will be array of two Axes objects
fig,ax=plt.subplots(2)

#call plot() method on appropriate object
ax[0].plot(x1,np.sin(x1),'b-')
ax[1].plot(x1,np.cos(x1),'g-')
```

Out[16]:

```
[<matplotlib.lines.Line2D at 0x1807bf78110>]
```



In [17]:
```
#objects and references
#create the reference to figure instance in fig variable.then create new instance a
```

In [18]:
```
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[18]:
```
Text(0.5, 1.0, 'title')
```

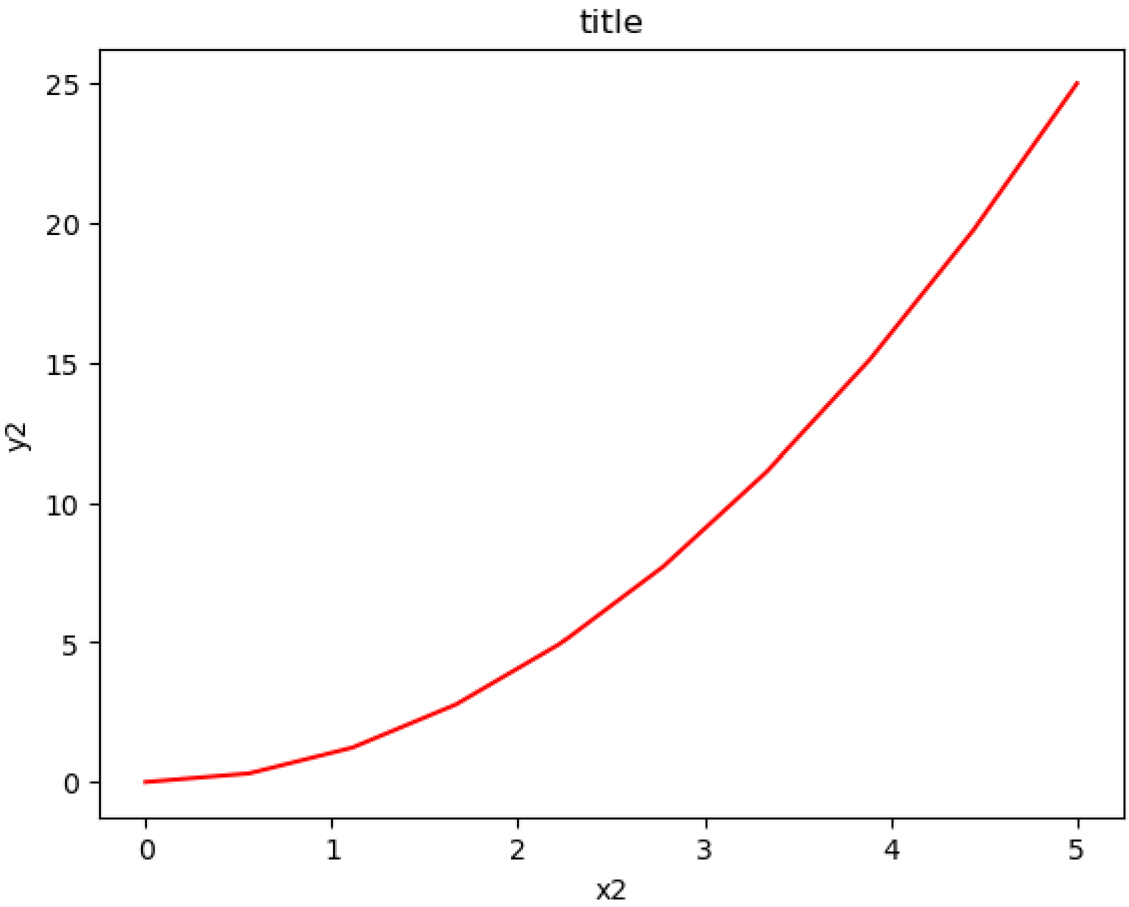


Figure and Axes

In [19]:

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

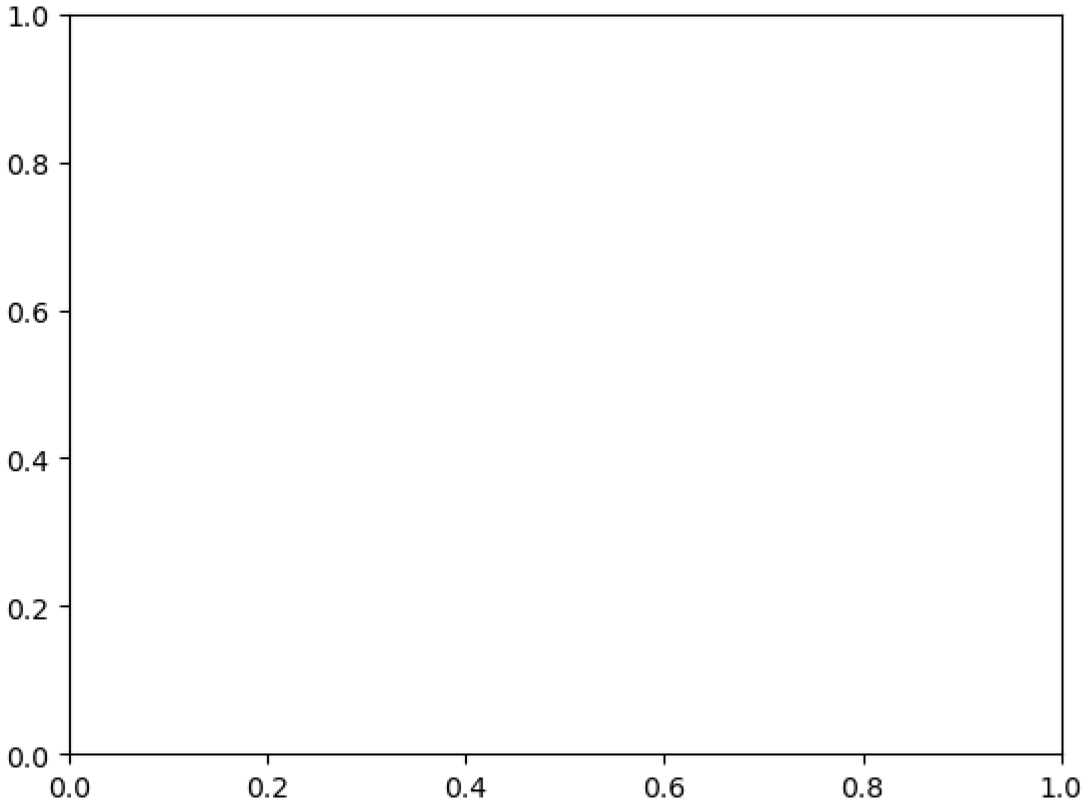
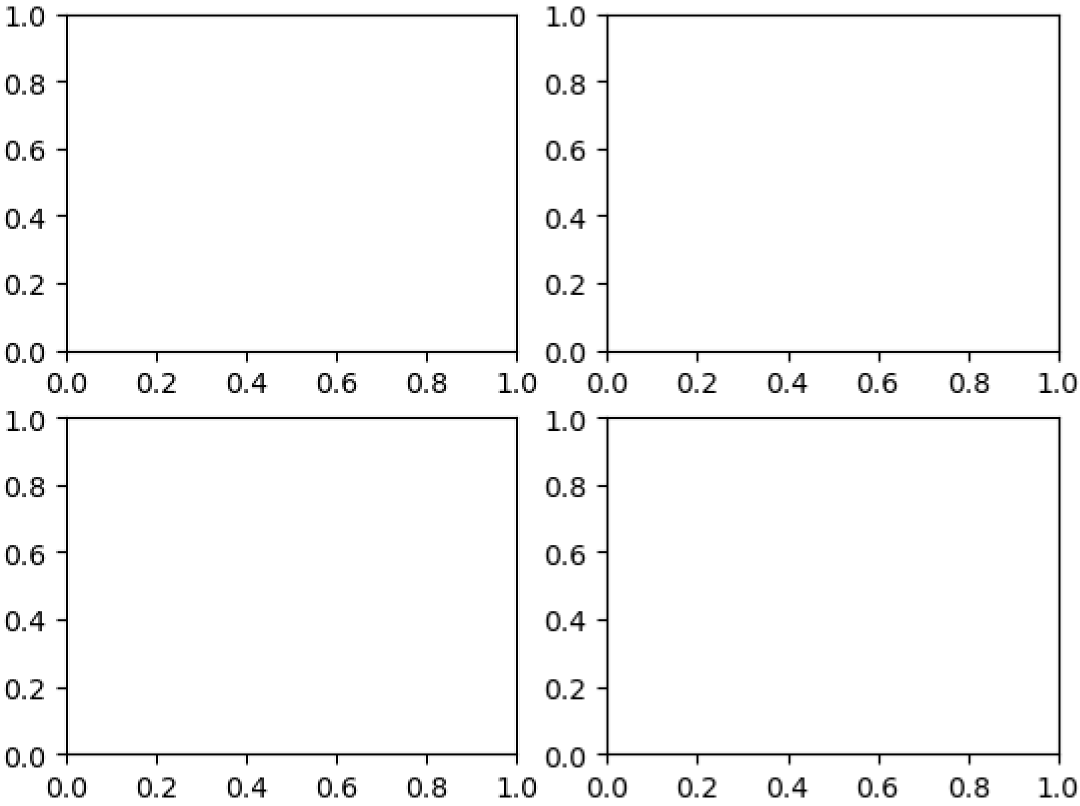


Figure and subplots

In [20]:

```
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)
```

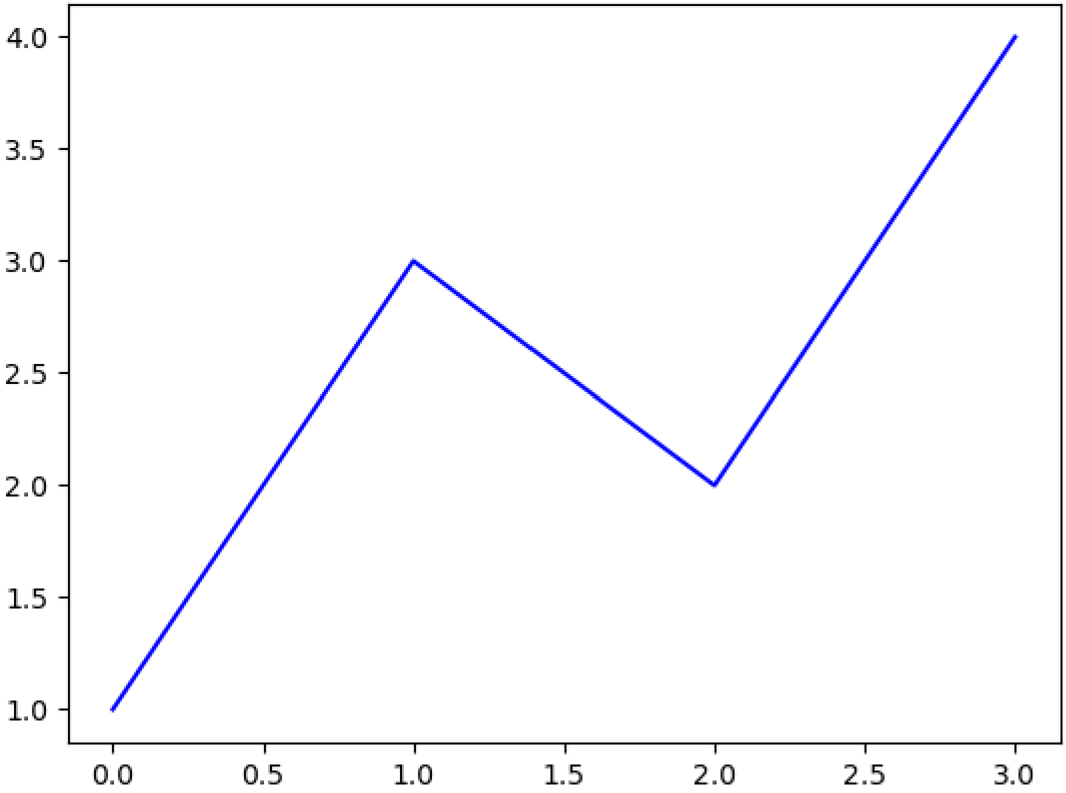


In [21]:

```
#First plot with matplotlib
```

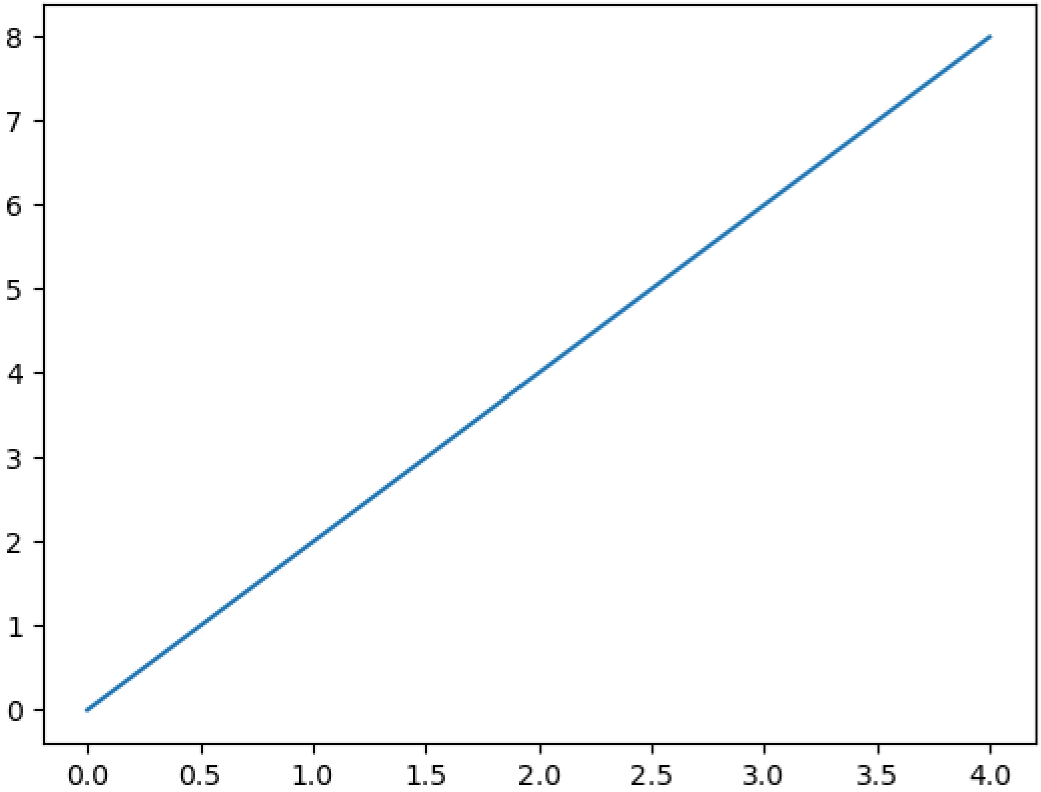
In [22]:

```
plt.plot([1,3,2,4],'-b')  #here  y axis [1 2 3 4] x axis [0 1 2 3]
plt.show()
```



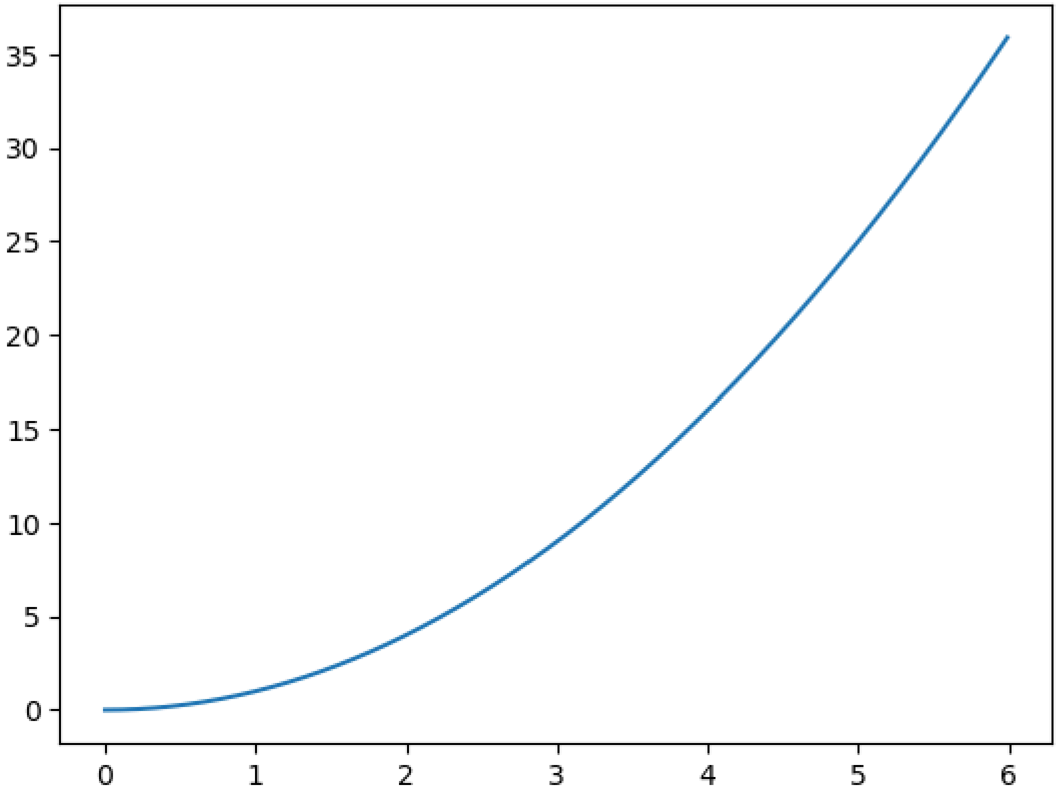
In [23]:

```
x3=range(5)
plt.plot(x3,[xi*2 for xi in x3])
plt.show()
```



In [24]:

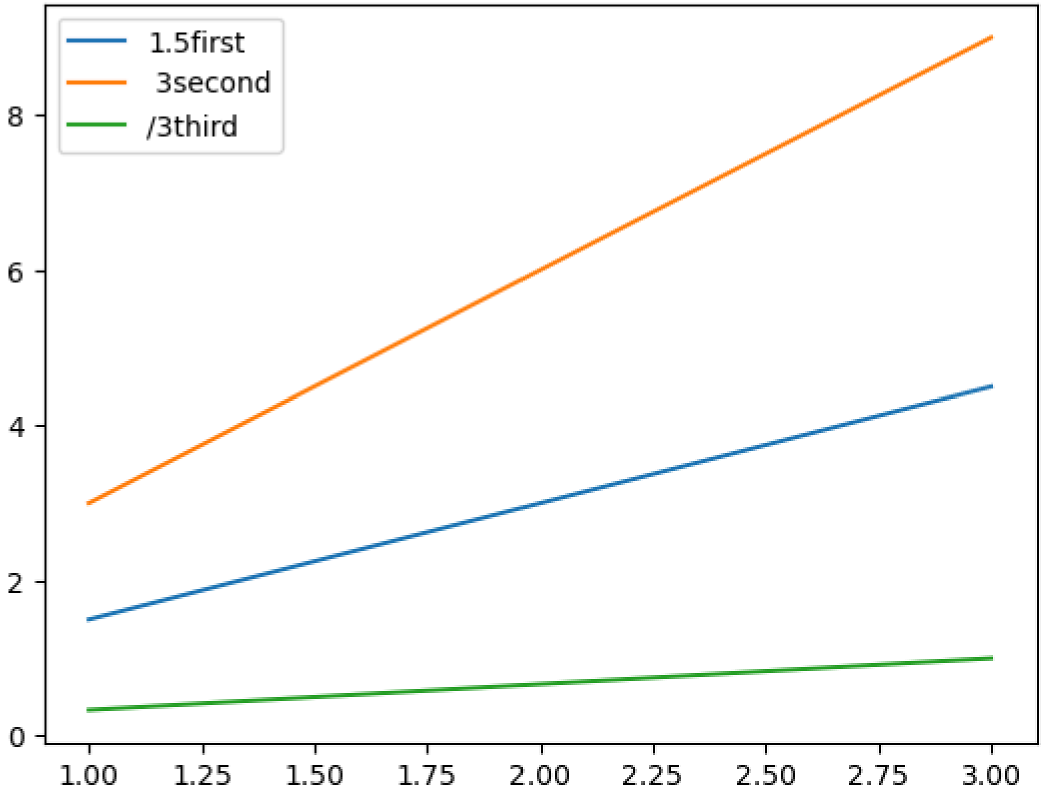
```
x3=np.arange(0.0,6.0,0.01)
plt.plot(x3,[xi**2 for xi in x3])
plt.show()
```



Multiline plots

In [25]:

```
x4=range(1,4)
plt.plot(x4,[xi*1.5 for xi in x4],label='1.5first')
plt.plot(x4,[xi*3  for xi in x4],label=' 3second')
plt.plot(x4,[xi/3.0 for xi in x4],label='/3third')
plt.legend()
plt.show()
```



Saving the plot

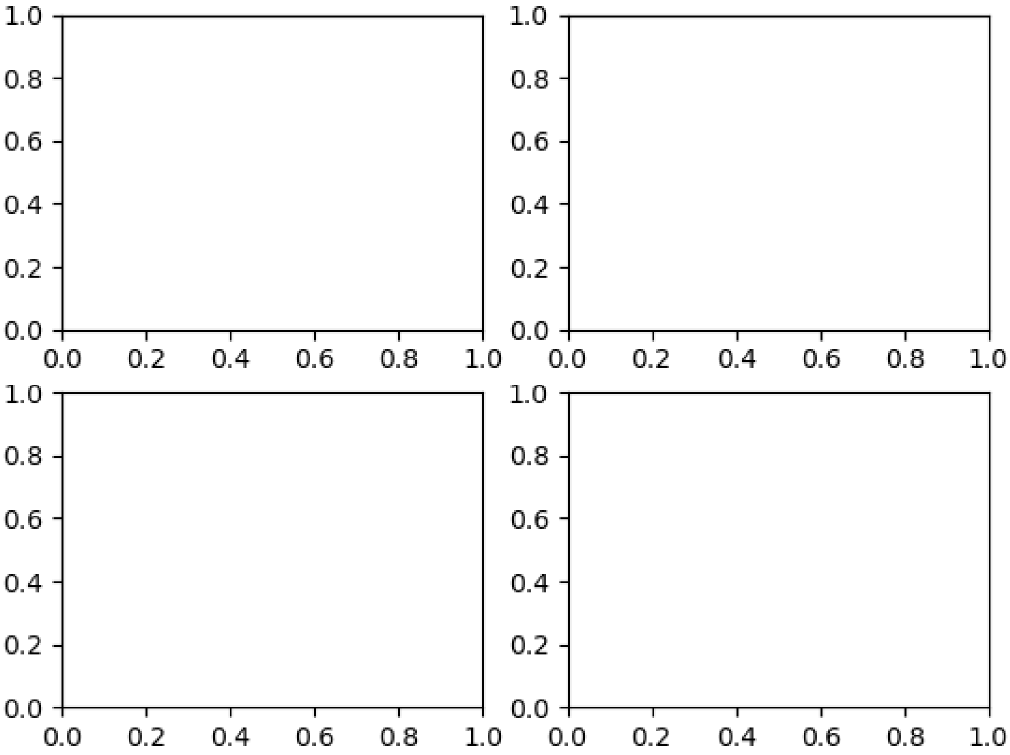
In [26]:

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

In [27]:

```
#explorre the contents of figure
from IPython.display import Image
Image('plot1.png')
```

Out[27]:



In [28]:
```
fig.canvas.get_supported_filetypes()
```

Out[28]:
```
{'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'}
```

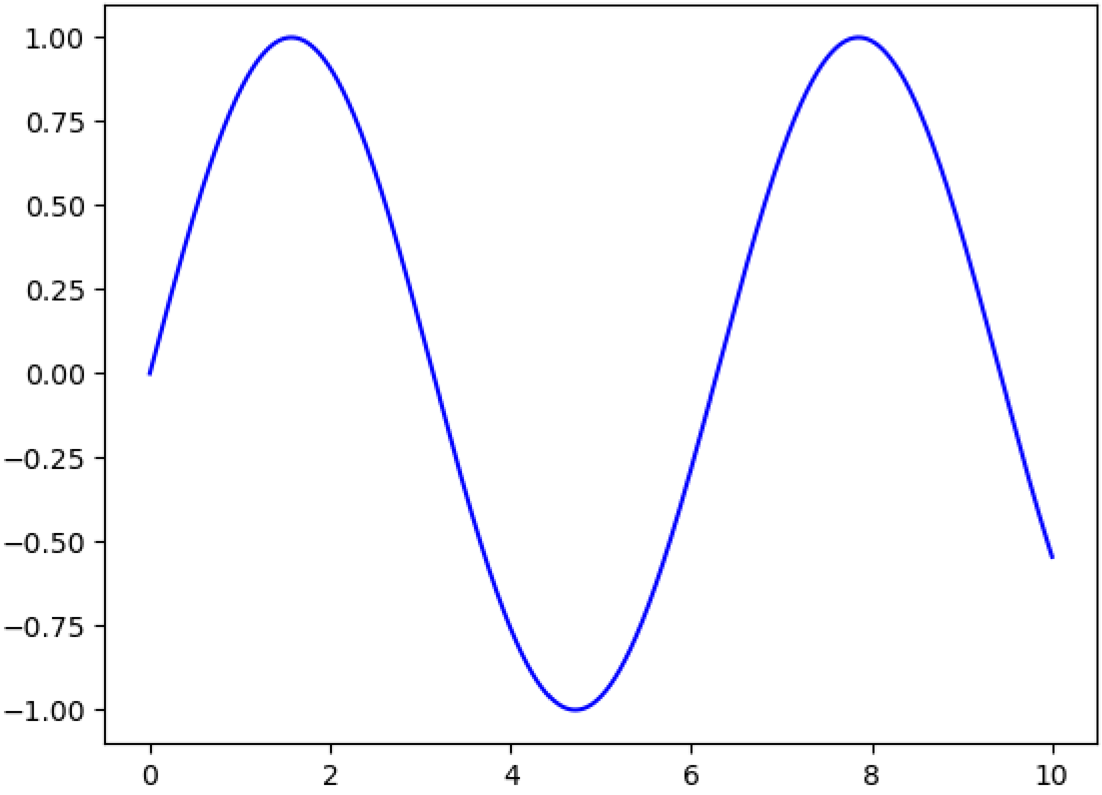
Line plots

In [29]:
```
#create figure and axes first
fig=plt.figure()
ax=plt.axes()
#Declare variable x5
x5=np.linspace(0,10,1000)
#plot ths sinusoid function
```

```
ax.plot(x5,np.sin(x5),'b-')
#plt.plot(x5,np.sin(x5),'b-')
```

Out[29]:

```
[<matplotlib.lines.Line2D at 0x1807d4cf0d0>]
```



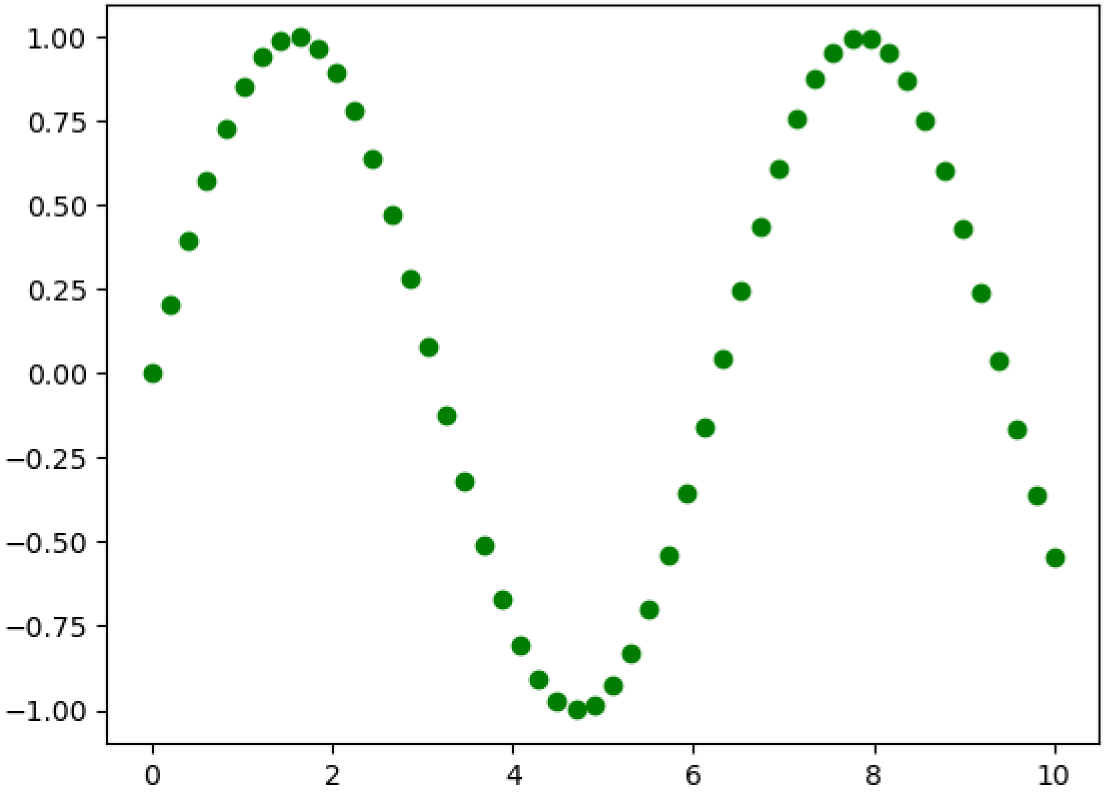
Scatter plots

In [30]:

```
x7=np.linspace(0,10,50)
y7=np.sin(x7)
plt.plot(x7,y7,'o',color='g')
```

Out[30]:

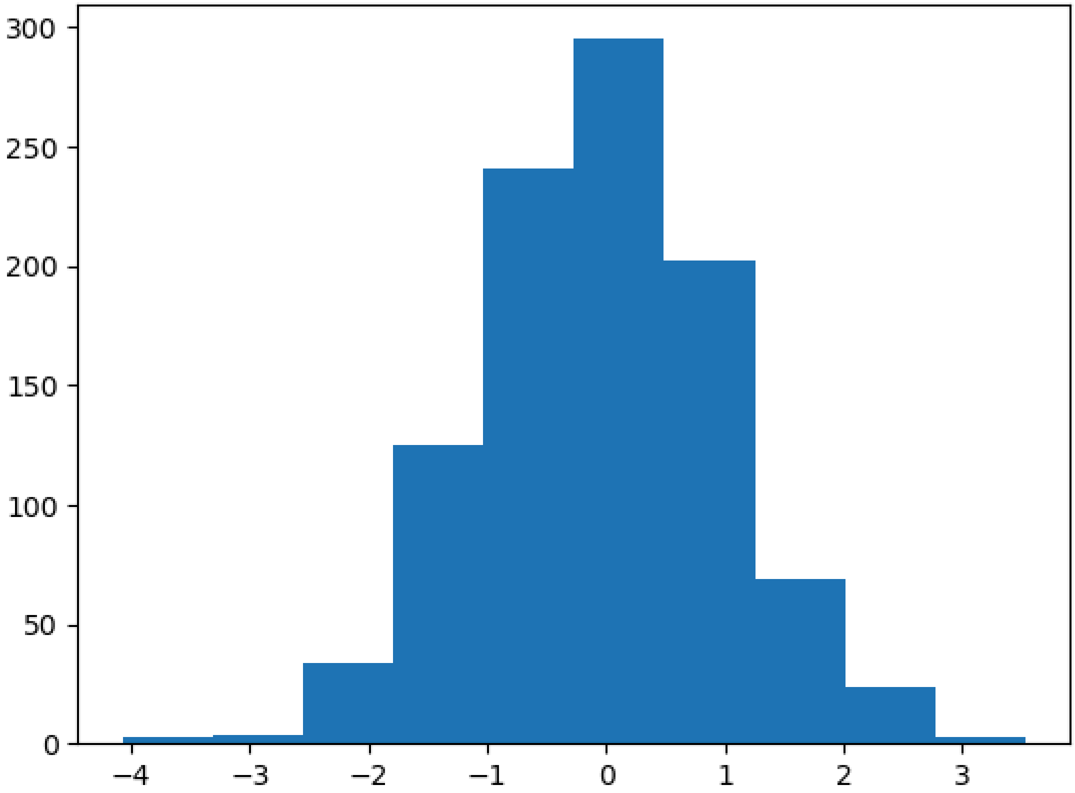
```
[<matplotlib.lines.Line2D at 0x1807d44bd90>]
```



Histogram

```
In [31]: data1=np.random.randn(1000)
         plt.hist(data1)
```

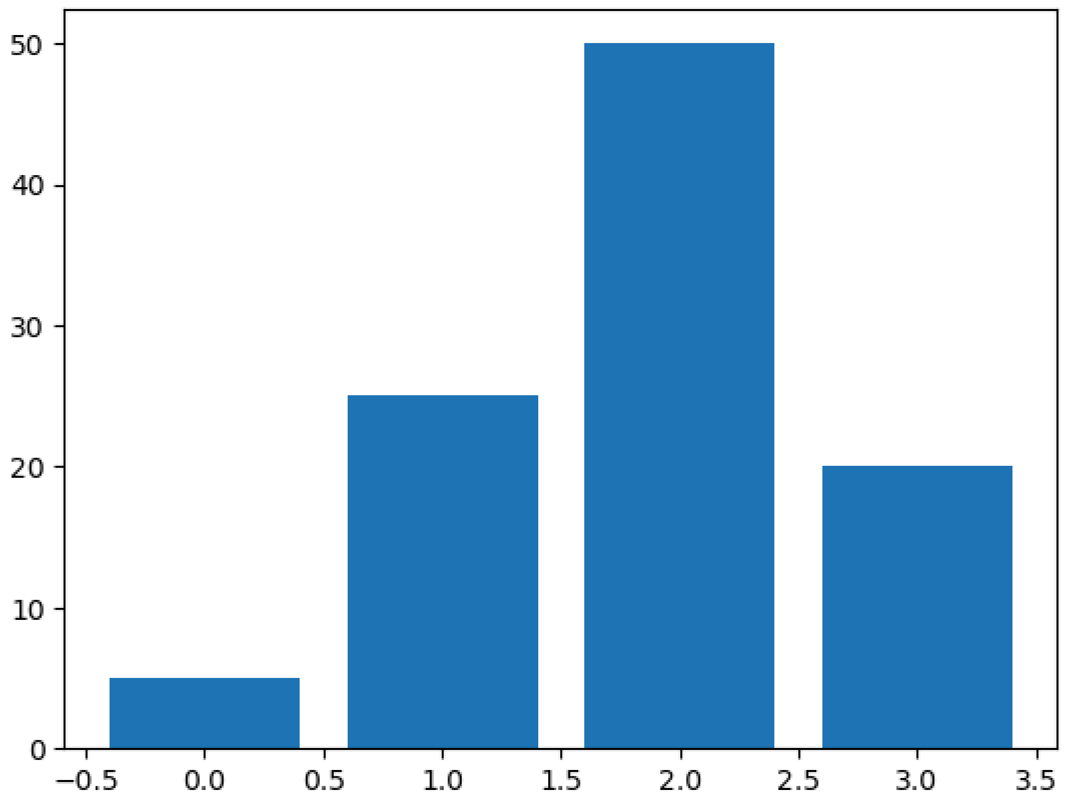
```
Out[31]: (array([  3.,   4.,  34., 125., 241., 295., 202.,  69.,  24.,   3.]),
          array([-4.07232168, -3.31192543, -2.55152917, -1.79113291, -1.03073665,
                 -0.2703404 ,  0.49005586,  1.25045212,  2.01084838,  2.77124463,
                  3.53164089]),
          <BarContainer object of 10 artists>)
```



Bar chart

```
In [32]: data2=[5.0,25.0,50.0,20.0]
         plt.bar(range(len(data2)),data2)
         p1t.show()
```

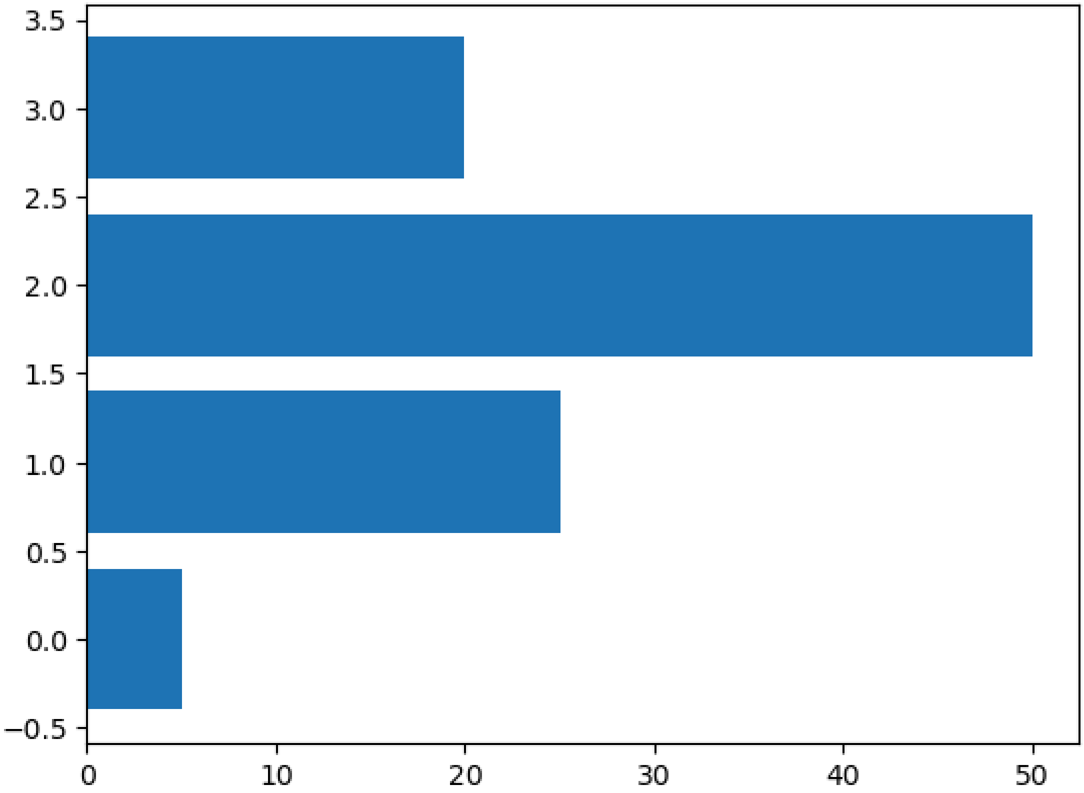
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[32], line 3
      1 data2=[5.0,25.0,50.0,20.0]
      2 plt.bar(range(len(data2)),data2)
----> 3 p1t.show()

NameError: name 'p1t' is not defined
```



In [49]:

```
#Horizontal bar chart
data2=[5.0,25.0,50.0,20.0]
plt.barh(range(len(data2)),data2)
p1t.show()
```

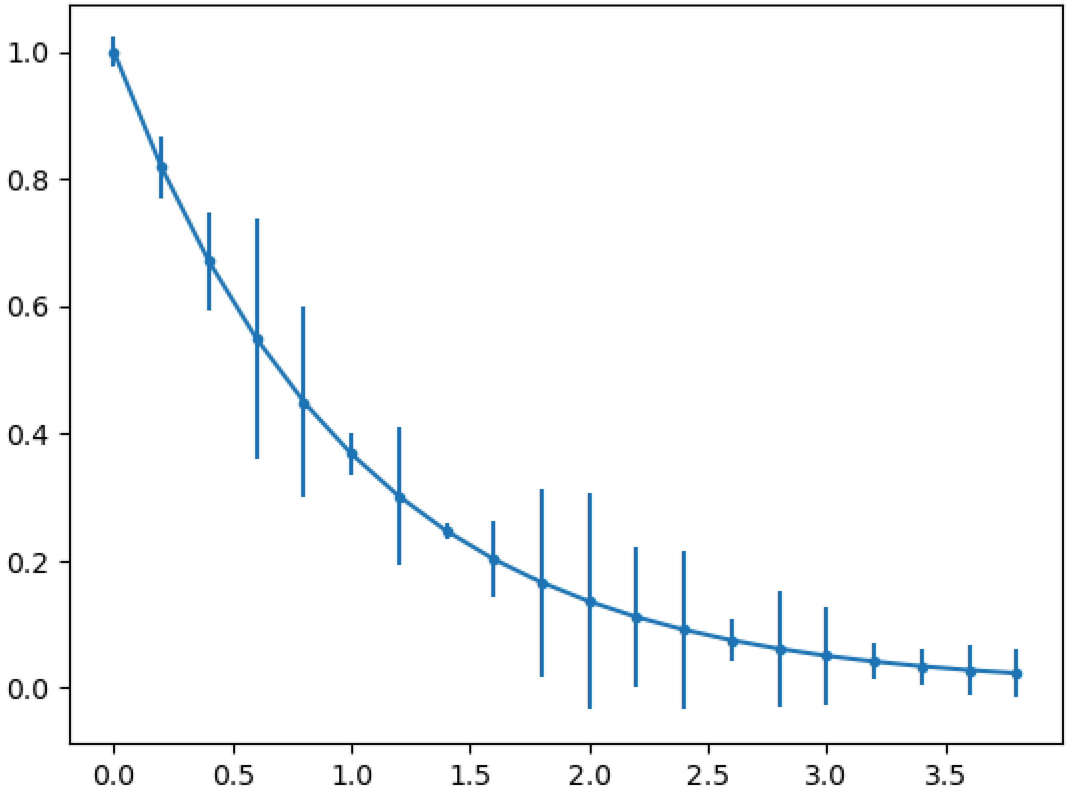
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[49], line 4
      2 data2=[5.0,25.0,50.0,20.0]
      3 plt.barh(range(len(data2)),data2)
----> 4 p1t.show()

NameError: name 'p1t' is not defined
```



Error bar chart

In [35]:

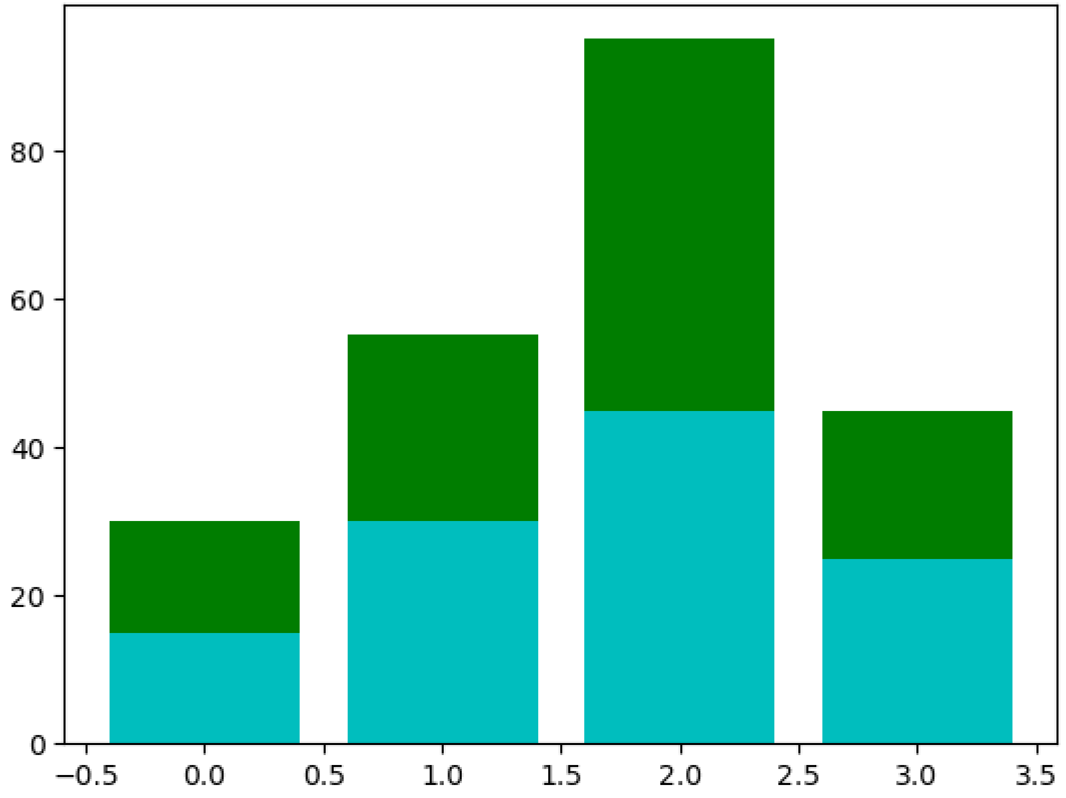
```
#Error bar chart
x9=np.arange(0,4,0.2)
y9=np.exp(-x9)
e1=0.1*np.abs(np.random.randn(len(y9)))
plt.errorbar(x9,y9,yerr=e1,fmt='.-')
plt.show()
```



Stacked bar chart

In [42]:

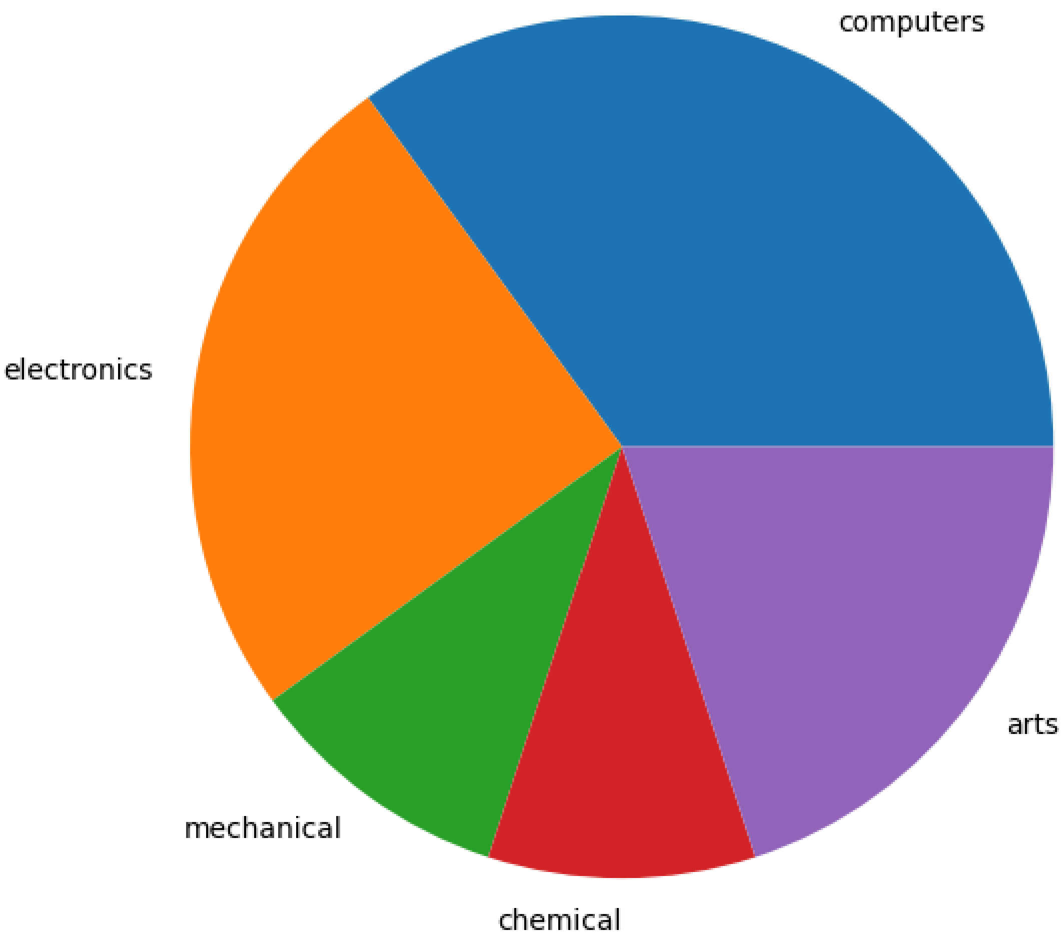
```
A=[15.0,30.0,45.0,25.0]
B=[15.0,25.0,50.0,20.0]
z2=range(4)
plt.bar(z2,A,color='c')
plt.bar(z2,B,color='g',bottom=A)
plt.show()
```



pie chart

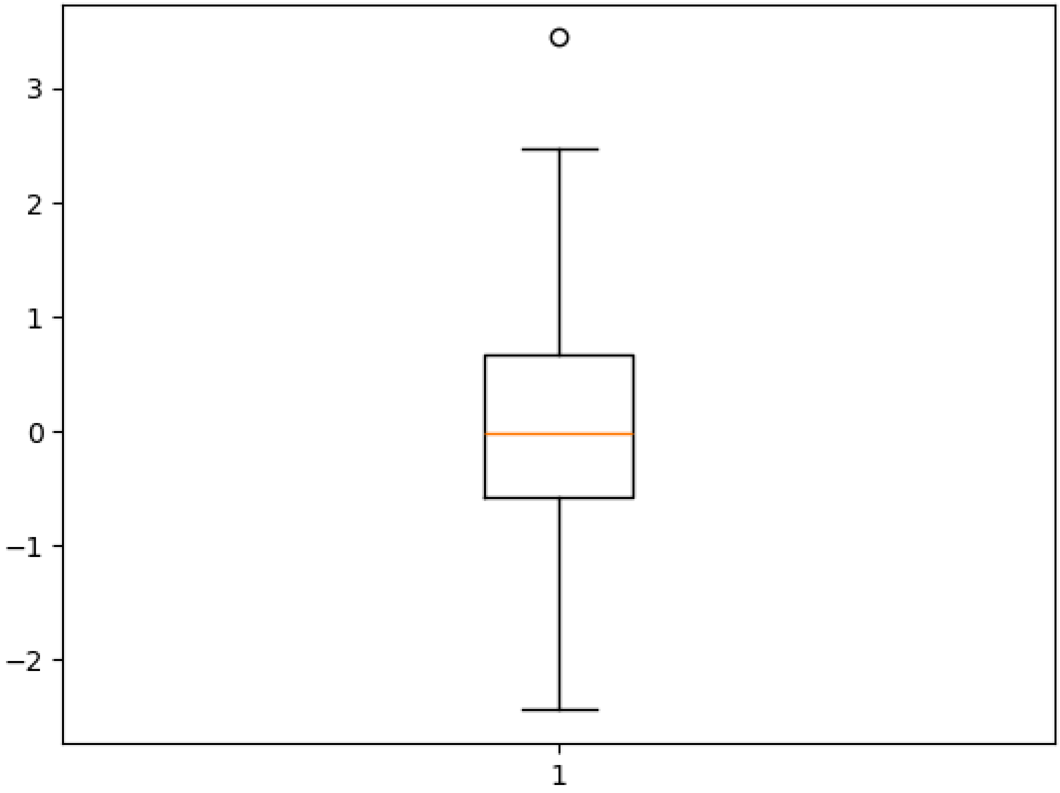
In [46]:

```
plt.figure(figsize=(7,7))
x10=[35,25,10,10,20]
labels=['computers','electronics','mechanical','chemical','arts']
plt.pie(x10,labels=labels)
plt.show()
```



Box plot

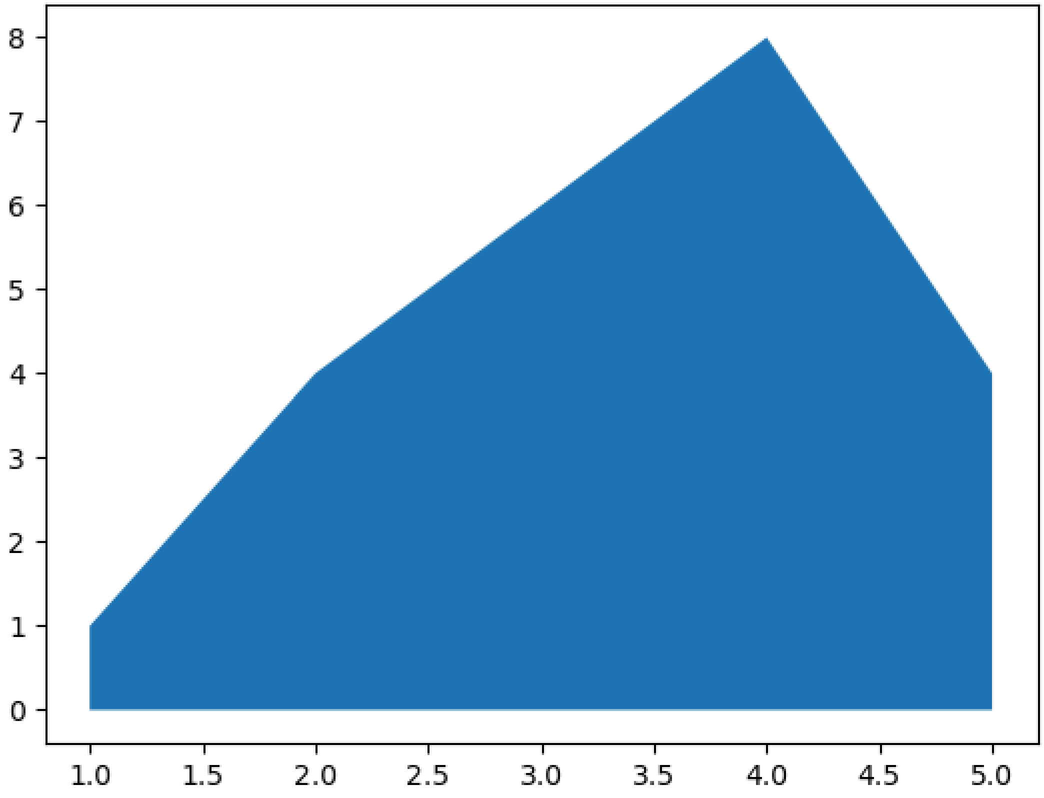
In [47]:
```
data3=np.random.randn(100)
plt.boxplot(data3)
plt.show()
```



AreaChart

In [53]:

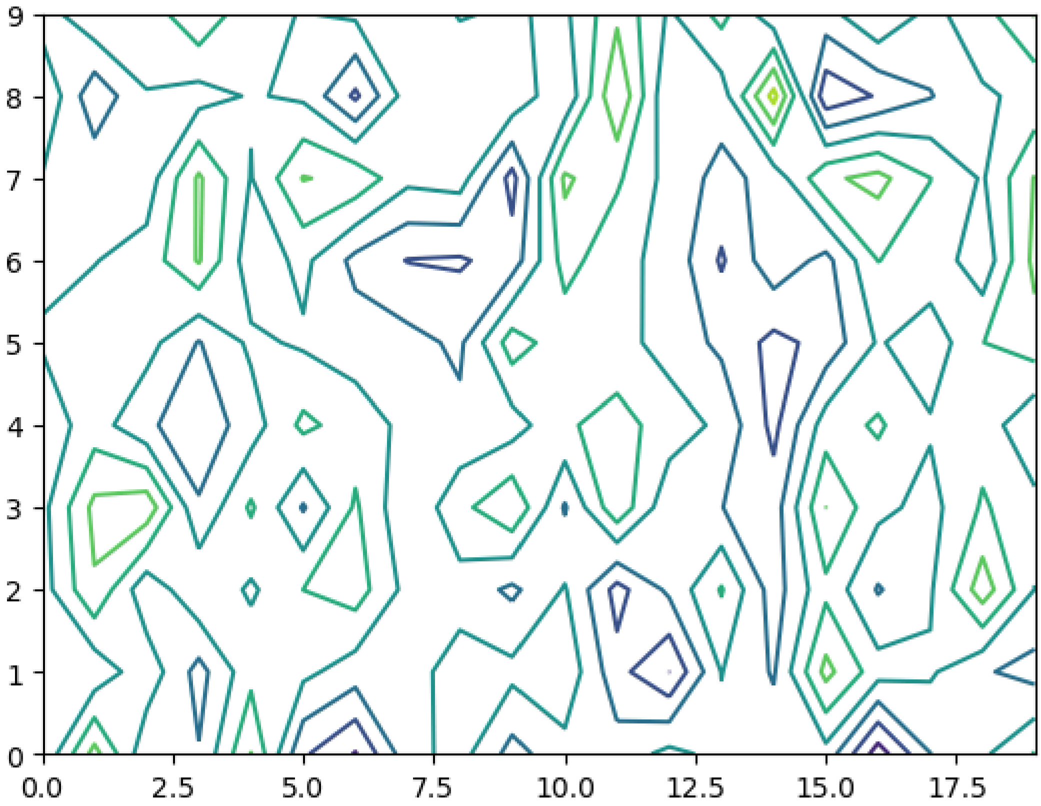
```
x12=range(1,6)
y12=[1,4,6,8,4]
#Area plot
plt.fill_between(x12,y12)
plt.show()
```



contour plots

In [55]:

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



styles with matplotlib

In [5]:

```
#view list of all available styles
import matplotlib.pyplot as plt
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']
```

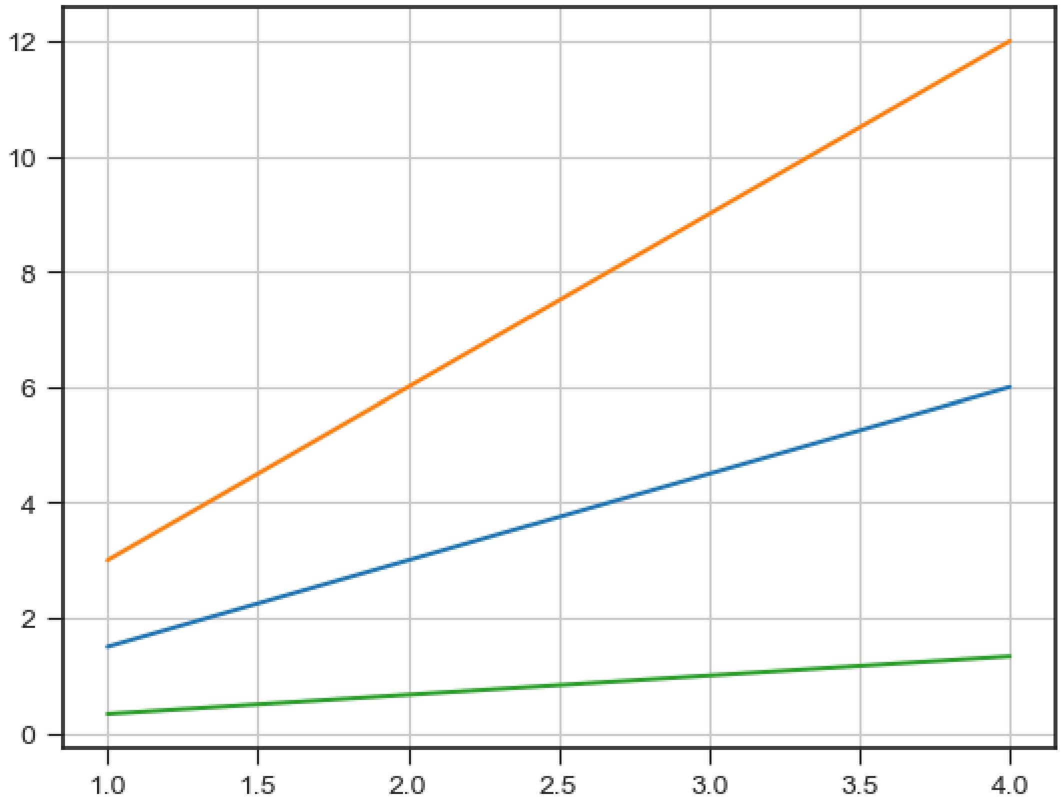
In [10]:

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

Adding grid

In [12]:

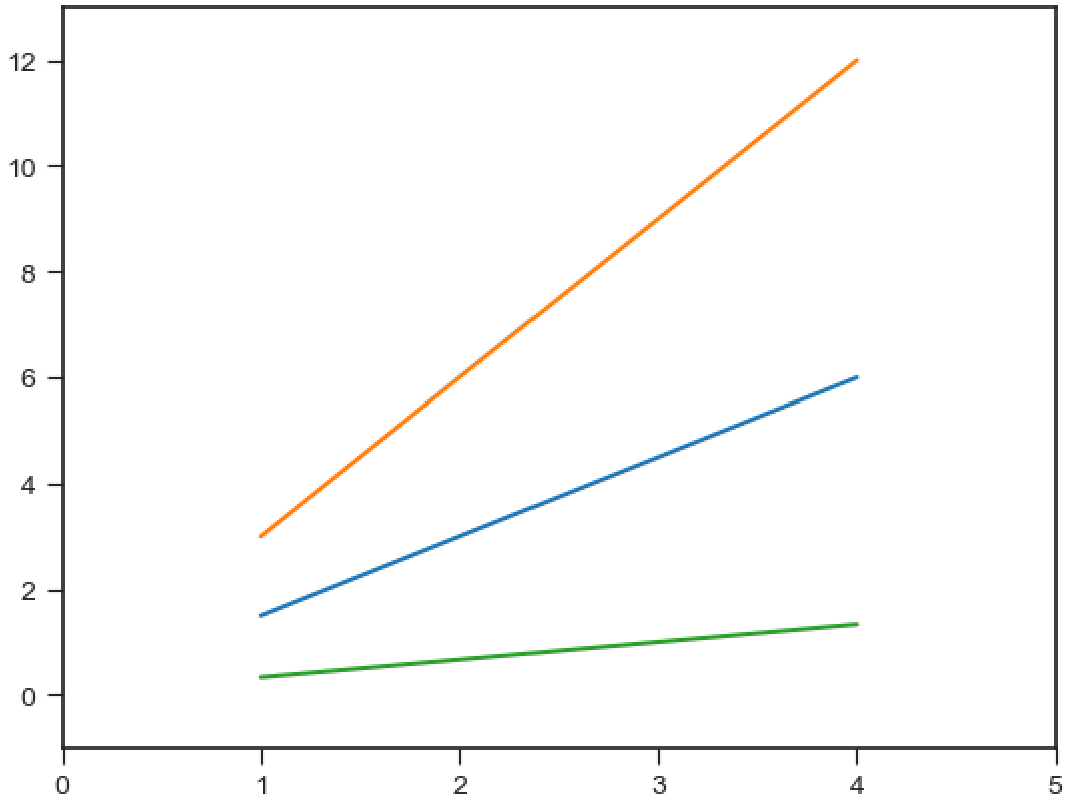
```
x15=np.arange(1,5)
plt.plot(x15,x15*1.5,x15,x15*3.0,x15,x15/3.0)
plt.grid(True)
plt.show()
```



Handling axes

In [16]:

```
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()
```

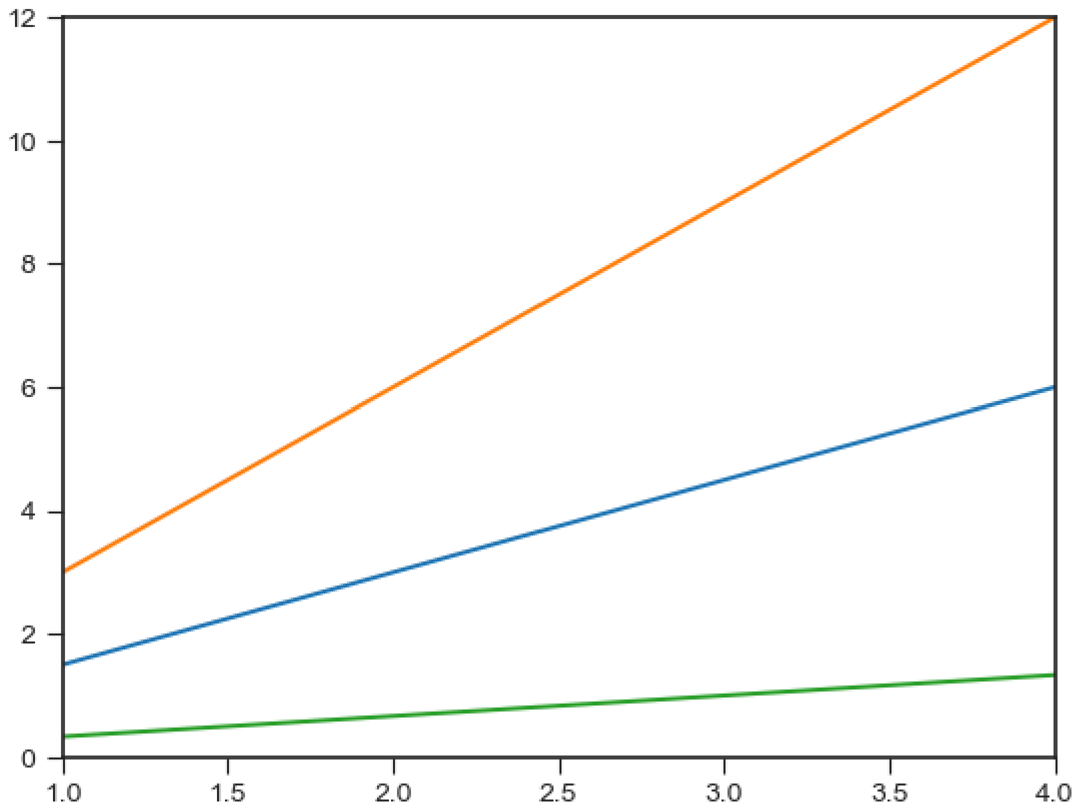


In [18]:

```
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[18]:

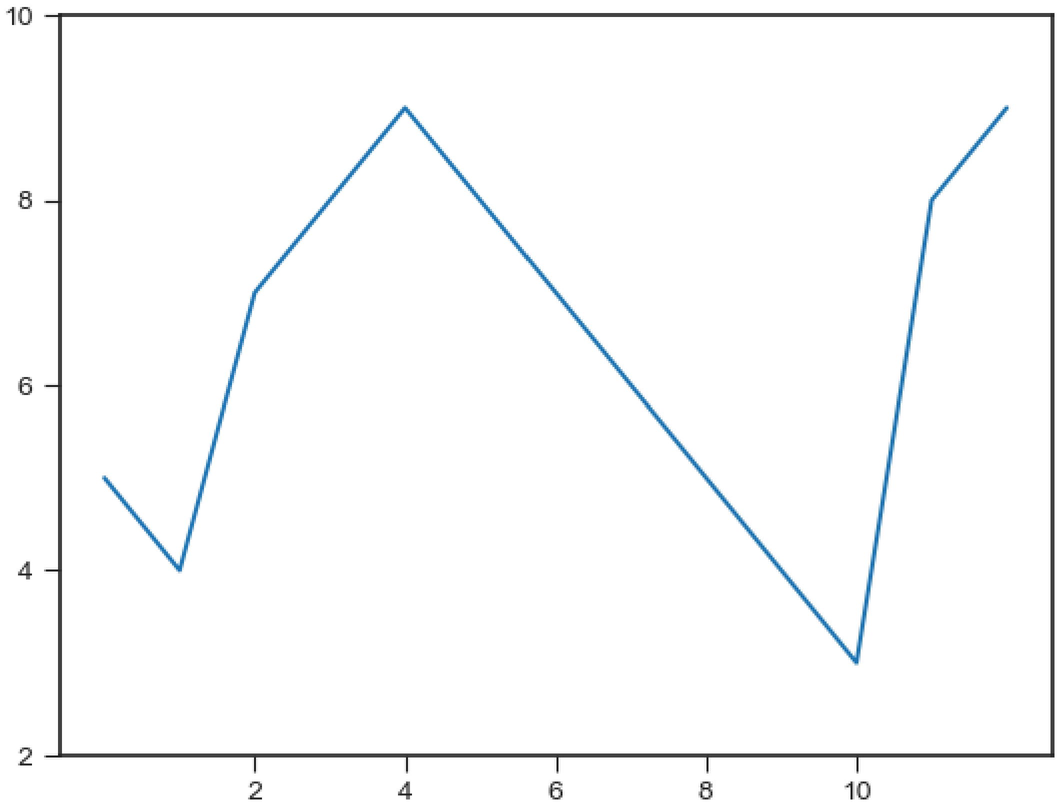
```
(0.0, 12.0)
```



Handling Xticks & Y ticks

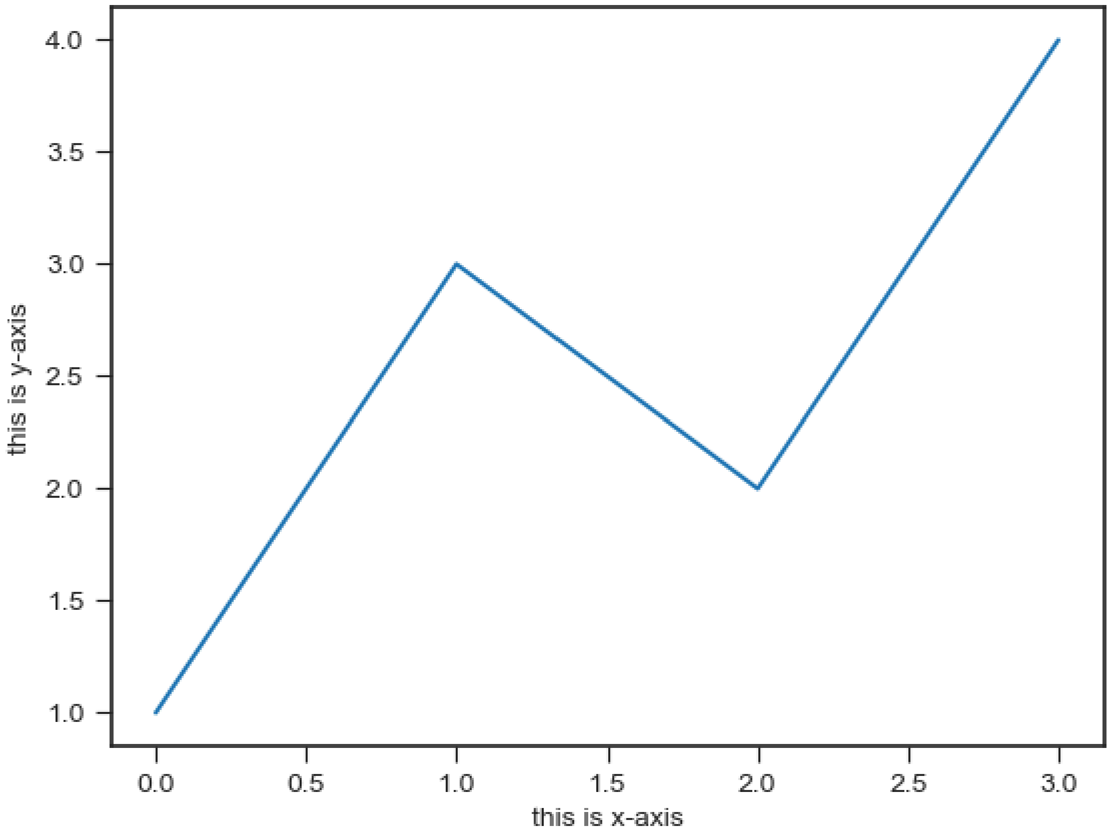
In [21]:

```
u=[5,4,7,8,9,8,7,6,5,4,3,8,9]
plt.plot(u)
plt.xticks([2,4,6,8,10])
plt.yticks([2,4,6,8,10])
plt.show()
```



```
In [ ]: # Adding label
```

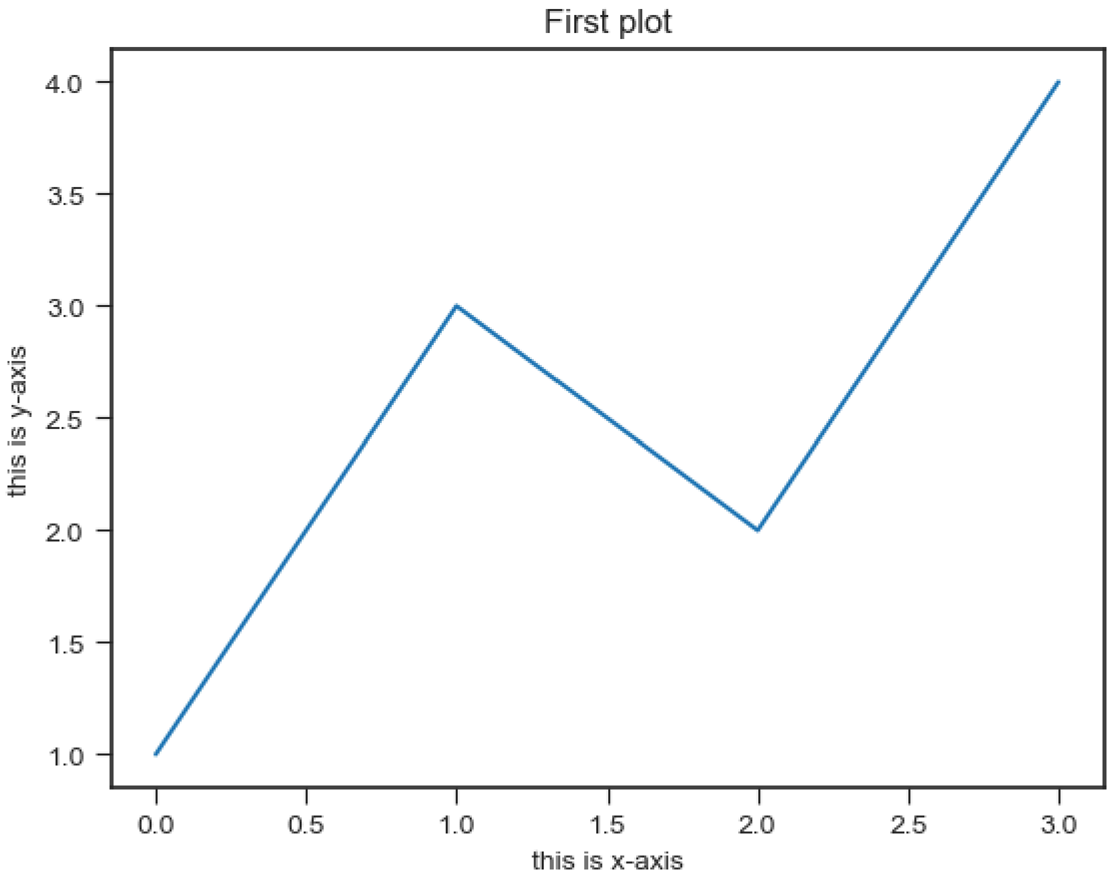
```
In [22]: plt.plot([1,3,2,4])     #bydefault x[0,1,2,3]
         plt.xlabel("this is x-axis")
         plt.ylabel("this is y-axis")
         plt.show()
```



Adding title

In [24]:

```
plt.plot([1,3,2,4])     #bydefault x[0,1,2,3]
plt.xlabel("this is x-axis")
plt.ylabel("this is y-axis")
plt.title("First plot")
plt.show()
```



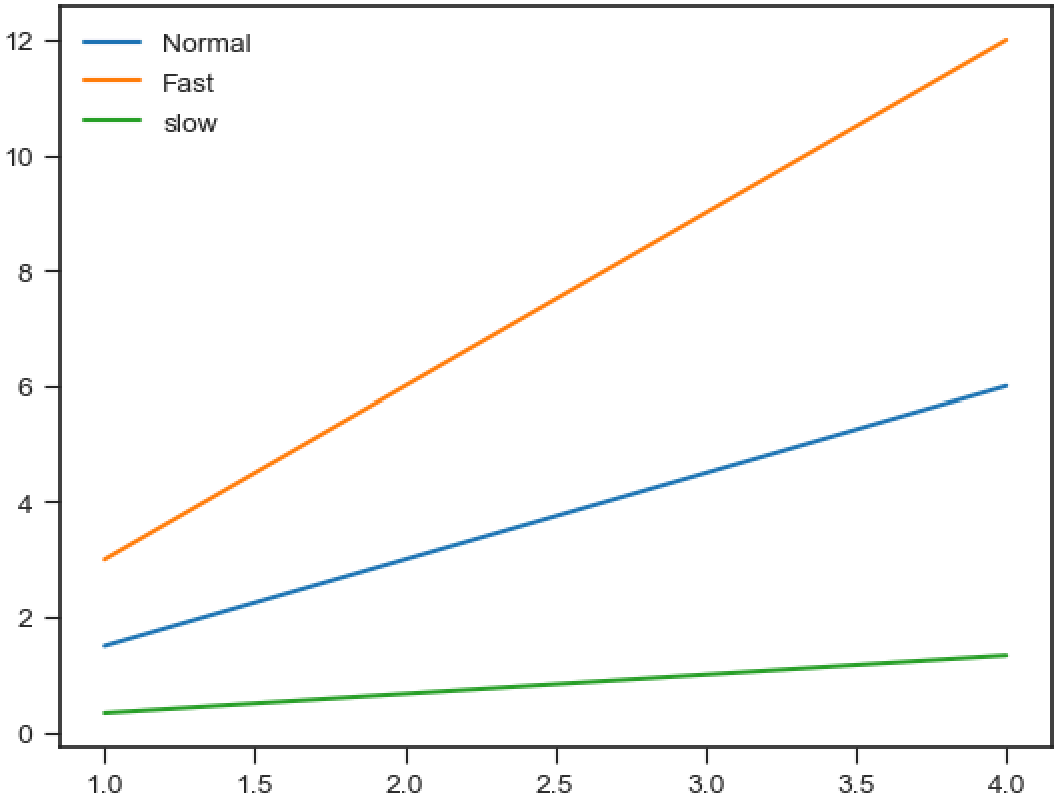
Adding legend

In [28]:

```
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'])
```

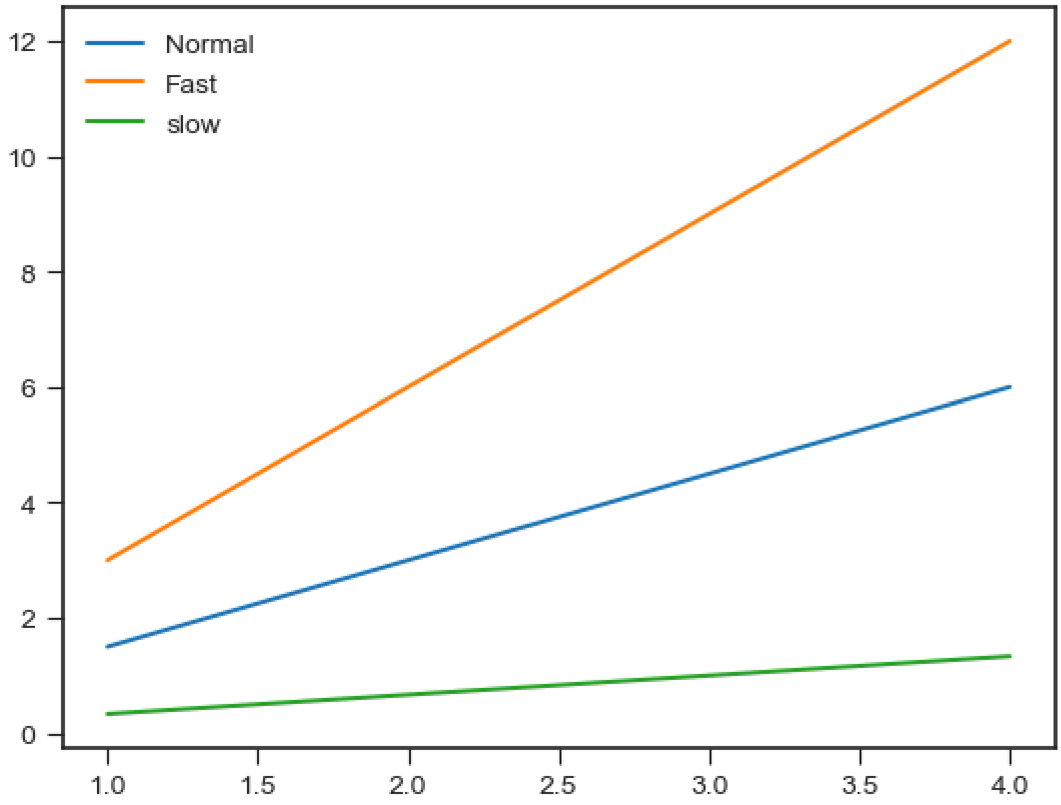
Out[28]:

```
<matplotlib.legend.Legend at 0x284619515d0>
```



```
In [29]: x15=np.arange(1,5)      #other method
         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()
```

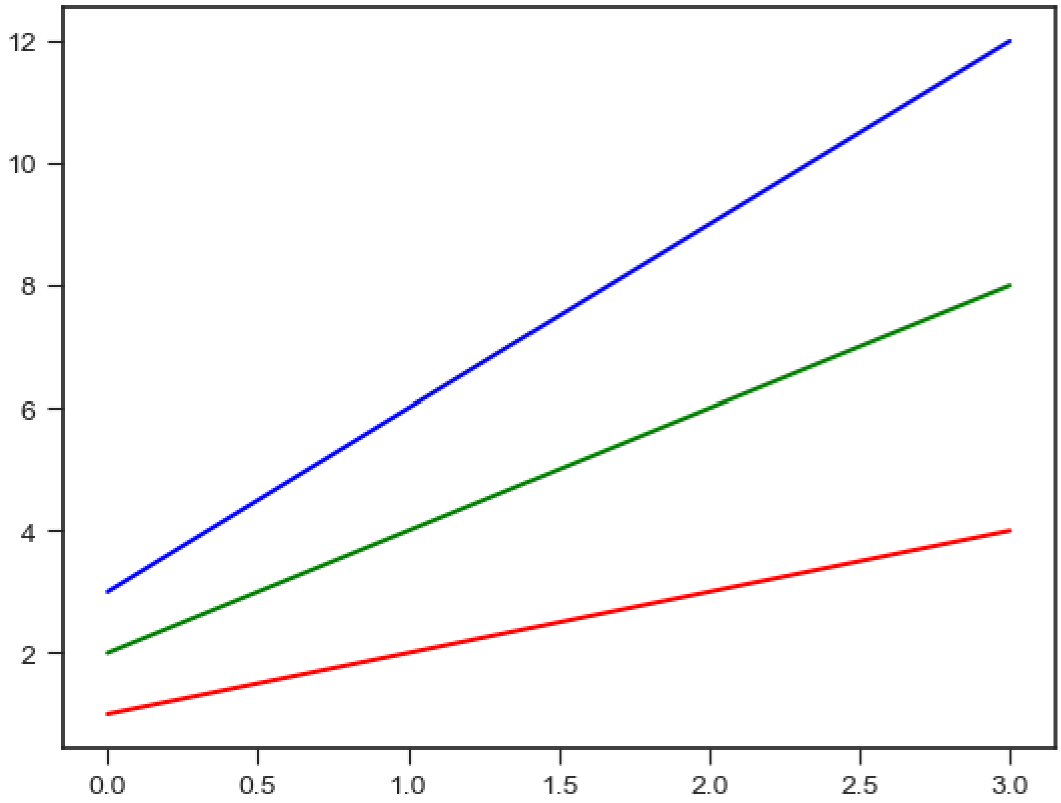
```
Out[29]: <matplotlib.legend.Legend at 0x2845e3553d0>
```



control colours

In [30]:

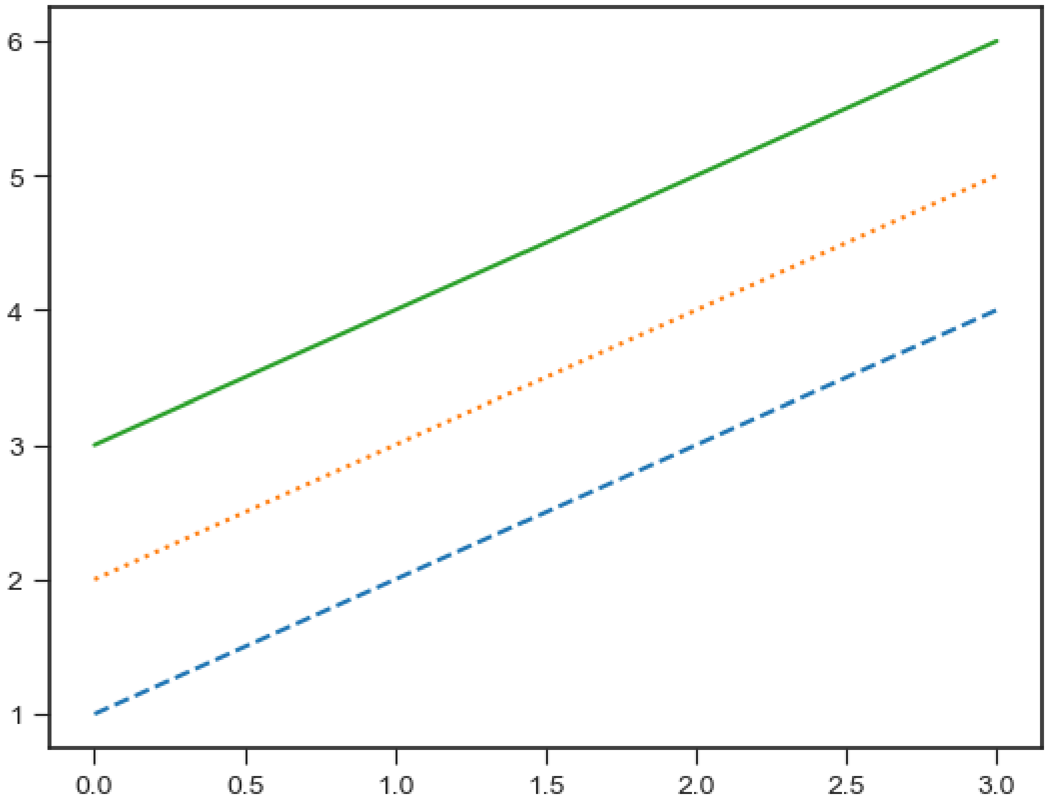
```
x16=np.arange(1,5)
plt.plot(x16,'r')
plt.plot(x16*2,'g')
plt.plot(x16*3,'b')
plt.show()
```



contrlol line styles

In [32]:

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



In [ ]: