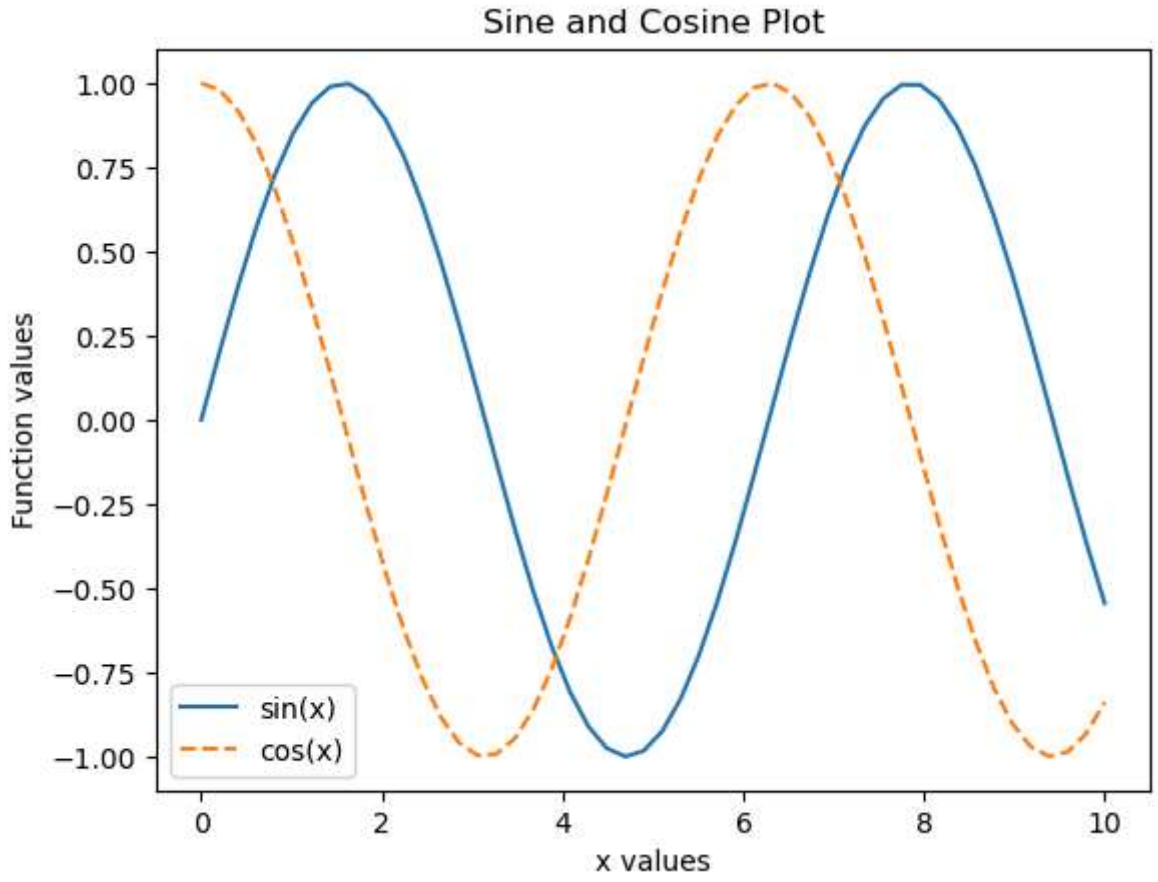
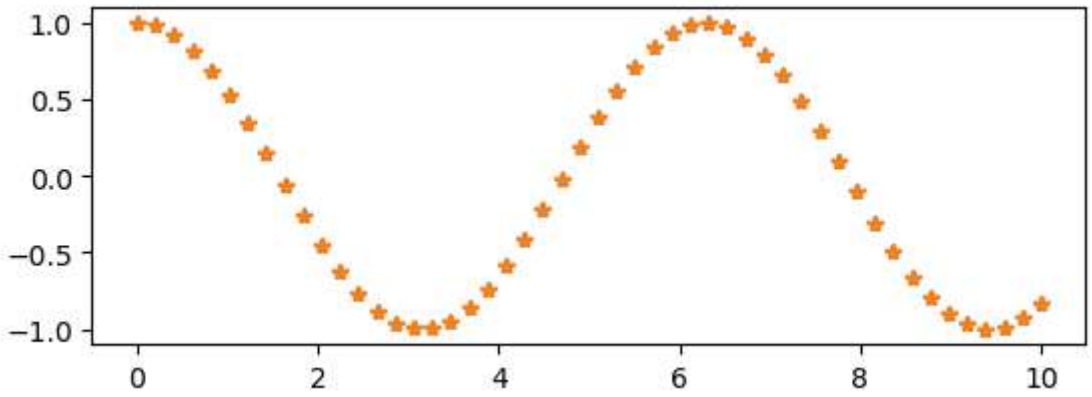
In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

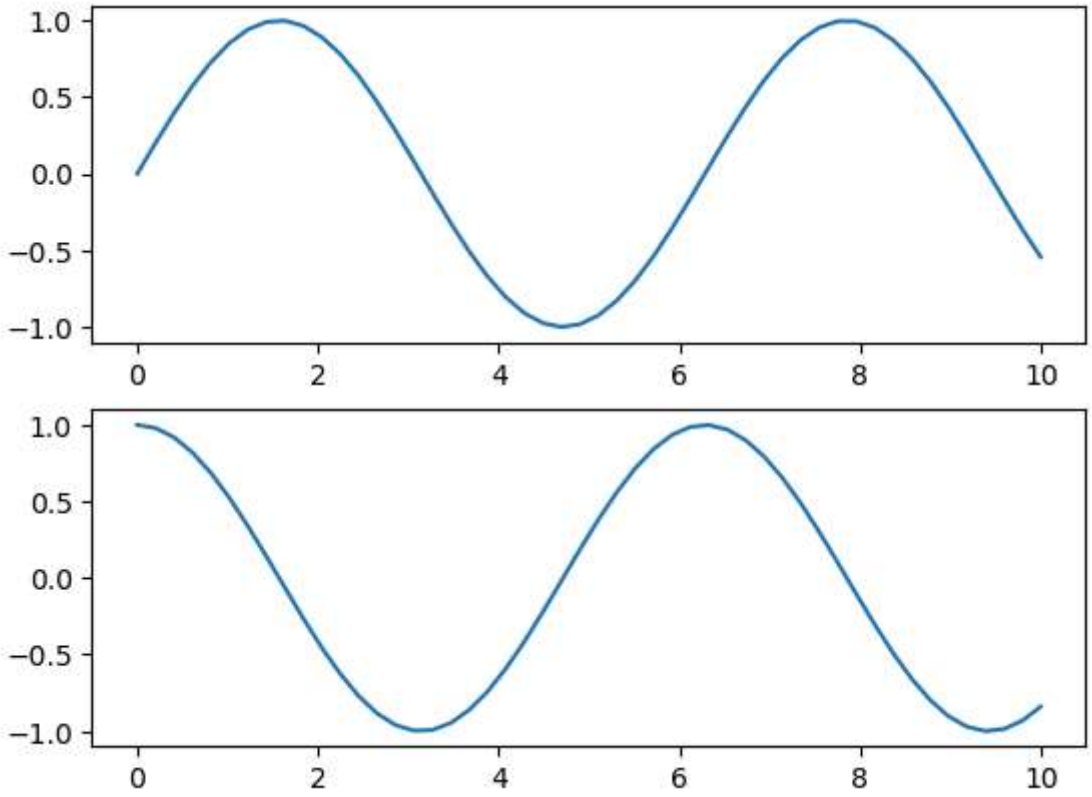
In [15]:
```
%matplotlib inline

x1 = np.linspace(0, 10, 50)  # Generate 50 values between 0 and 10

plt.plot(x1, np.sin(x1), '-',label="sin(x)")  # Solid line for sin(x)
plt.plot(x1, np.cos(x1), '--', label="cos(x)")  # Dashed line for cos(x)
#plt.plot(x1, np.tan(x1), '--', label="tan(x)")  # Avoid tan(x) due to asymptotes

plt.legend()  # Add a legend
plt.xlabel("x values")  # X-axis label
plt.ylabel("Function values")  # Y-axis label
plt.title("Sine and Cosine Plot")  # Plot title

plt.show()  # Display the plot
```



In [19]:
```
# create the first of two panels and set current axis
plt.subplot(2, 1, 1)   # (rows, columns, panel number)
plt.plot(x1, np.cos(x1), '*')
plt.show()
```



In [31]:
```
# create a plot figure
plt.figure()


# create the first of two panels and set current axis
plt.subplot(2, 1, 1)   # (rows, columns, panel number)
plt.plot(x1, np.sin(x1))

# create the second of two panels and set current axis
plt.subplot(2, 1, 2)   # (rows, columns, panel number)
plt.plot(x1, np.cos(x1));
plt.show()
```



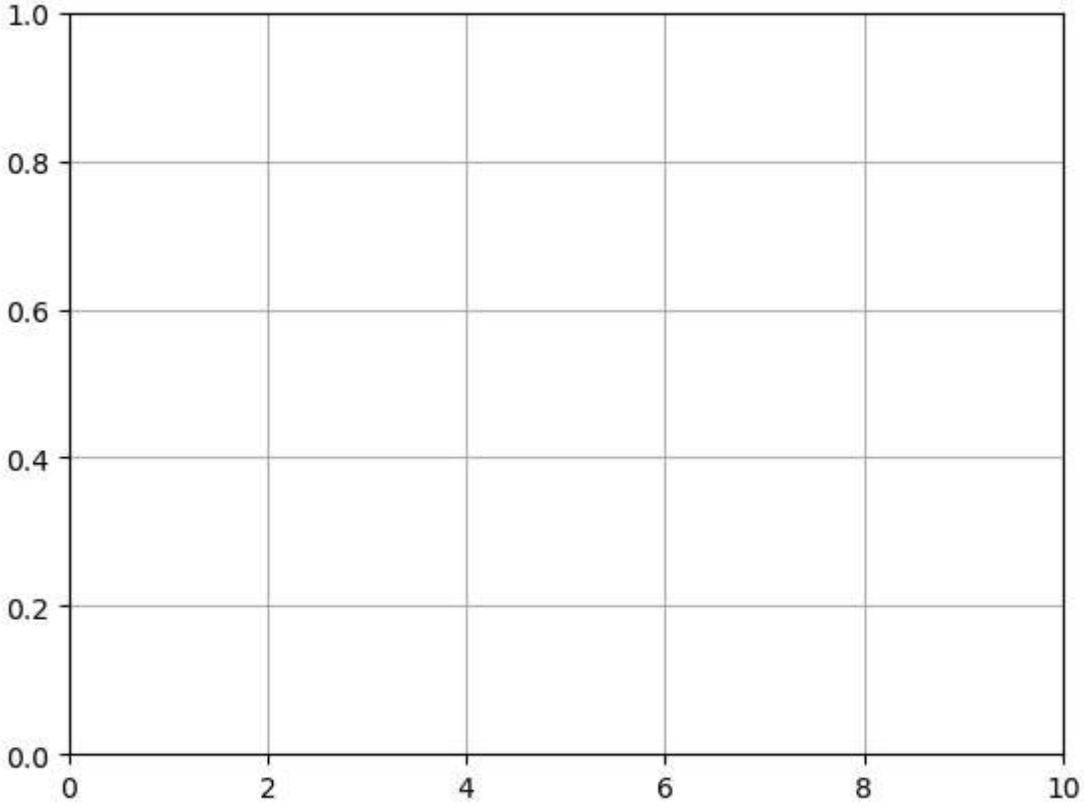
In [25]:
```
# get current figure information

print(plt.gcf())
```

```
Figure(640x480)
```

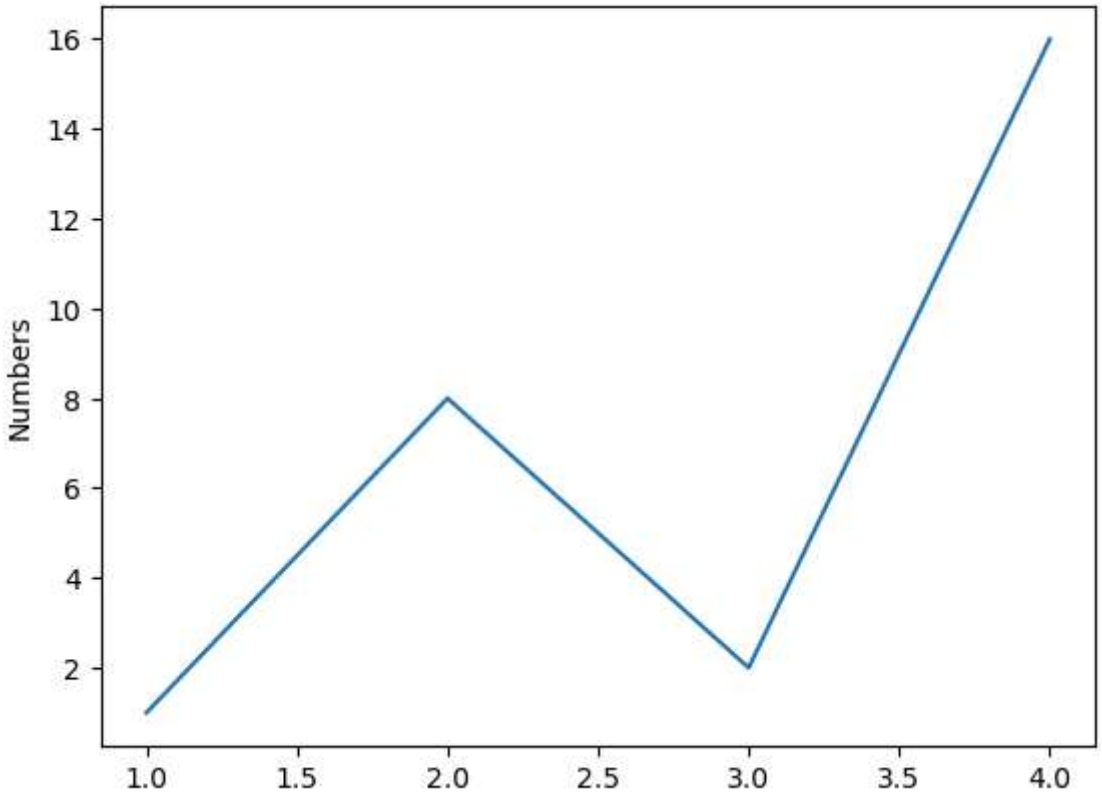
In [27]:
```
# get current axis information

print(plt.gca())
```

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

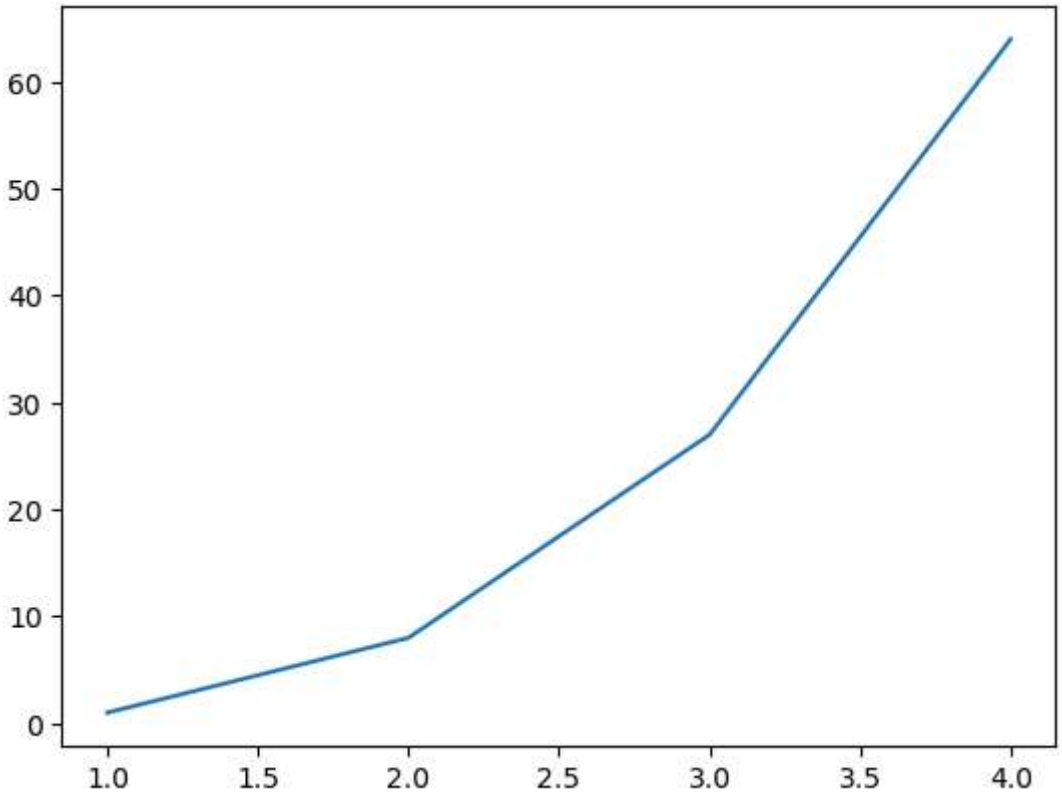
In [29]:
```
ax = plt.gca()
ax.set_xticks([0, 2, 4, 6, 8, 10])  # Set custom x-axis ticks
ax.grid(True)  # Enable grid
plt.show()
```



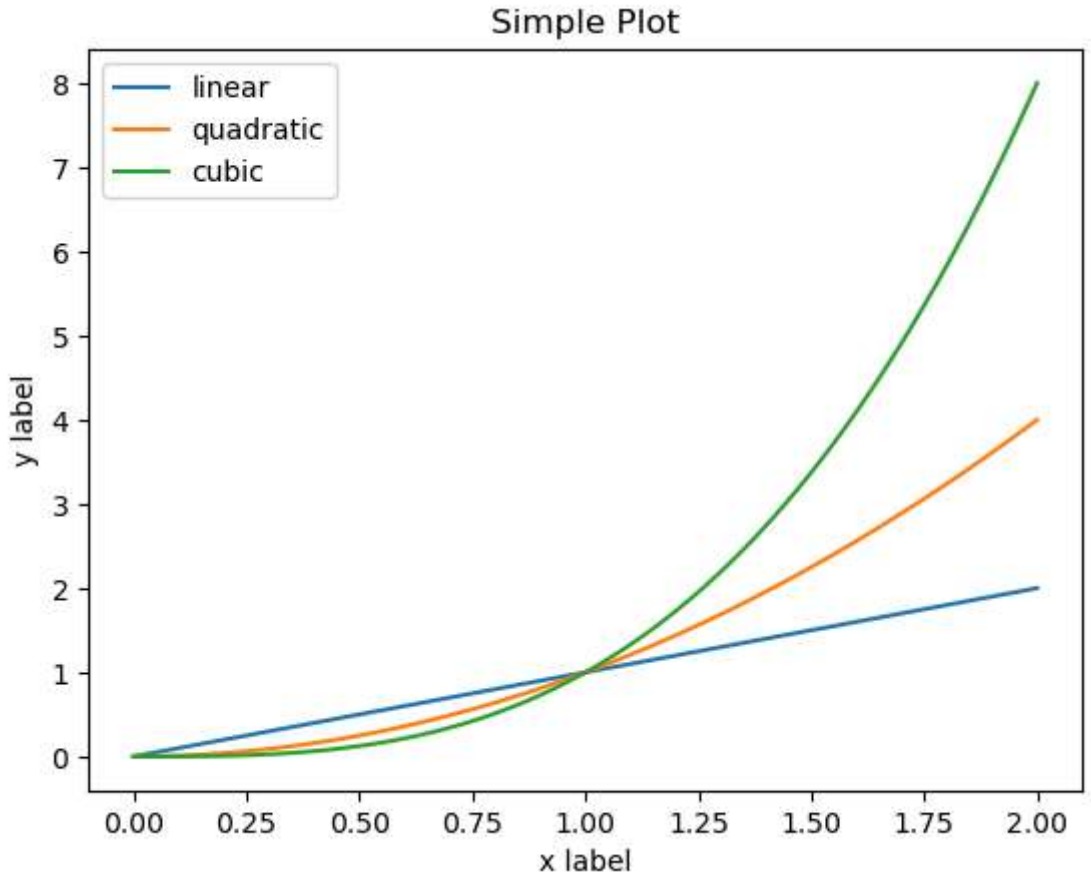
In [33]:
```
plt.plot([1,2,3,4], [1,8,2,16])
plt.ylabel('Numbers')
plt.show()
```



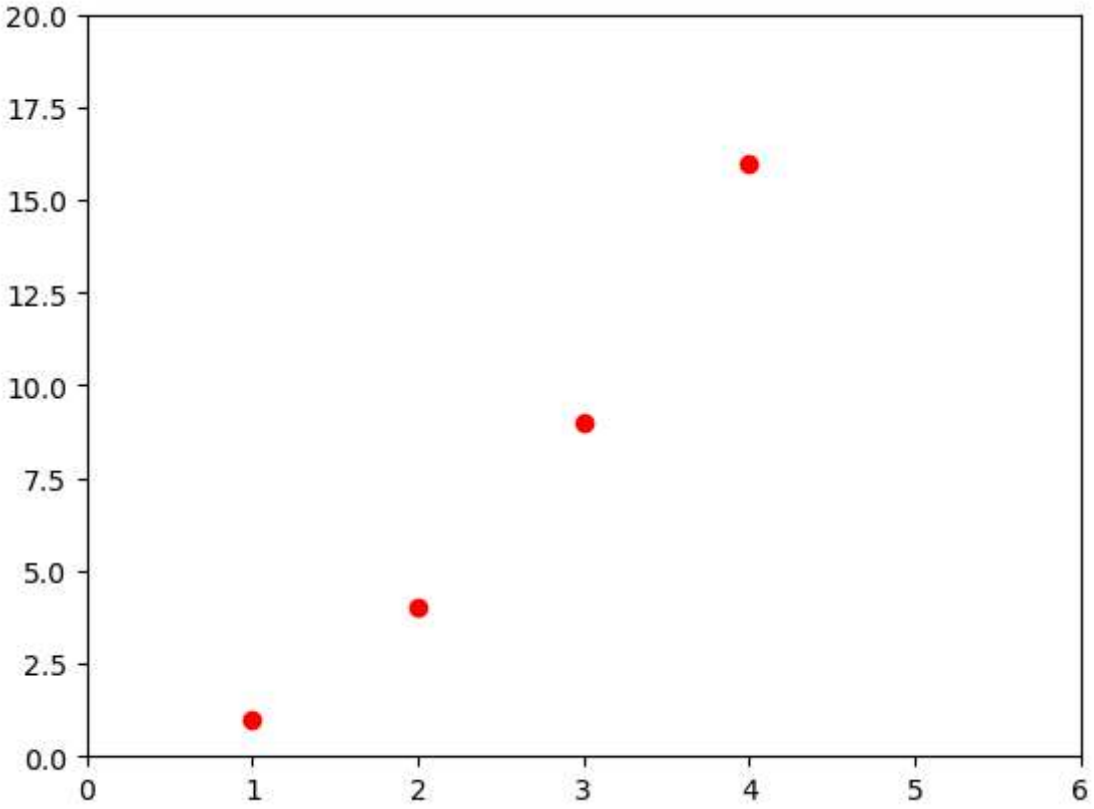
In [35]:
```
plt.plot([1, 2, 3, 4], [1, 8, 27, 64])
plt.show()
```



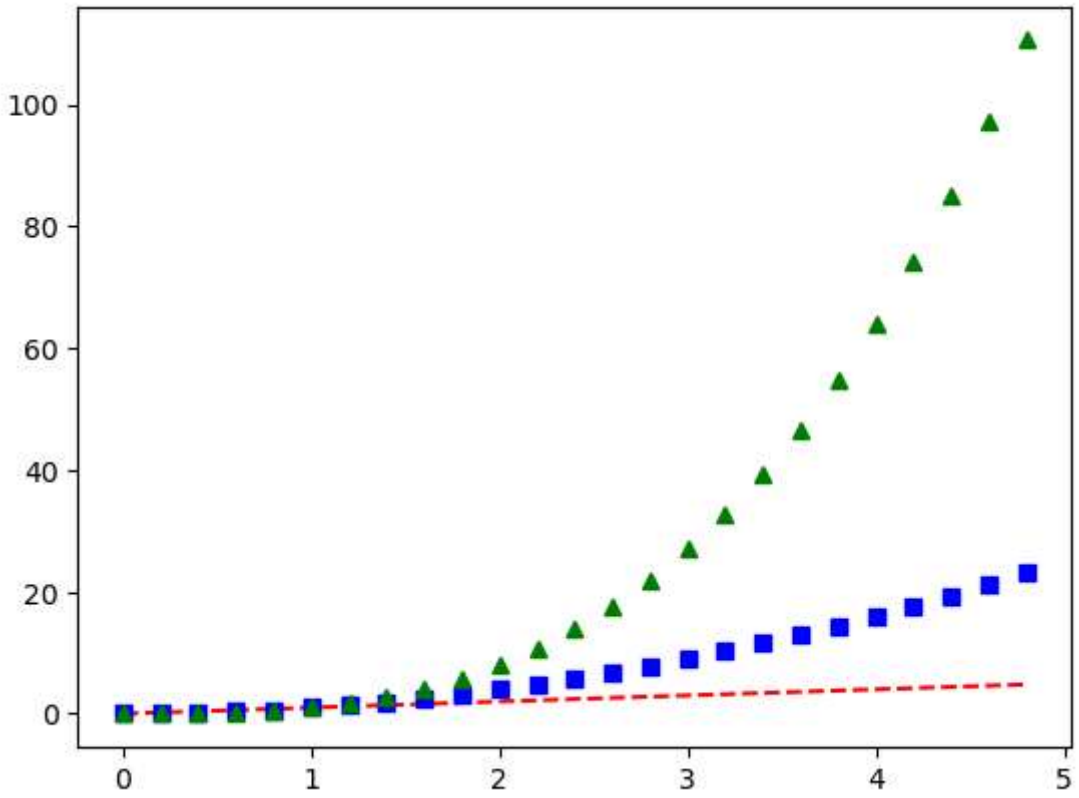
In [37]:
```
x = np.linspace(0, 2, 100)

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()
```



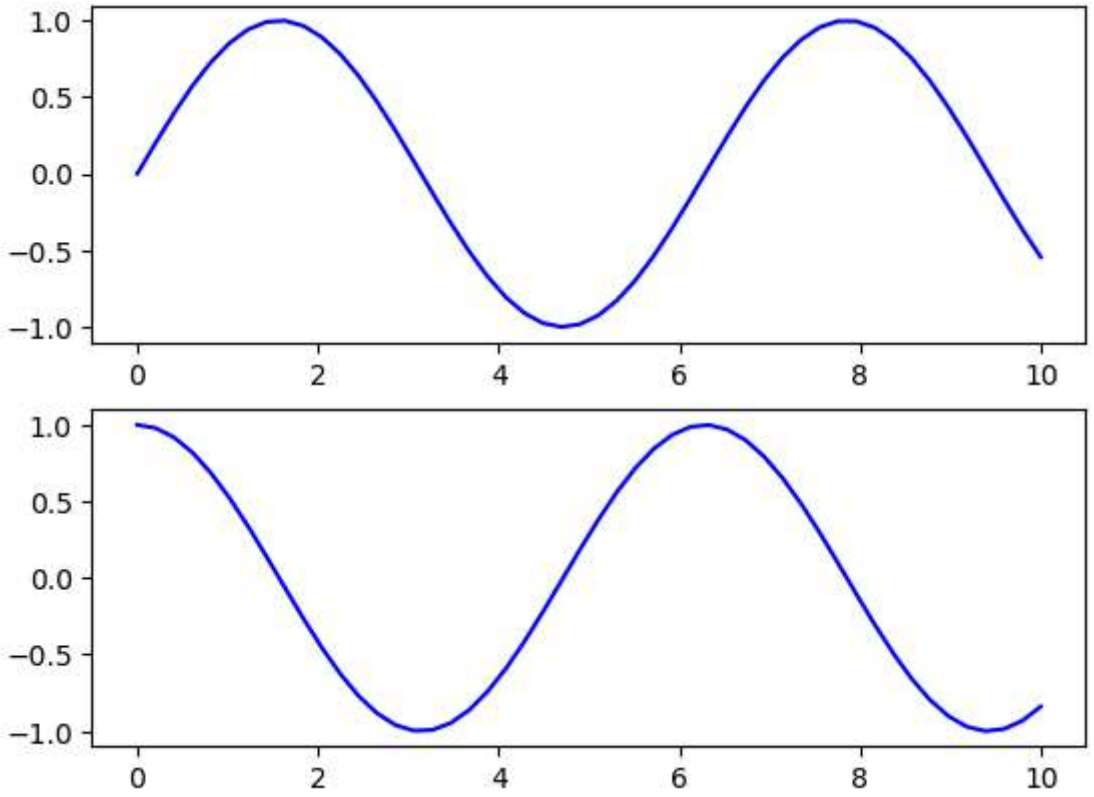
In [39]:
```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
plt.axis([0, 6, 0, 20])
plt.show()
```



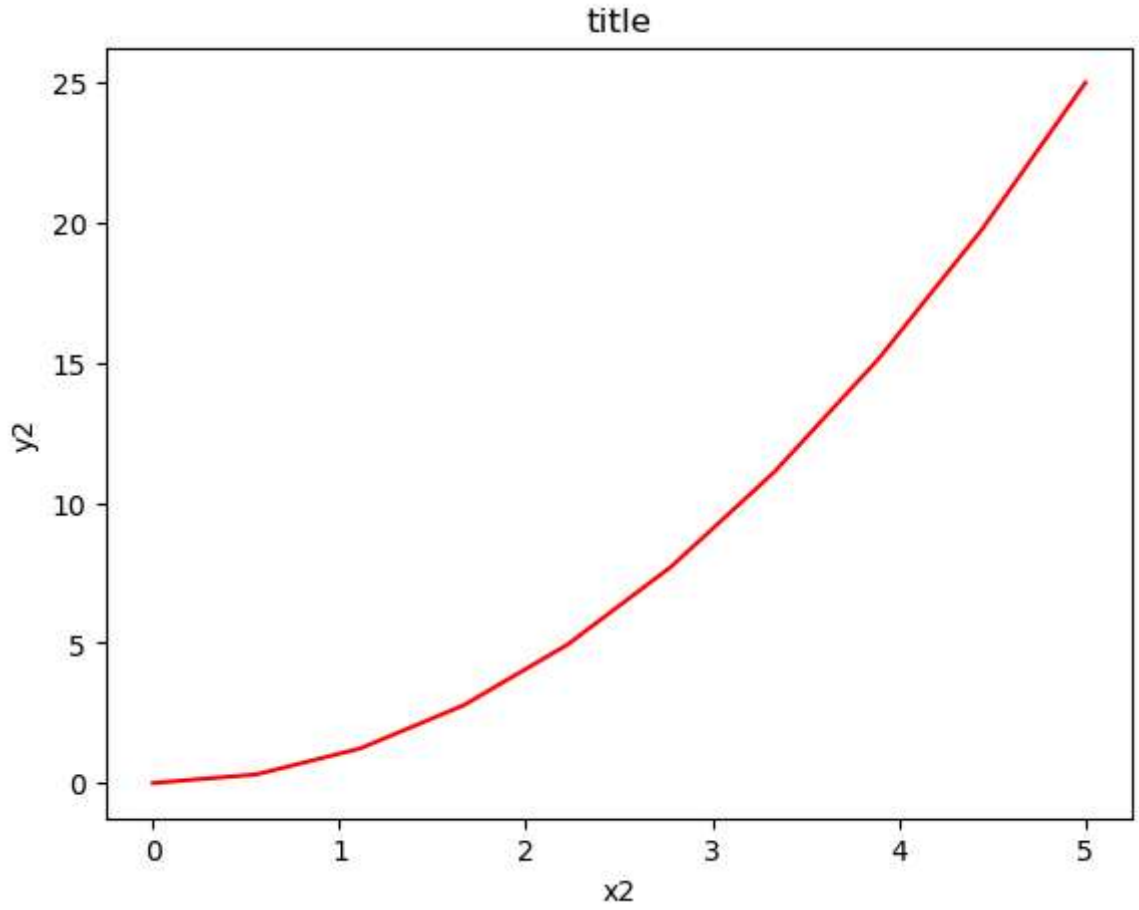
In [41]:
```
# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)

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



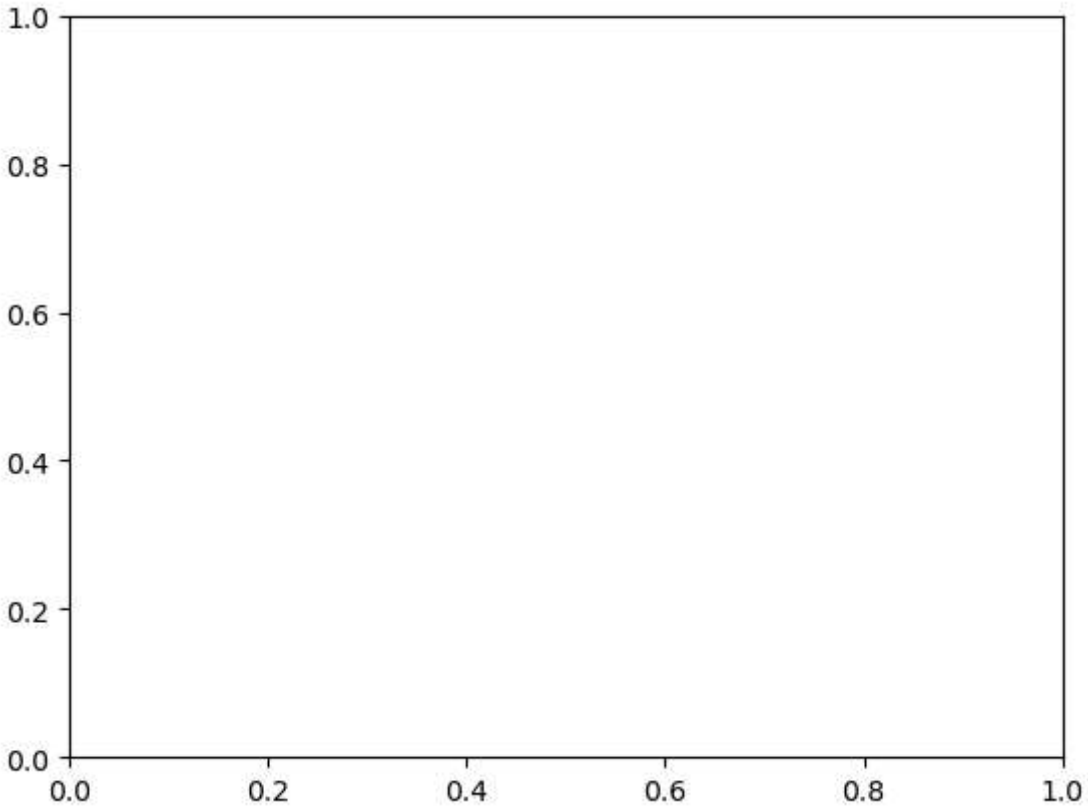
In [47]:
```
# First create a grid of plots
# ax will be an array of two Axes objects
fig, ax = plt.subplots(2)


# Call plot() method on the appropriate object
ax[0].plot(x1, np.sin(x1), 'b-')
ax[1].plot(x1, np.cos(x1), 'b-');
plt.show()
```



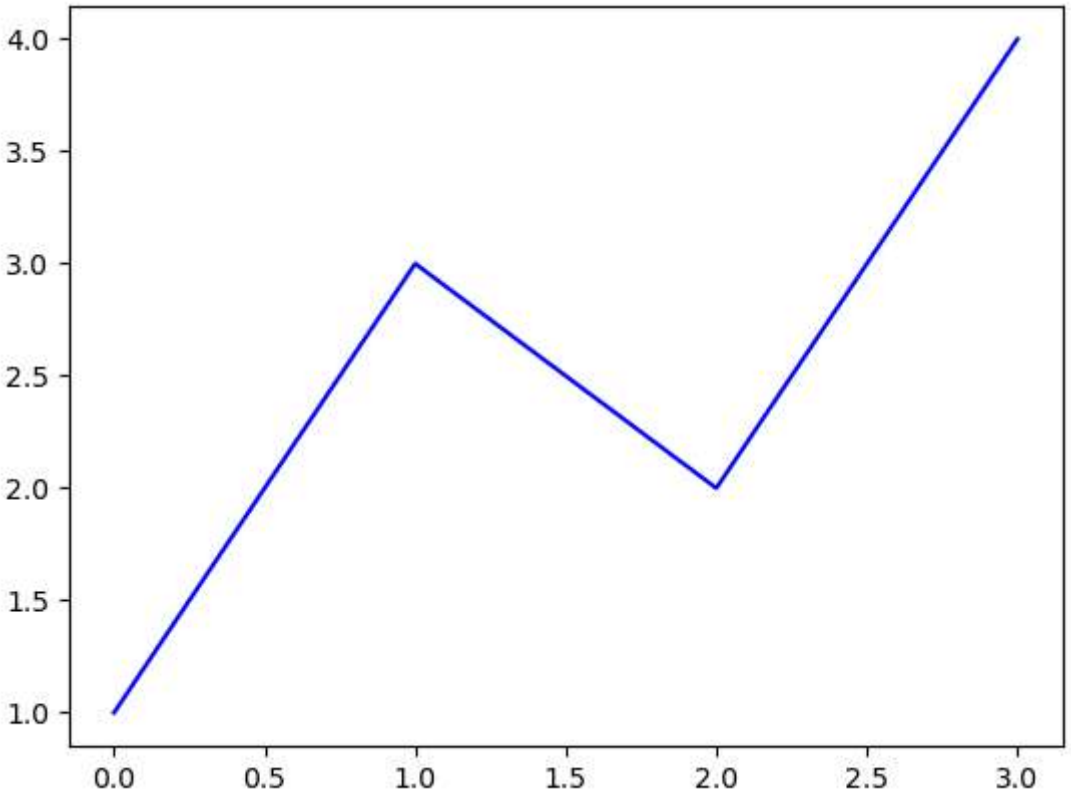
In [53]:
```
fig = plt.figure()

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()
```



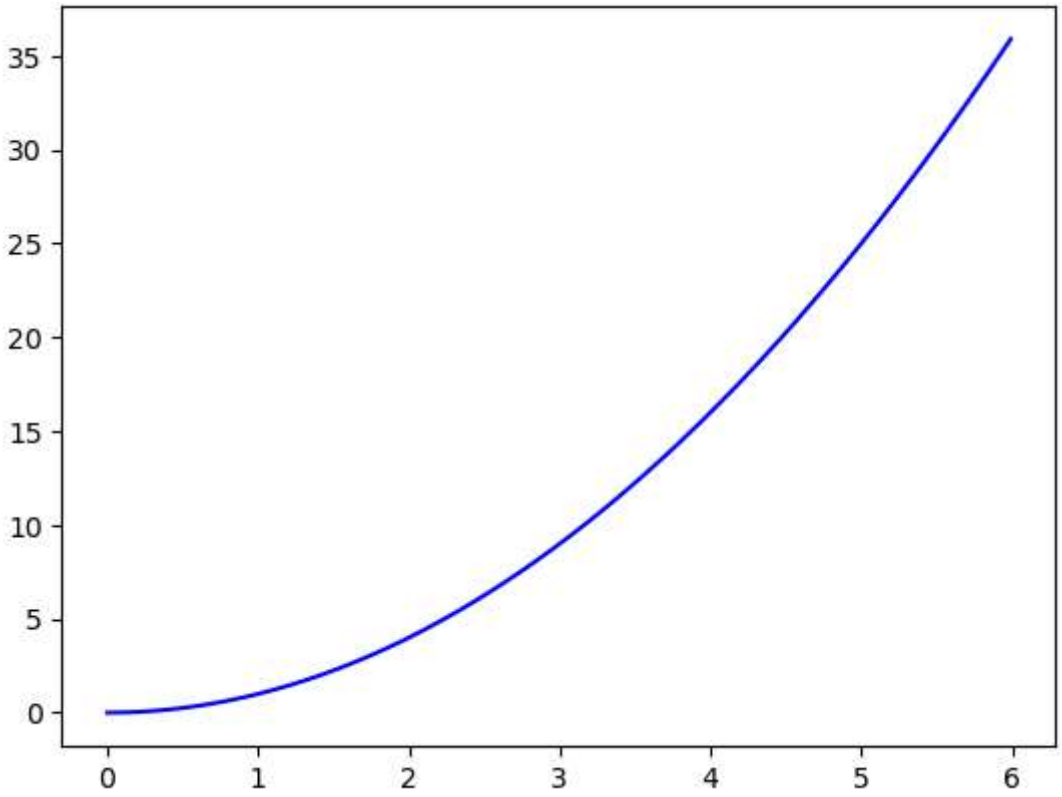
In [55]:
```
fig = plt.figure()

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



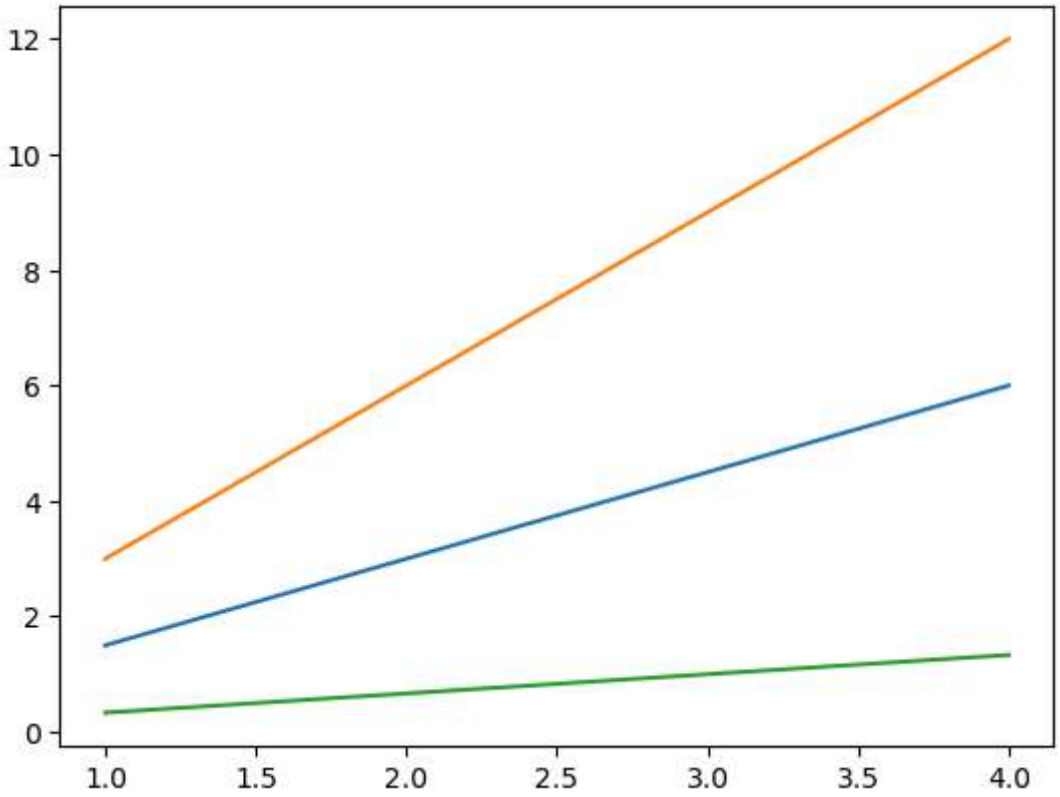
In [57]:
```
plt.plot([1, 3, 2, 4], 'b-')

plt.show( )
```



In [59]:
```
x3 = np.arange(0.0, 6.0, 0.01)

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

plt.show()
```



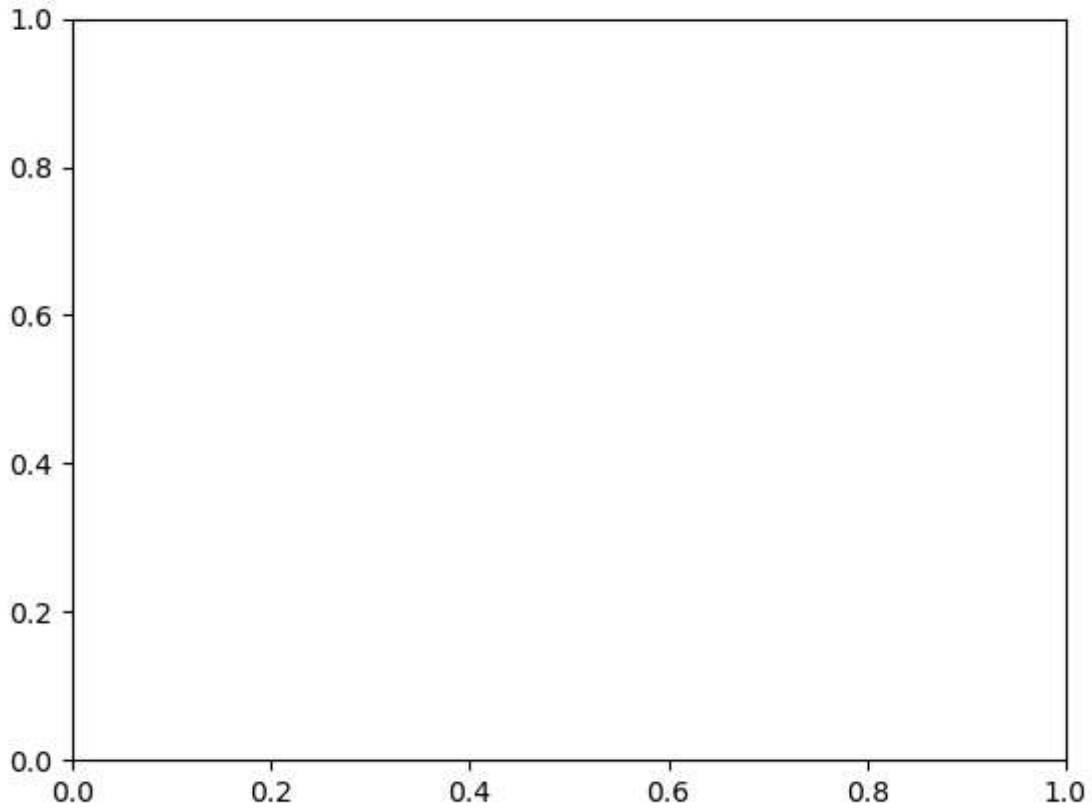
In [65]:
```
x4 = range(1, 5)

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 [67]:
```
# Saving the figure

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

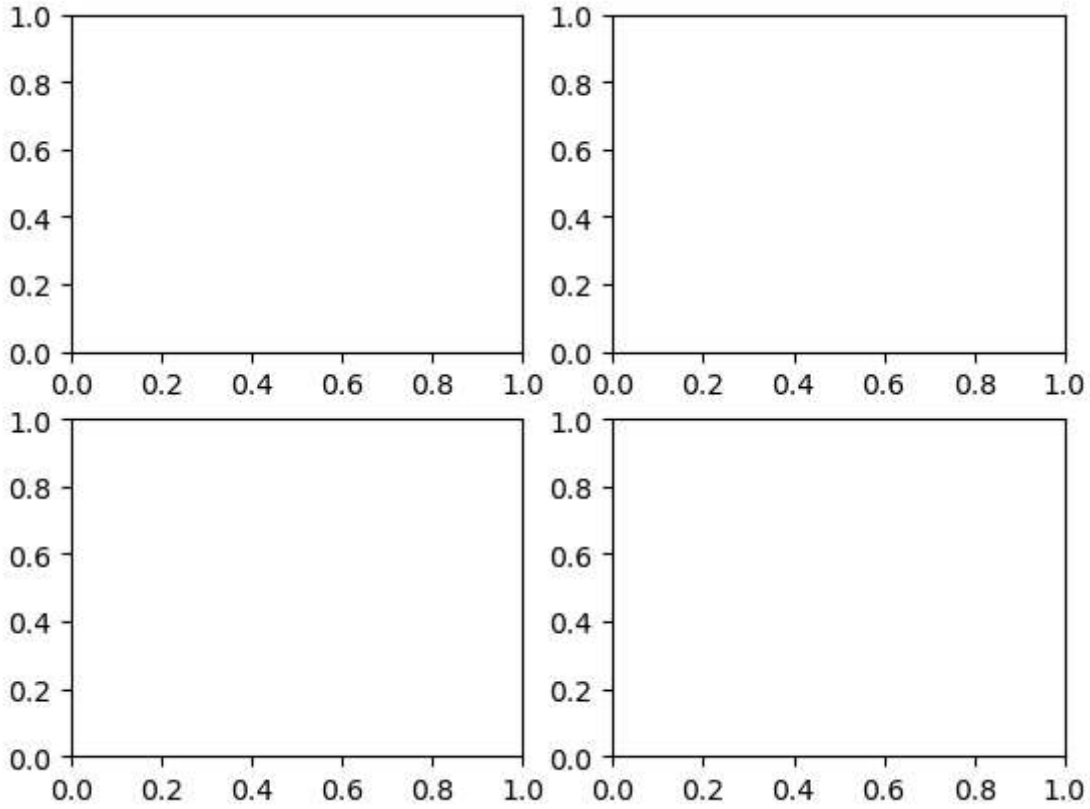
In [69]:
```
# Explore the contents of figure

from IPython.display import Image

Image('plot1.png')
```

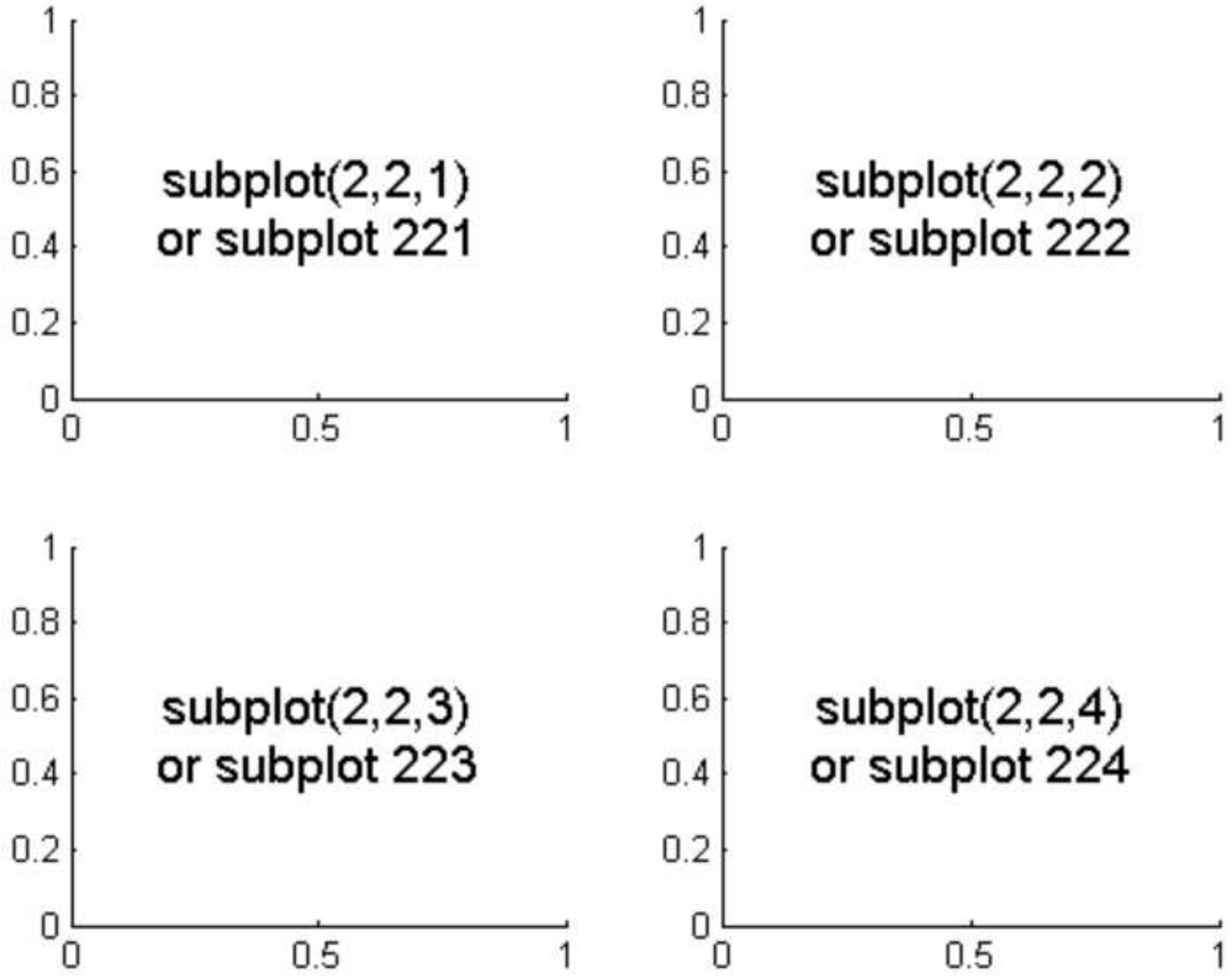
Out[69]:



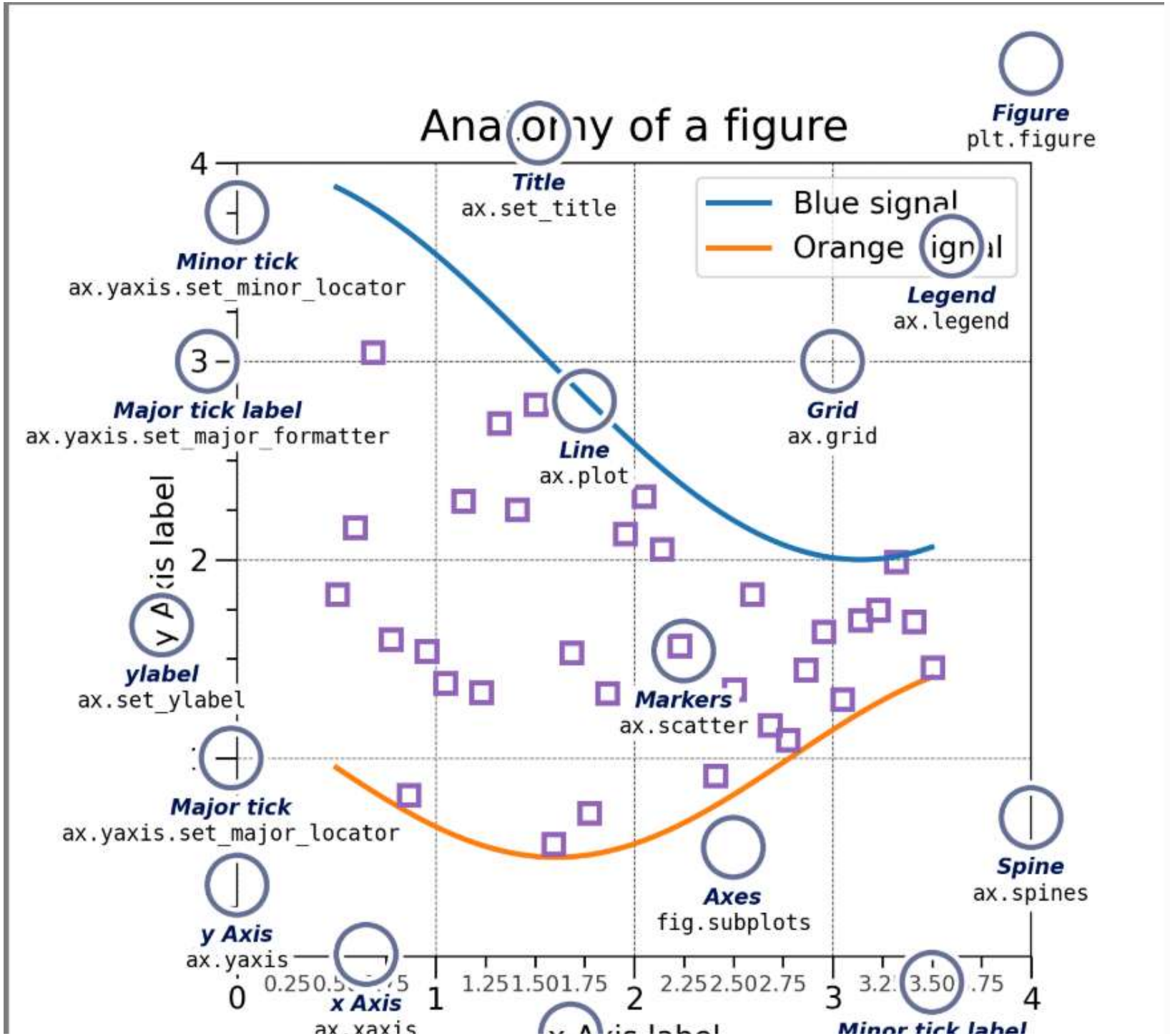
In [107…

```
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()
```

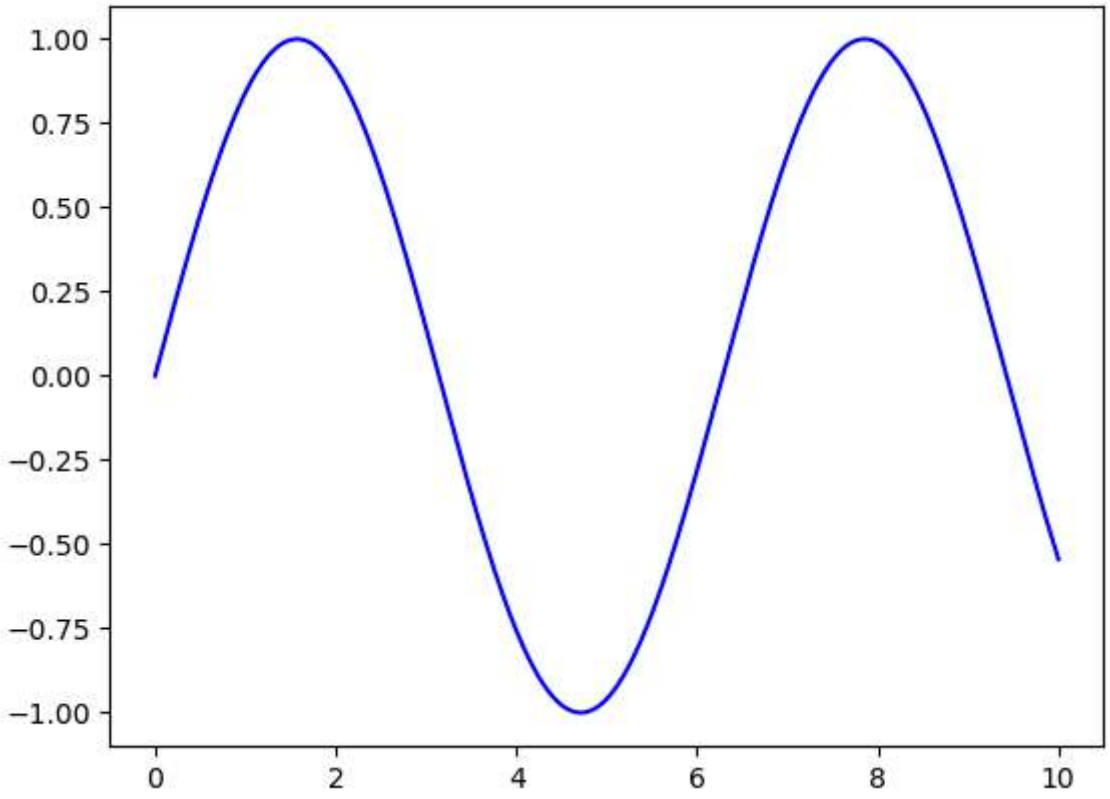

1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

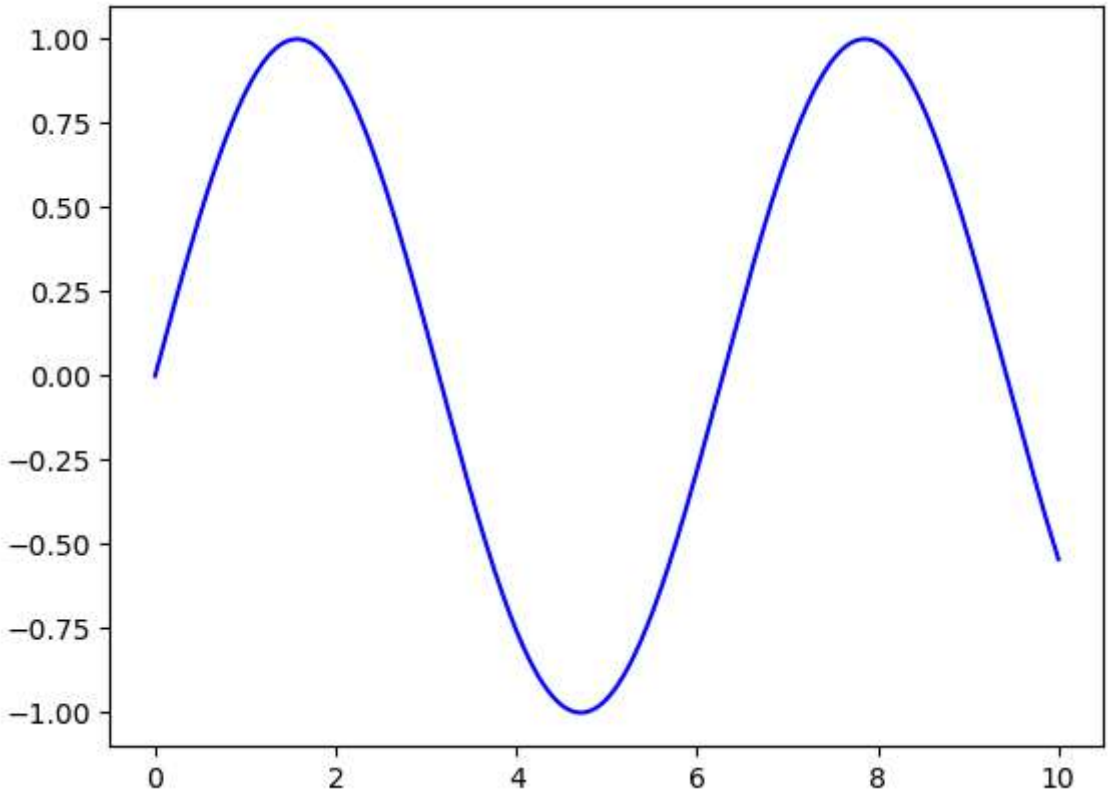


PARTS OF SPLOT

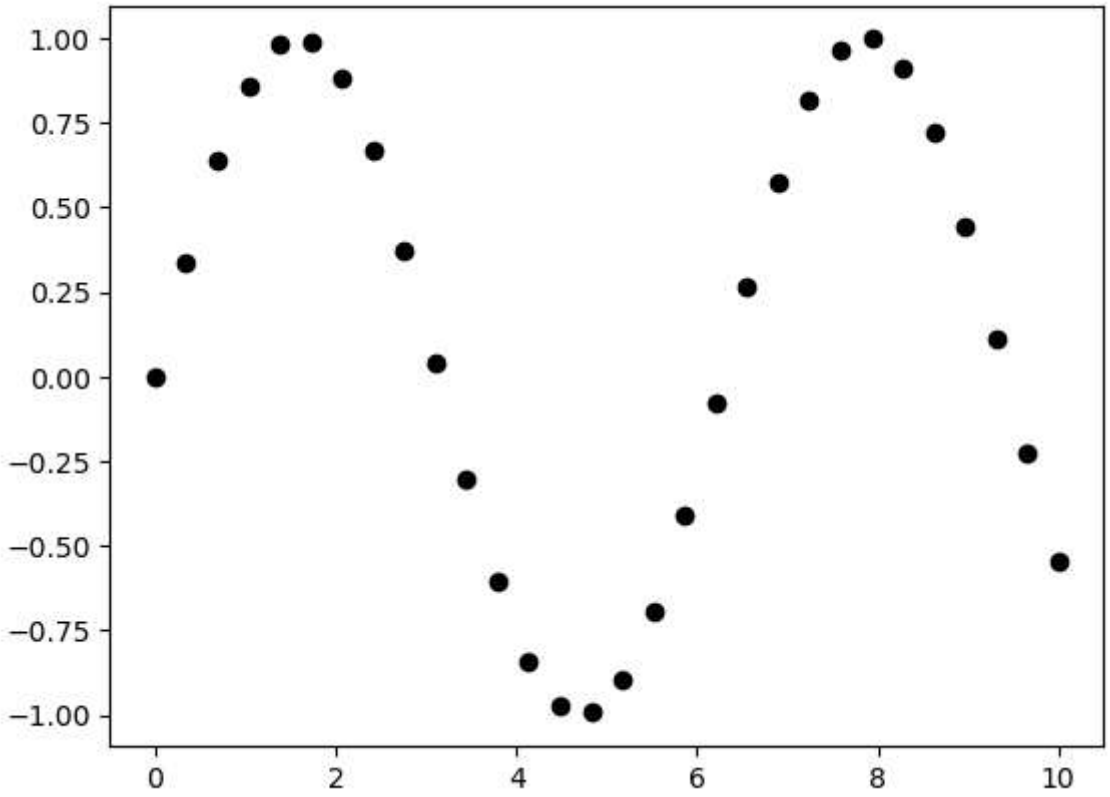

Anatomy of a figure
Figure
plt.figure
Title
ax.set_title
Blue signal
Orange signal
Legend
ax.legend
Minor tick
ax.yaxis.set_minor_locator
Major tick label
ax.yaxis.set_major_formatter
Grid
ax.grid
Line
ax.plot
ylabel
ax.set_ylabel
Markers
ax.scatter
Major tick
ax.yaxis.set_major_locator
Axes
fig.subplots
Spine
ax.spines
y Axis
ax.yaxis
x Axis
Minor tick label
0
1
2
3
4

In [73]:
```
# 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()
```

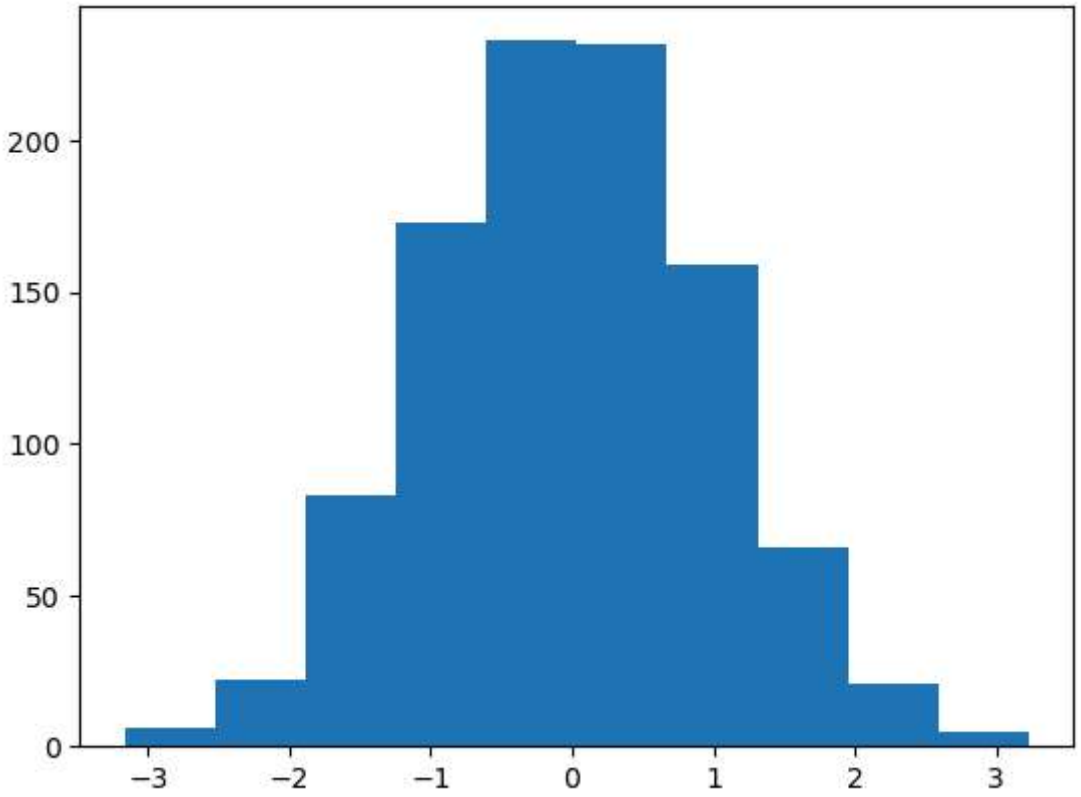

1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10



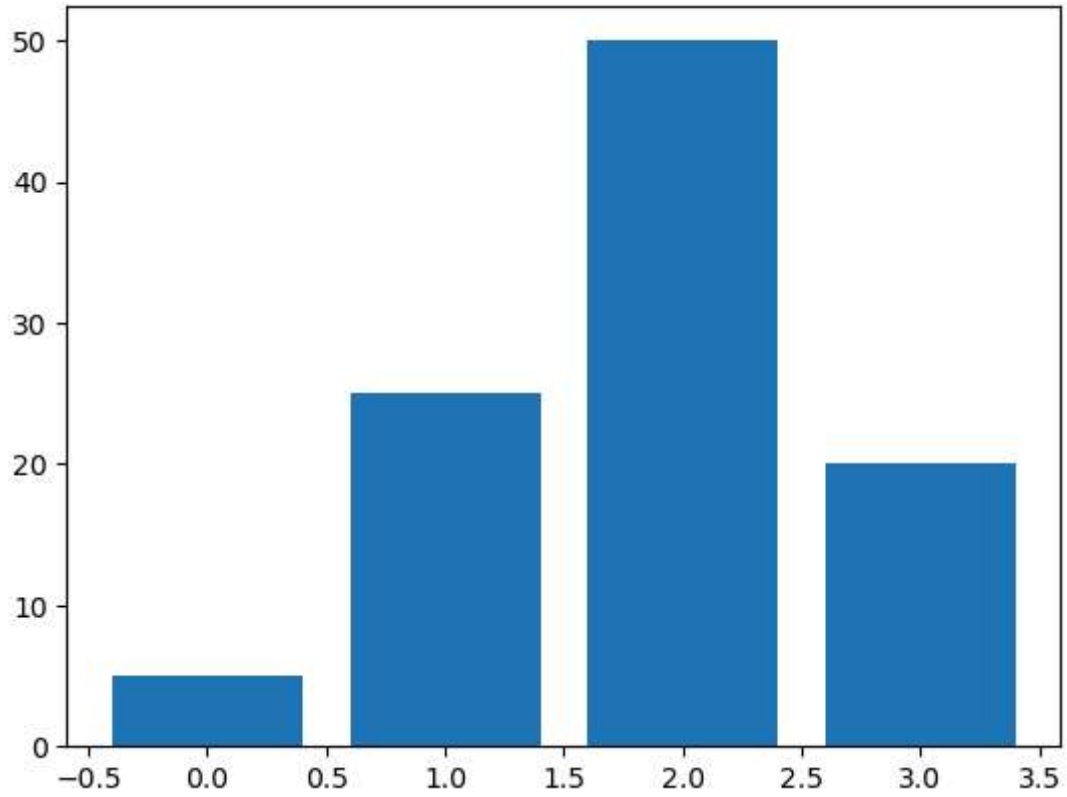
In [75]:
```
x7 = np.linspace(0, 10, 30)

y7 = np.sin(x7)

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

plt.show()
```



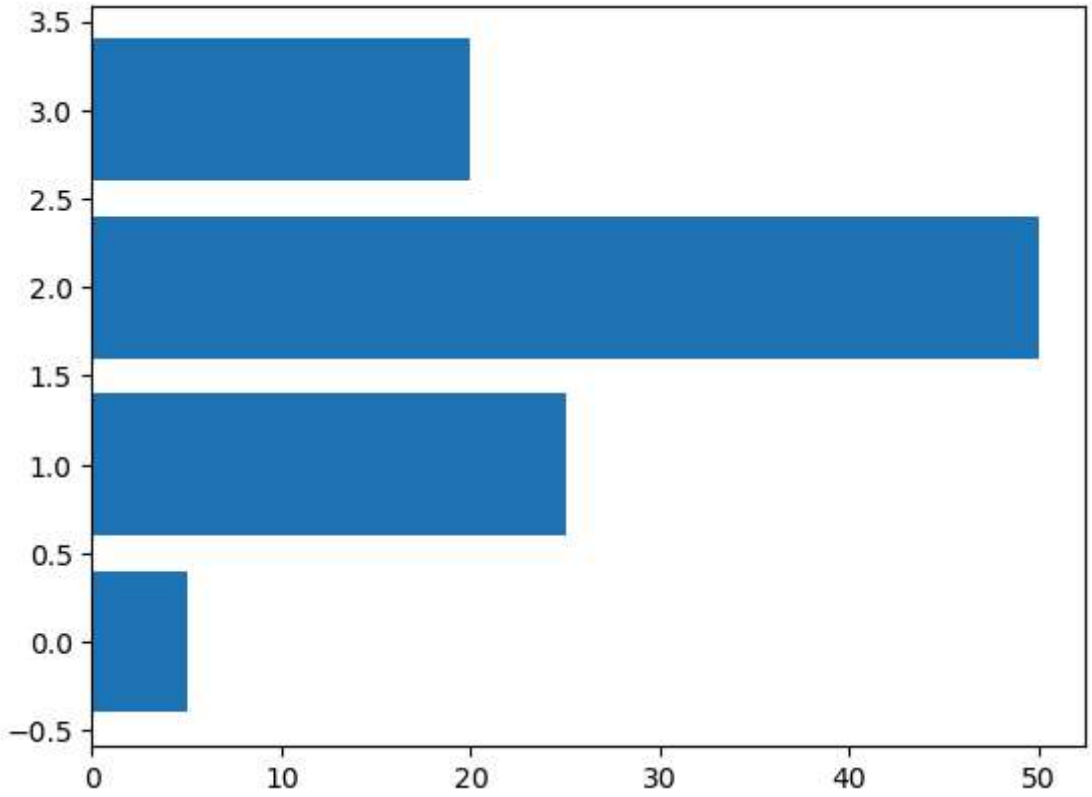
In [77]:
```
data1 = np.random.randn(1000)

plt.hist(data1);

plt.show()
```



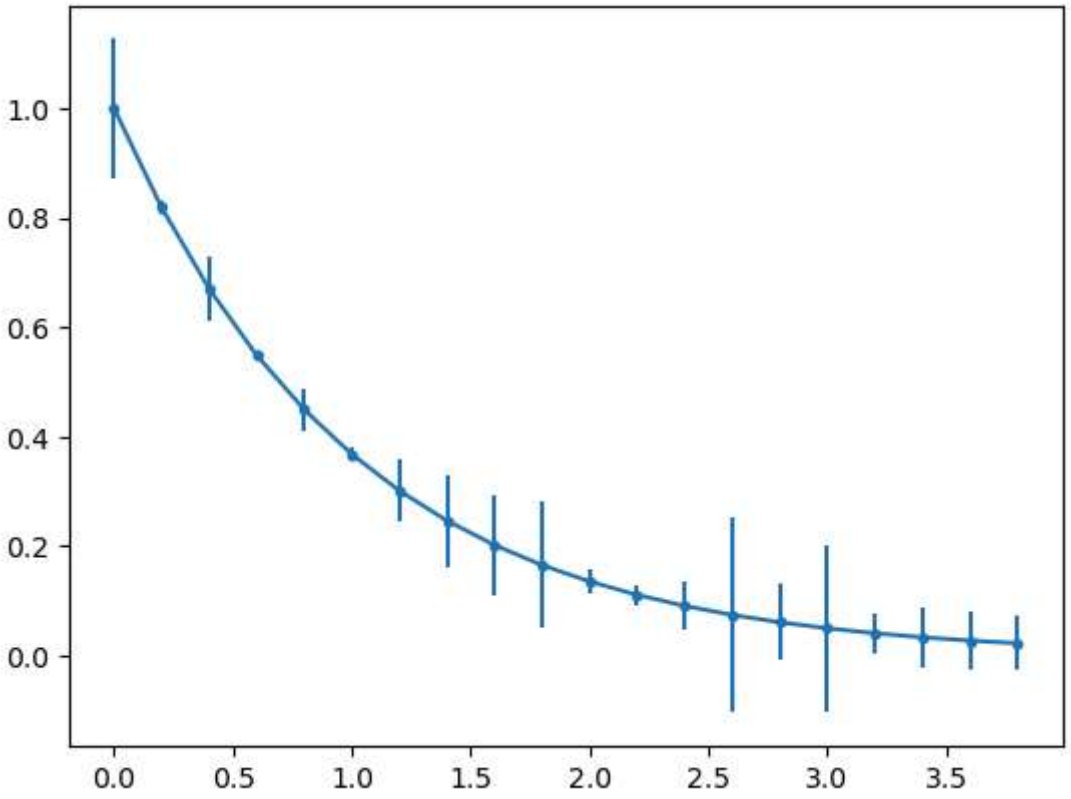
In [79]:
```
data2 = [5. , 25. , 50. , 20.]

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

plt.show()
```



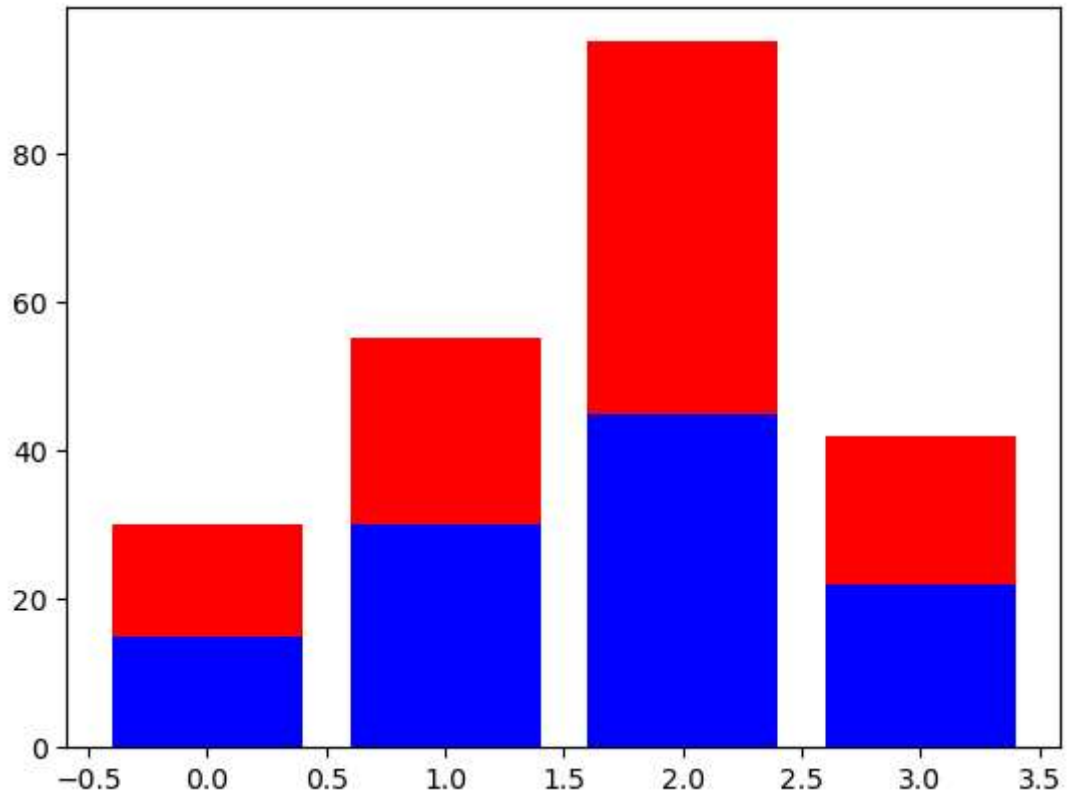
In [83]:
```
data2 = [5. , 25. , 50. , 20.]

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

plt.show()
```



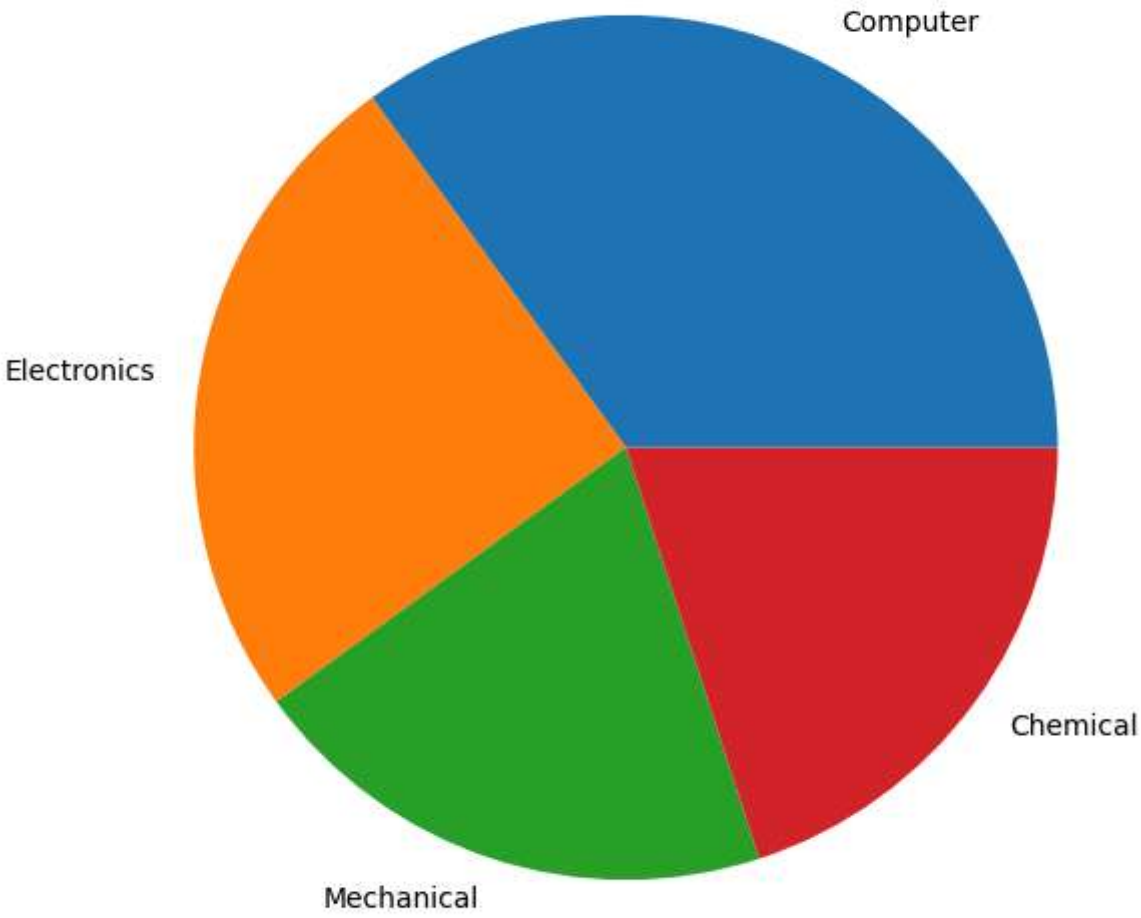
In [85]:
```
x9 = np.arange(0, 4, 0.2)

y9 = np.exp(-x9)

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

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

plt.show();
```



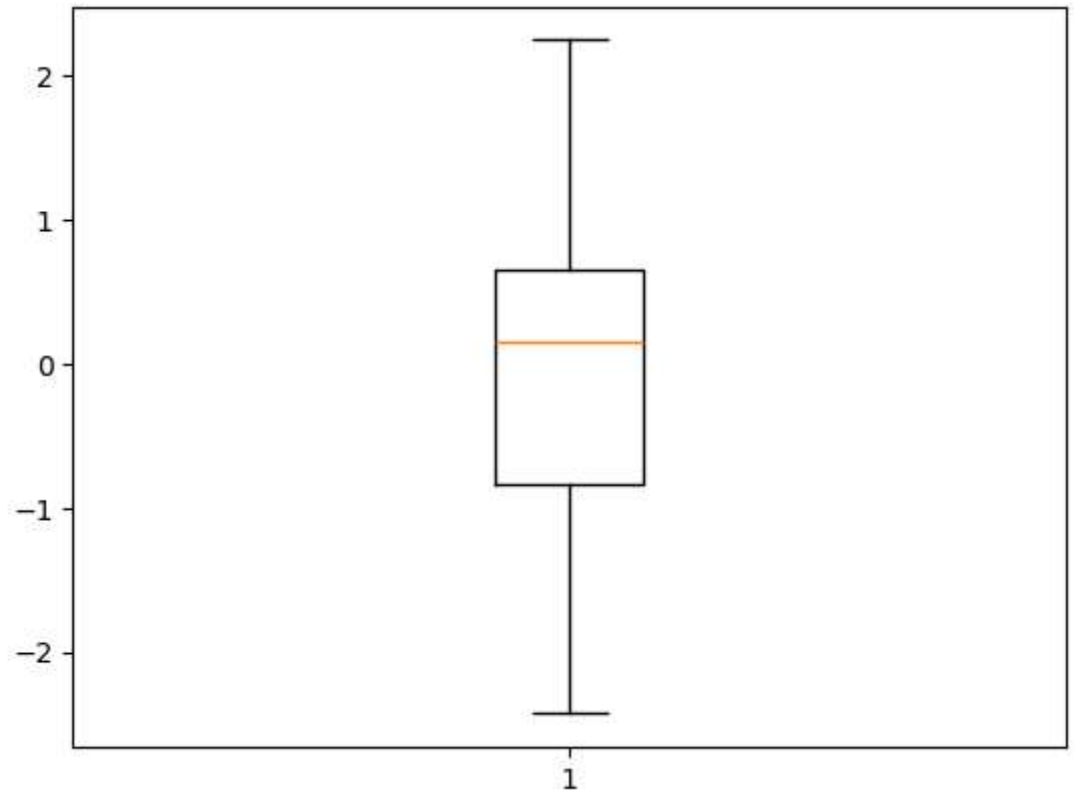
In [87]:
```
A = [15., 30., 45., 22.]

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

z2 = range(4)

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

plt.show()
```



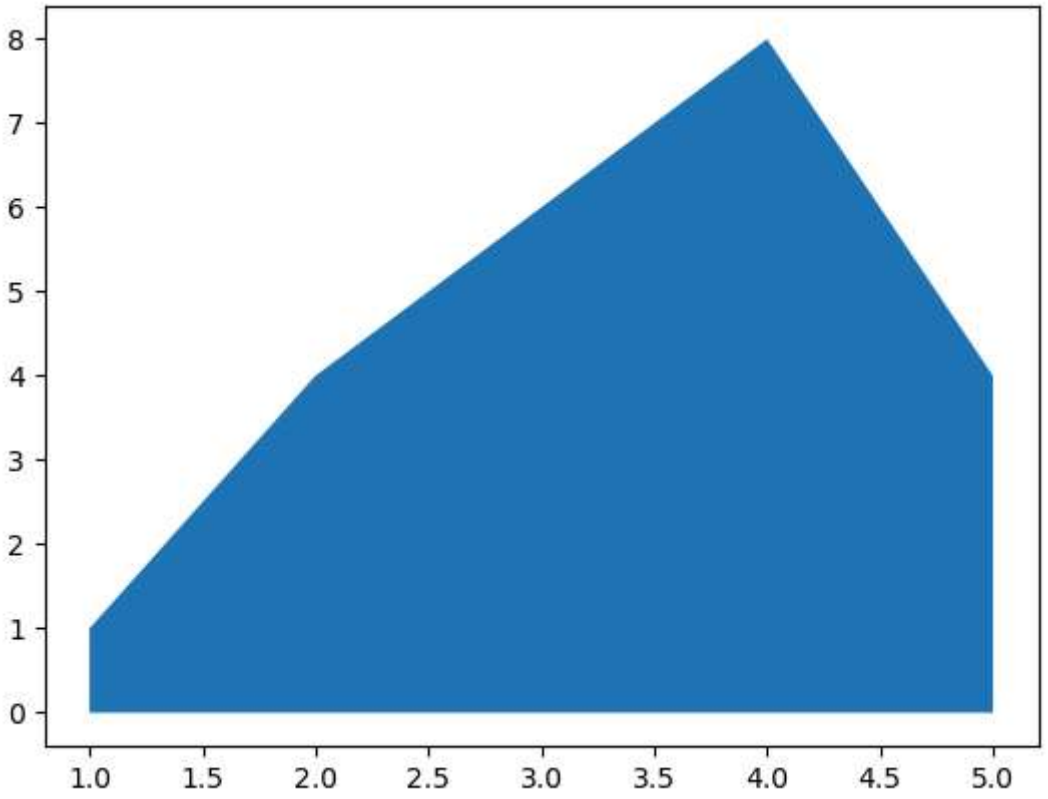
In [89]:
```
plt.figure(figsize=(7,7))

x10 = [35, 25, 20, 20]

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

plt.pie(x10, labels=labels);

plt.show()
```



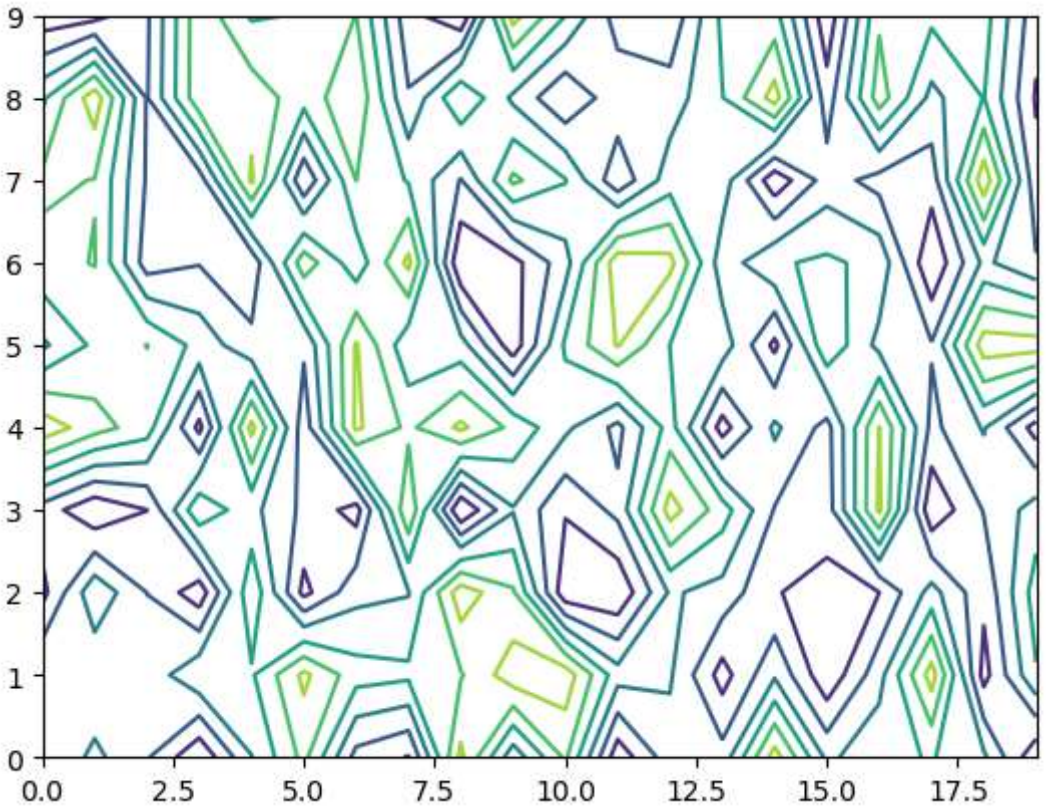
In [91]:
```
data3 = np.random.randn(100)

plt.boxplot(data3)

plt.show()
```



In [93]:
```
# Create some data
x12 = range(1, 6)
y12 = [1, 4, 6, 8, 4]

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



In [95]:
```
# Create a matrix
matrix1 = np.random.rand(10, 20)

cp = plt.contour(matrix1)

plt.show()
```



In [97]:
```
# View list of all available styles

print(plt.style.available)
```

```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'bmh', 'classic', 'dark_backgroun
d', 'fast', 'fivethirtyeight', 'ggplot', 'grayscale', 'seaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblin
d', 'seaborn-v0_8-dark', 'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-deep', 'seaborn-v0_8-mut
ed', 'seaborn-v0_8-notebook', 'seaborn-v0_8-paper', 'seaborn-v0_8-pastel', 'seaborn-v0_8-poster', 'seaborn-v0_8-tal
k', 'seaborn-v0_8-ticks', 'seaborn-v0_8-white', 'seaborn-v0_8-whitegrid', 'tableau-colorblind10']
```

In [ ]: