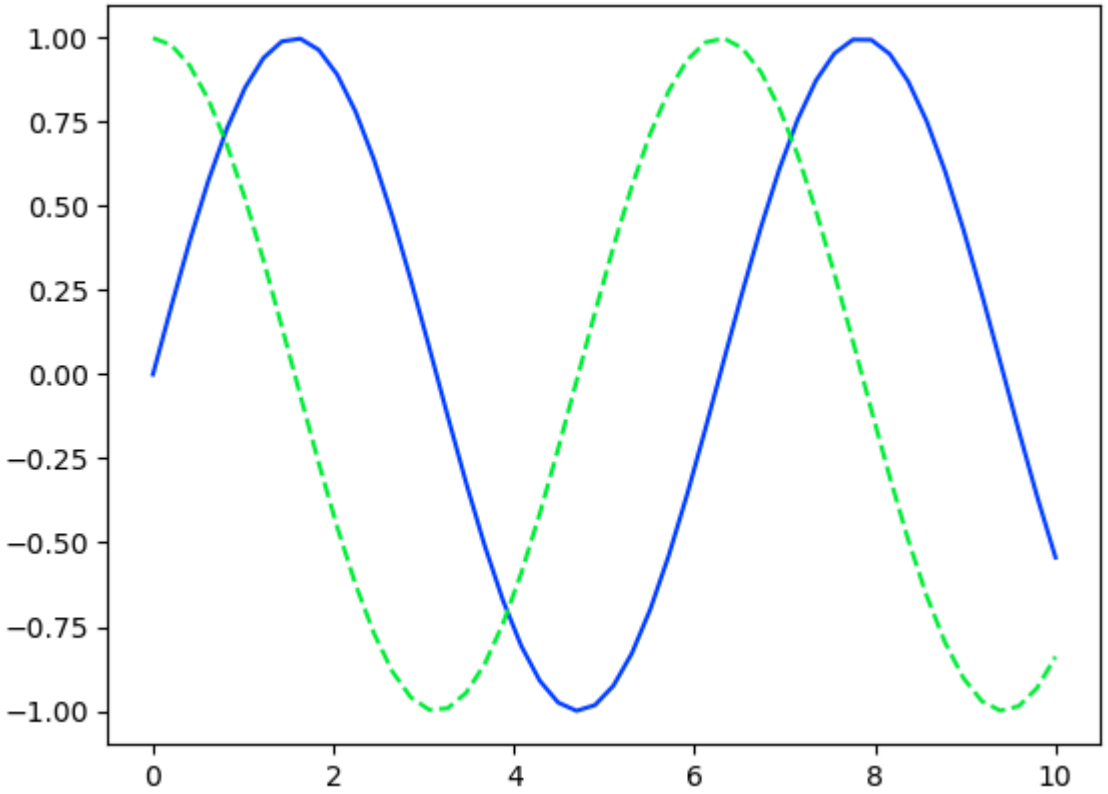
Matplotlib

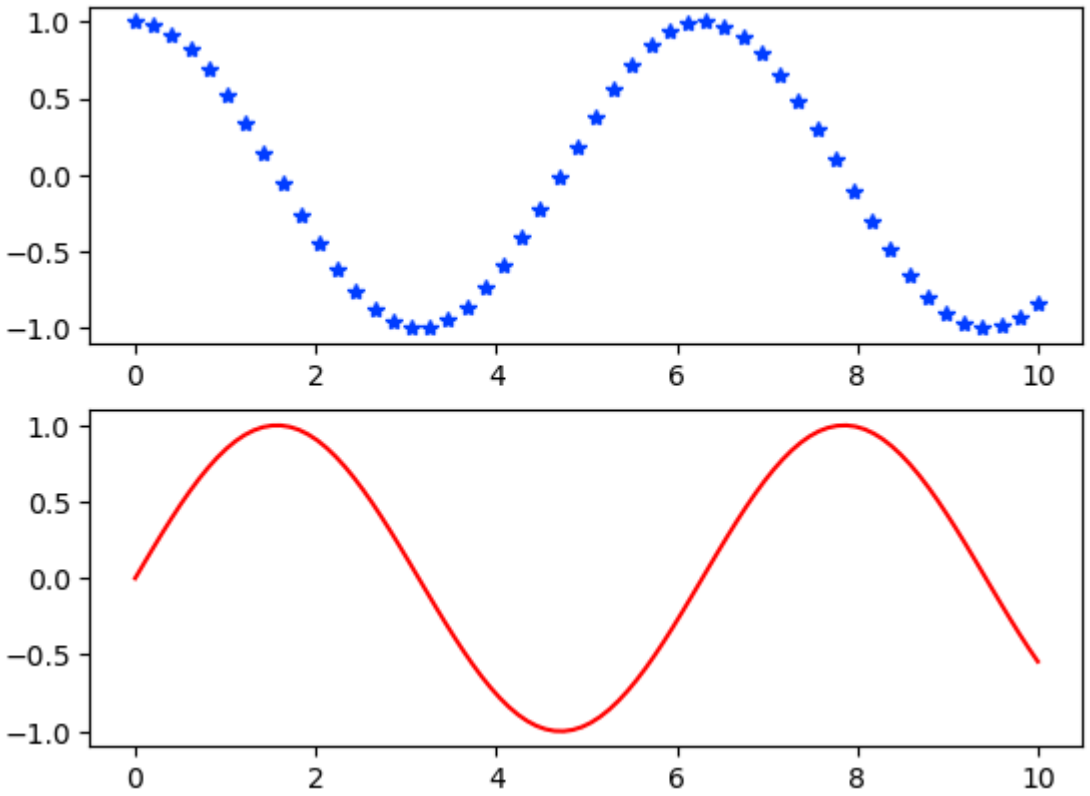
In [132…
```
# import libraries
import numpy as np
import matplotlib.pyplot as plt
```

In [133…
```
# Displaying Plots in Matplotlib
%matplotlib inline
x1 = np.linspace(0, 10, 50)

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



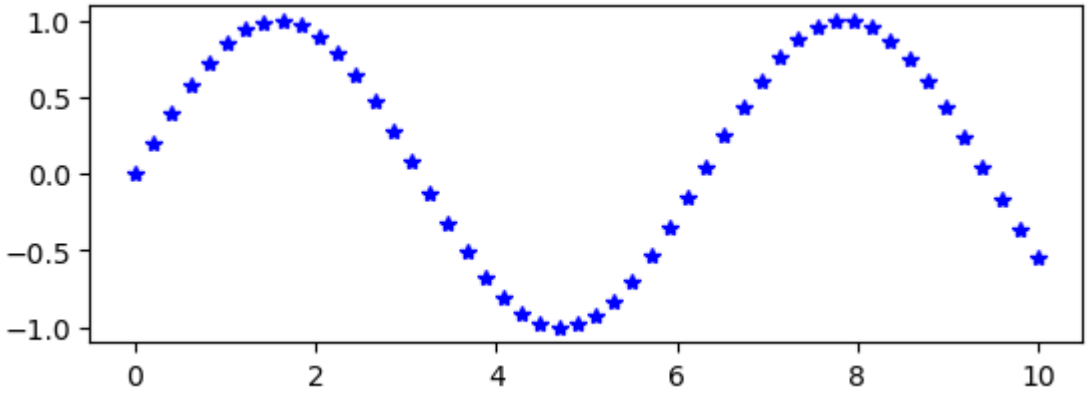
In [134…

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


plt.subplot(2, 1, 1)   # (rows, columns, panel number)
plt.plot(x1, np.cos(x1), '*')
x1 = np.linspace(0, 10, 100)
plt.subplot(2, 1, 2)
plt.plot(x1, np.sin(x1), 'r-')

plt.show()
```



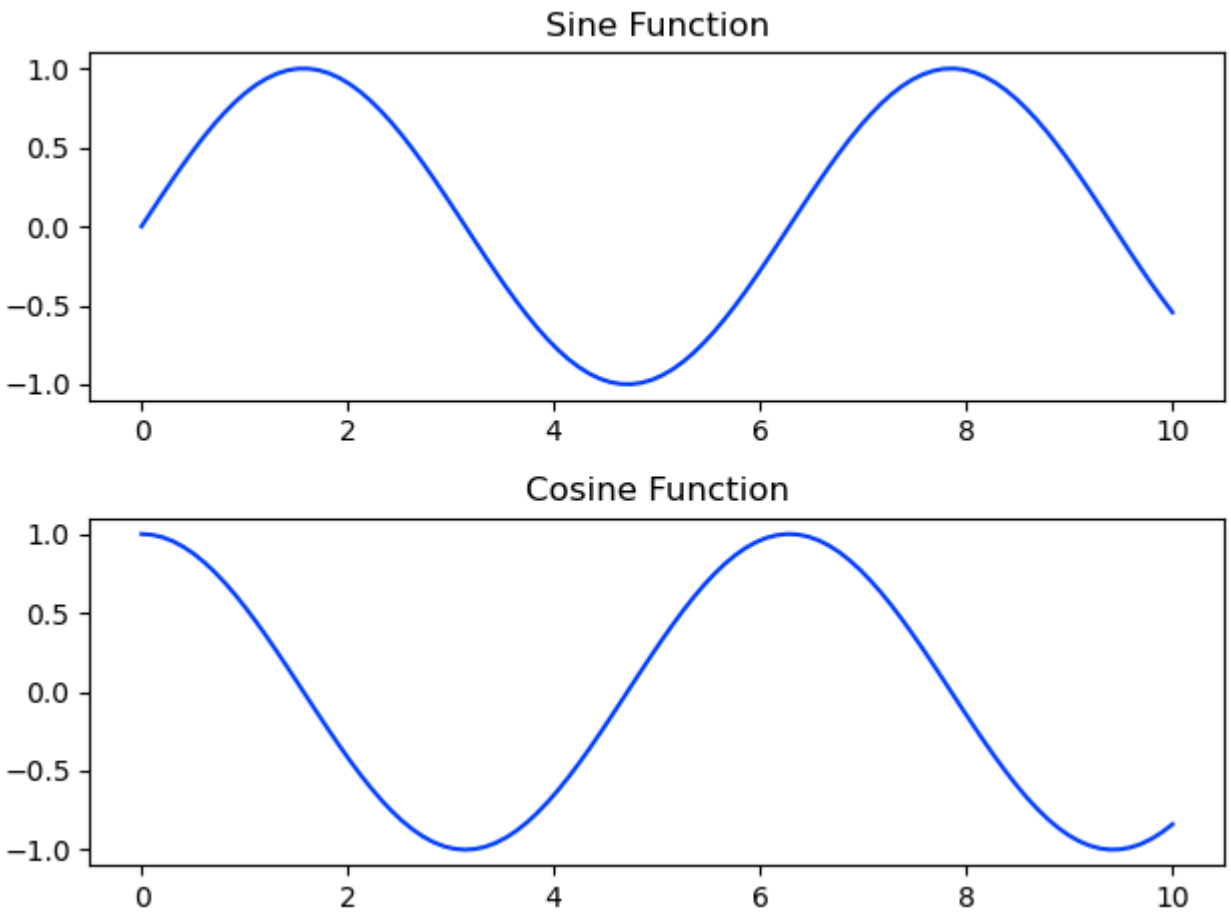
In [142…

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


plt.subplot(2, 1, 1)   # (rows, columns, panel number)
x1 = np.linspace(0, 10,)
plt.subplot(2, 1, 1)
plt.plot(x1, np.sin(x1), 'b''*')

plt.show()
```



In [144…

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

# Create data
x1 = np.linspace(0, 10, 100)

# Create a new figure
plt.figure()

# Create the first of two panels (top)
plt.subplot(2, 1, 1)   # (rows, columns, panel number)
plt.plot(x1, np.sin(x1))
plt.title('Sine Function')

# Create the second of two panels (bottom)
plt.subplot(2, 1, 2)
plt.plot(x1, np.cos(x1))
plt.title('Cosine Function')

# Add some spacing between the two plots
plt.tight_layout()

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



In [63]:
```
# get current figure information

print(plt.gcf())
```

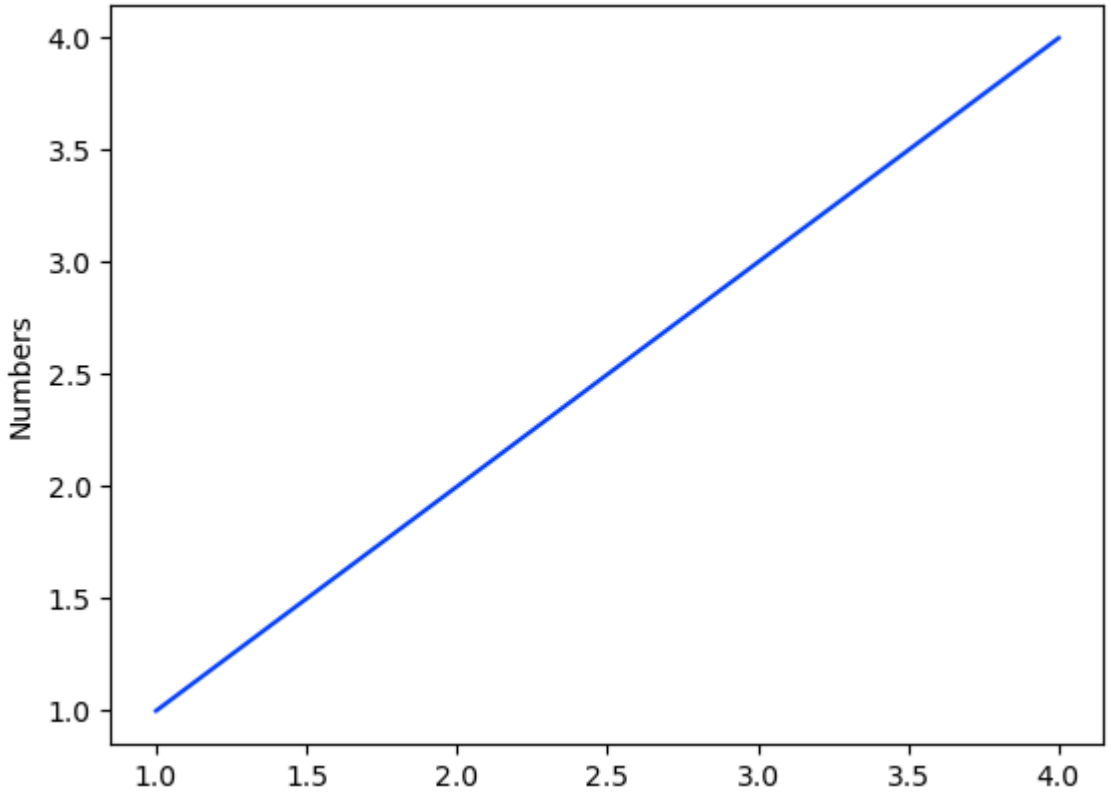
```
Figure(640x480)
```

In [64]:
```
# get current axis information

print(plt.gca())
```

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

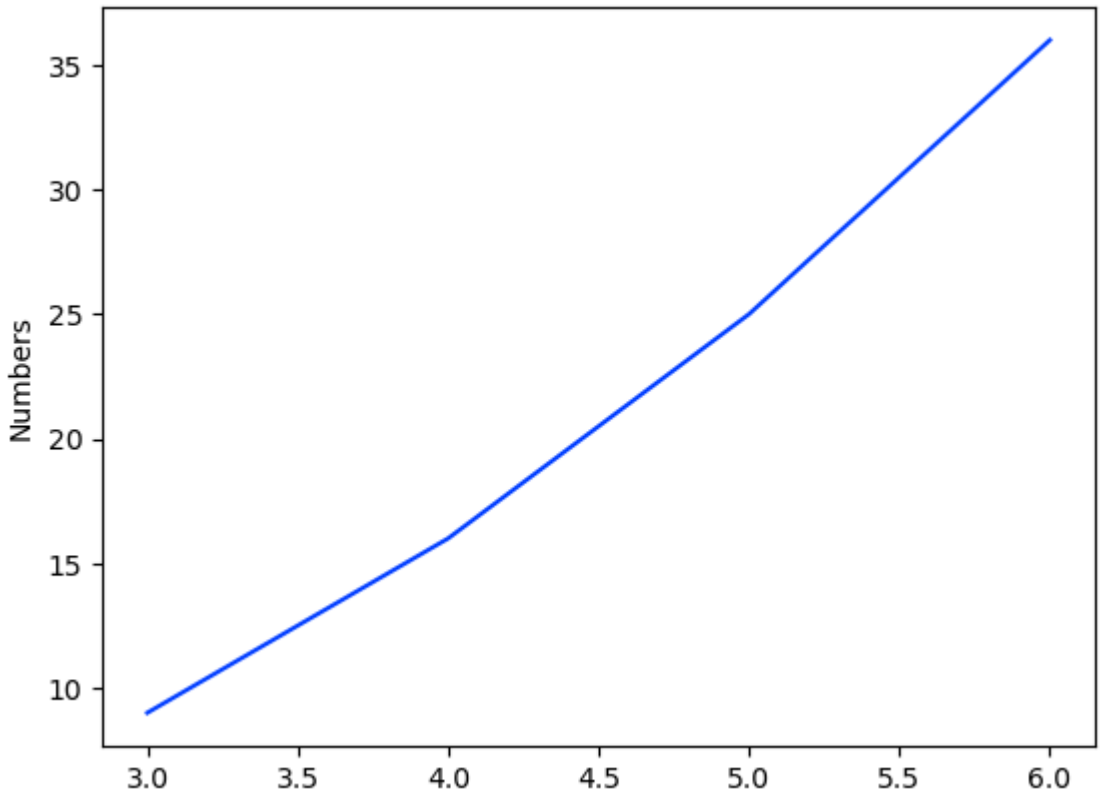
In [151…

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

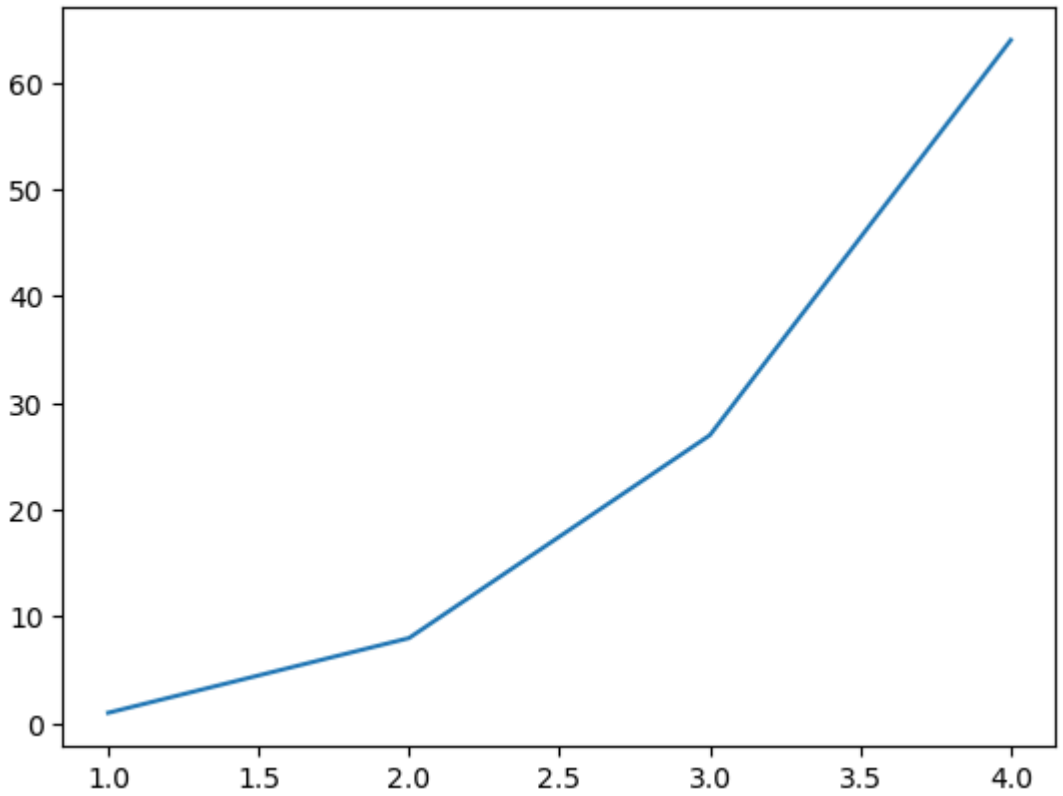


In [145…

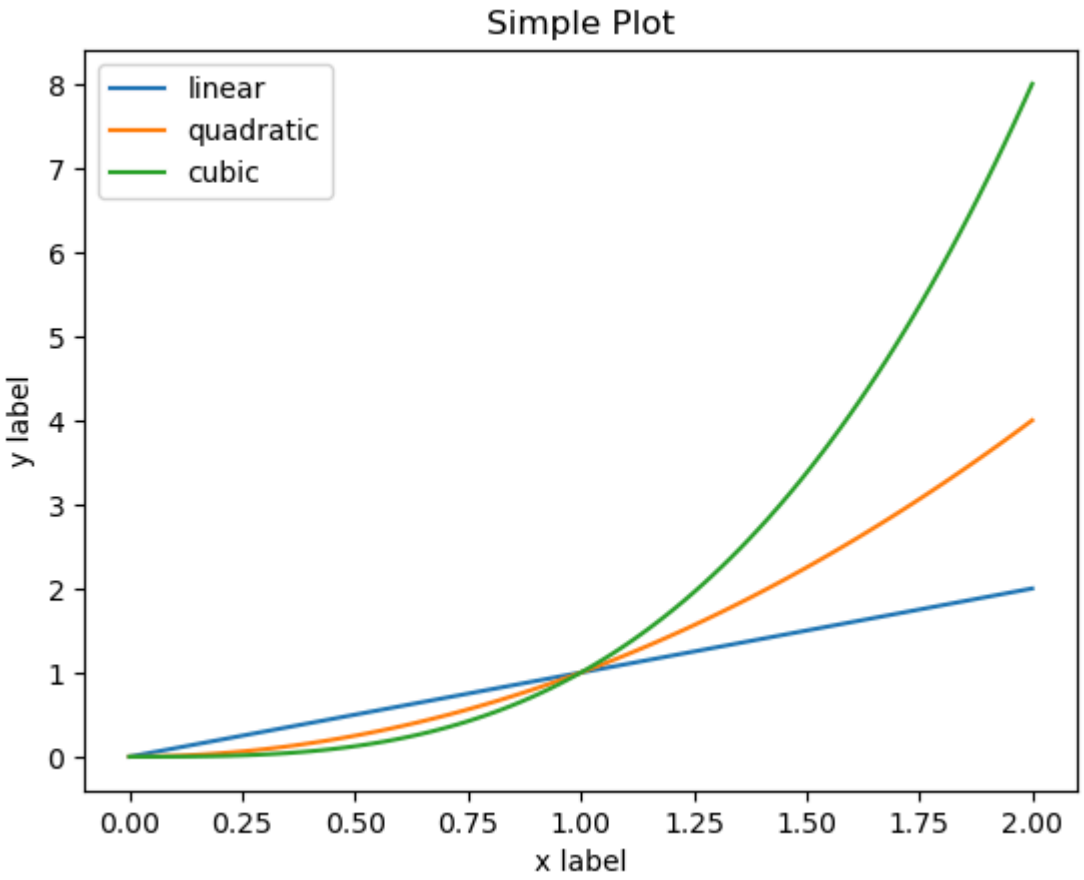
```
plt.plot([3,4,5,6], [9,16,25,36])
plt.ylabel('Numbers')
plt.show()
```



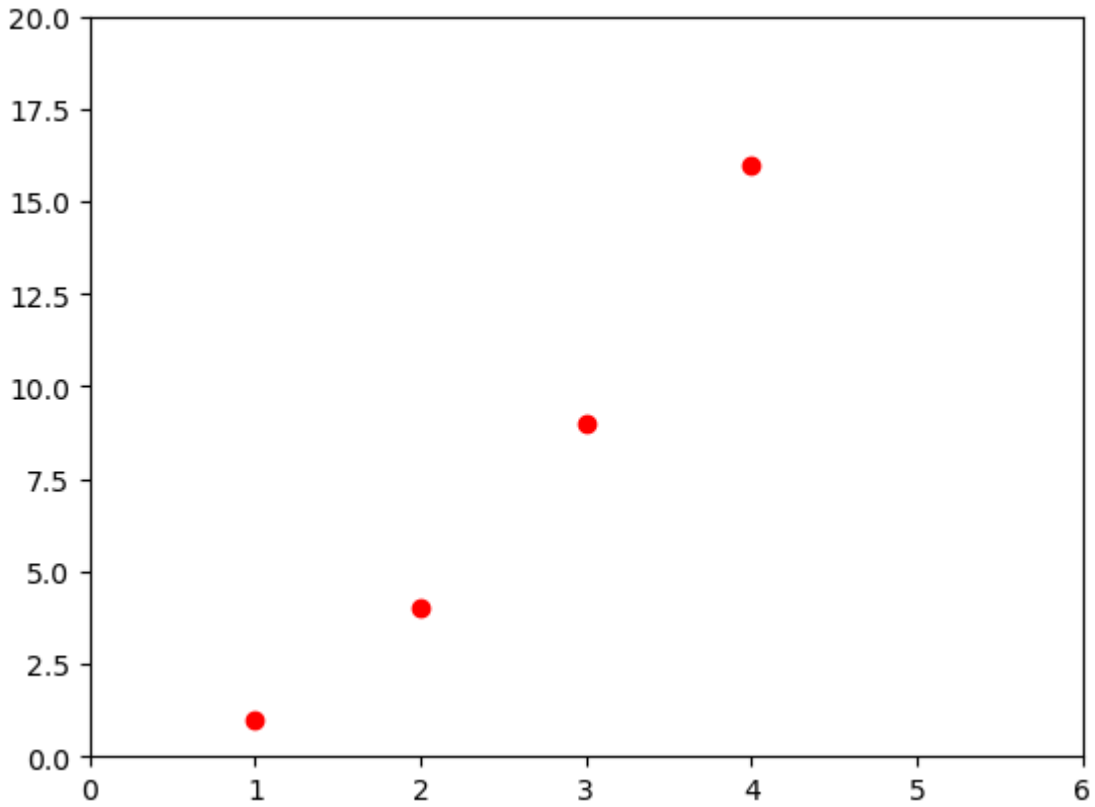
In [66]:
```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4], [1, 8, 27, 64])
plt.show()
```



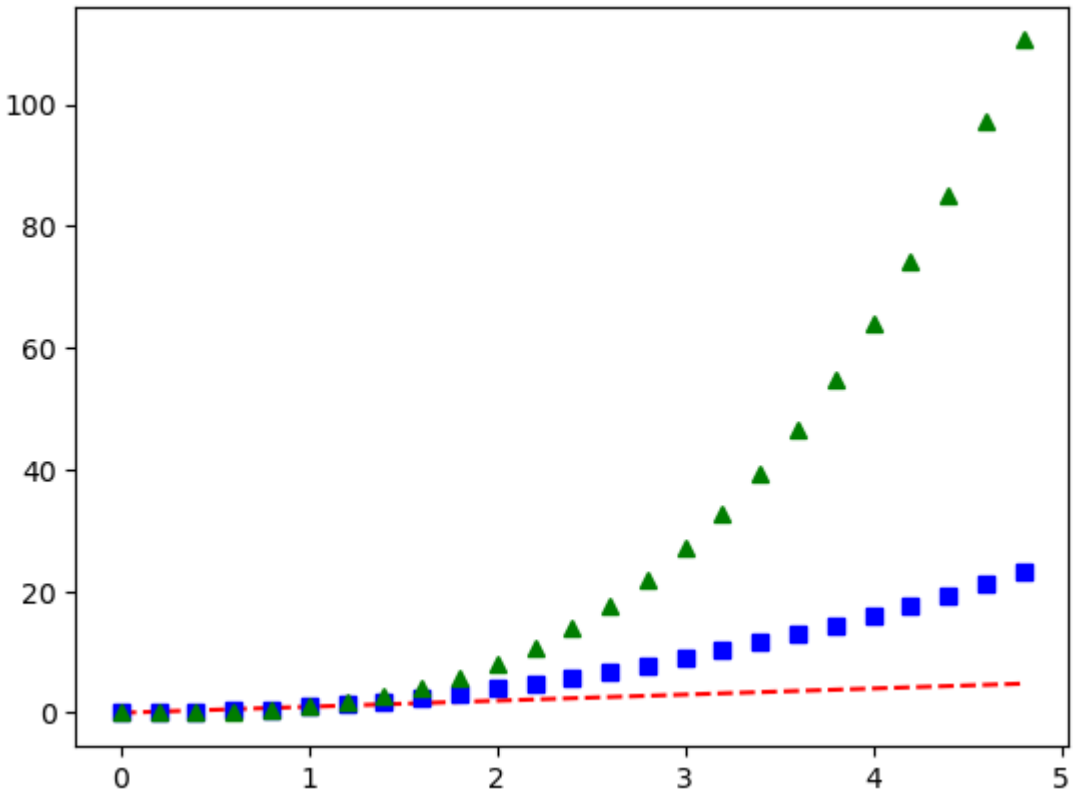
In [67]:
```
x = np.linspace(0, 2, 100)

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 [68]:
```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
plt.axis([0, 6, 0, 20])
plt.show()
```

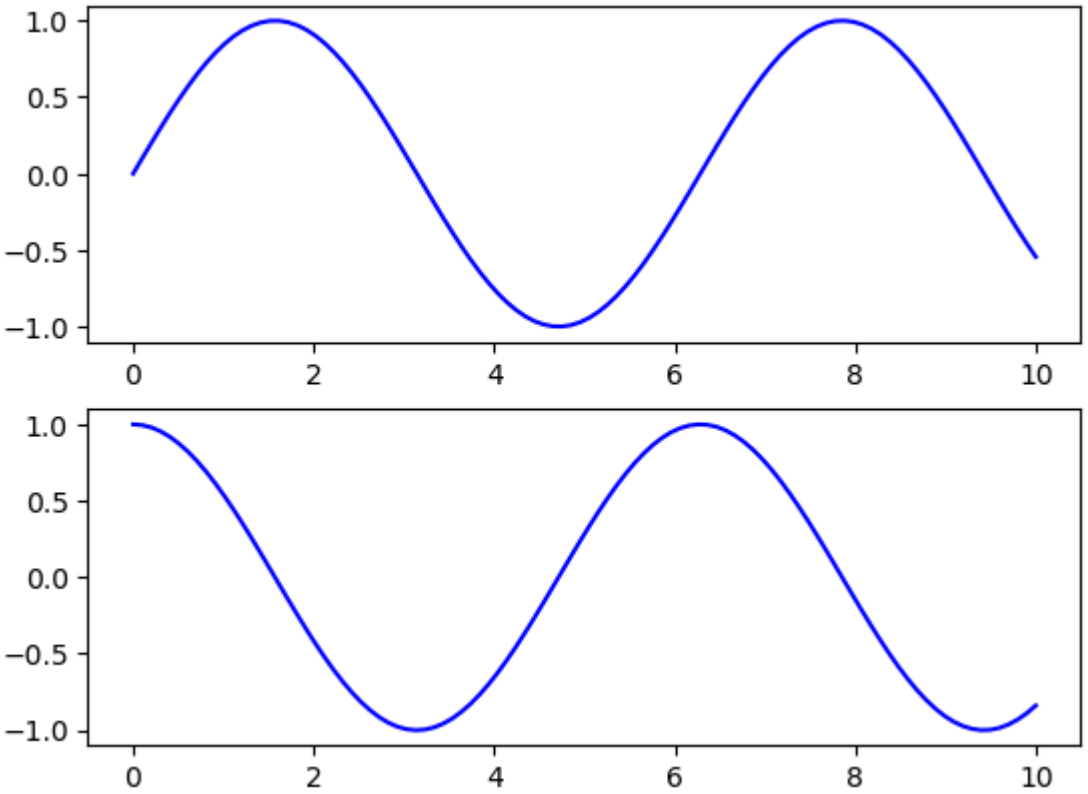


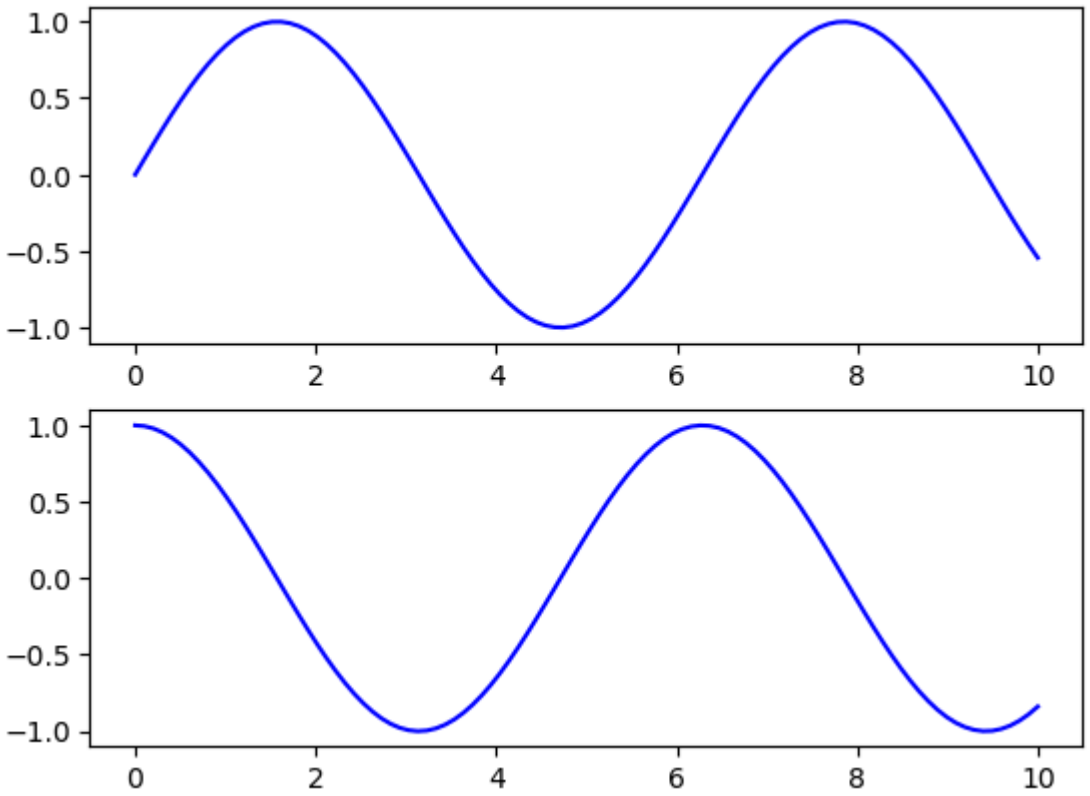
In [69]:
```
# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)

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

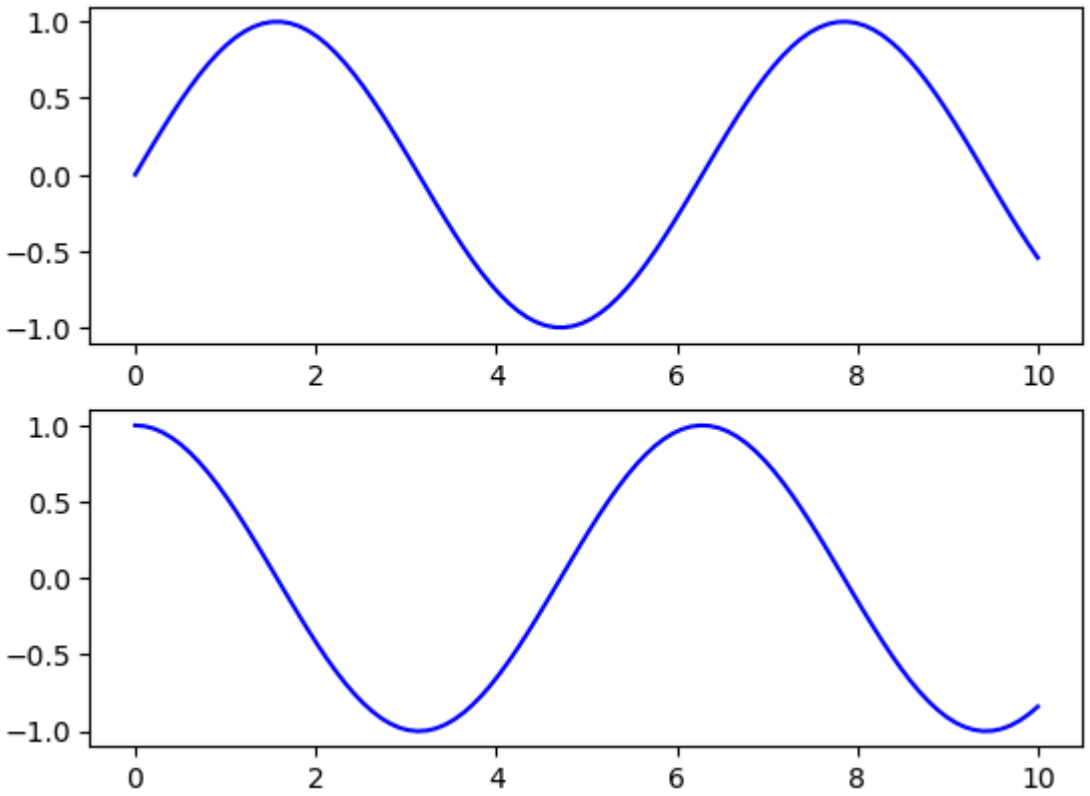


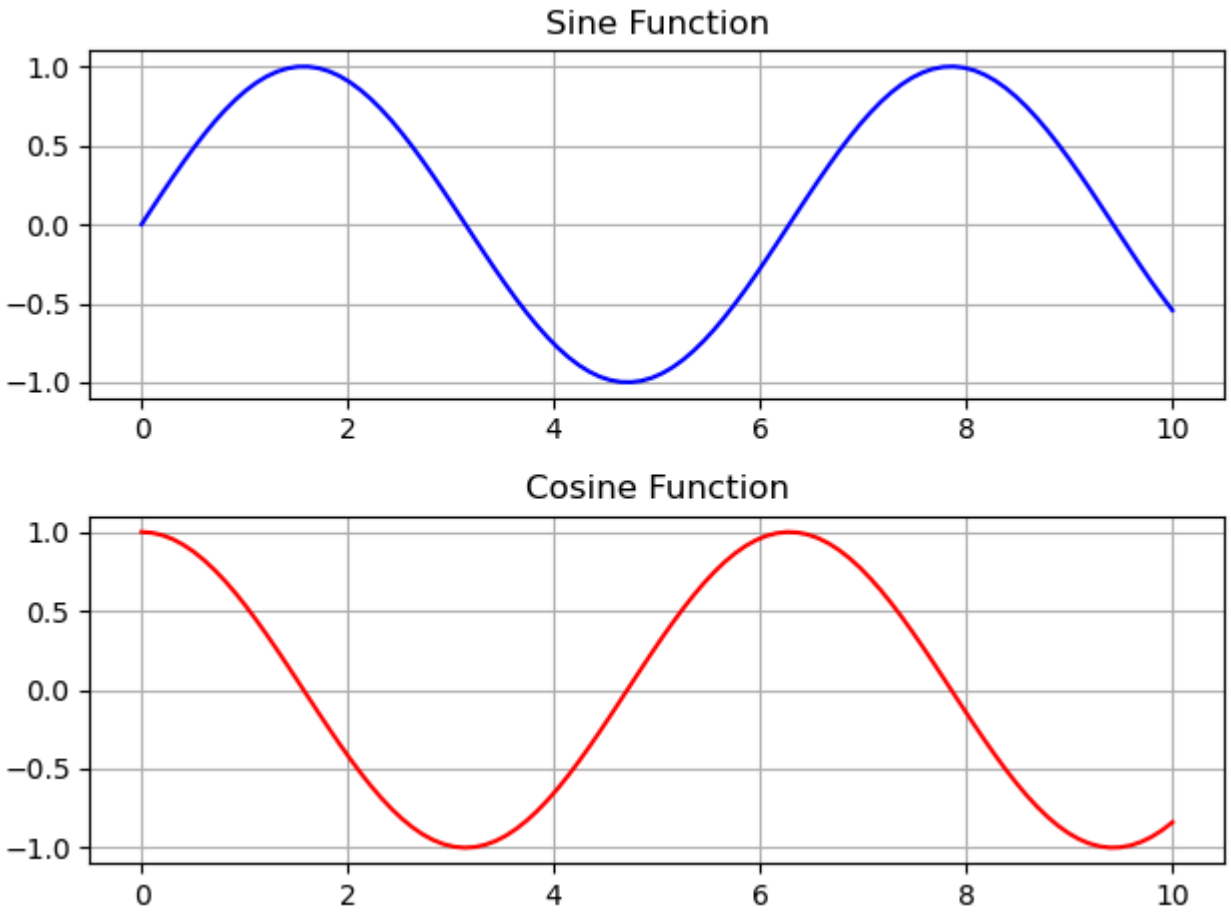
In [73]:
```
import numpy as np
import matplotlib.pyplot as plt

# Create data
x1 = np.linspace(0, 10, 100)

# Create a grid of 2 subplots (vertically stacked)
fig, ax = plt.subplots(2)

# Plot sine and cosine
ax[0].plot(x1, np.sin(x1), 'b-')
ax[0].set_title('Sine Function')
ax[0].grid(True)

ax[1].plot(x1, np.cos(x1), 'r-')
```

```
ax[1].set_title('Cosine Function')
ax[1].grid(True)

# Adjust layout so titles and labels don’t overlap
plt.tight_layout()

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



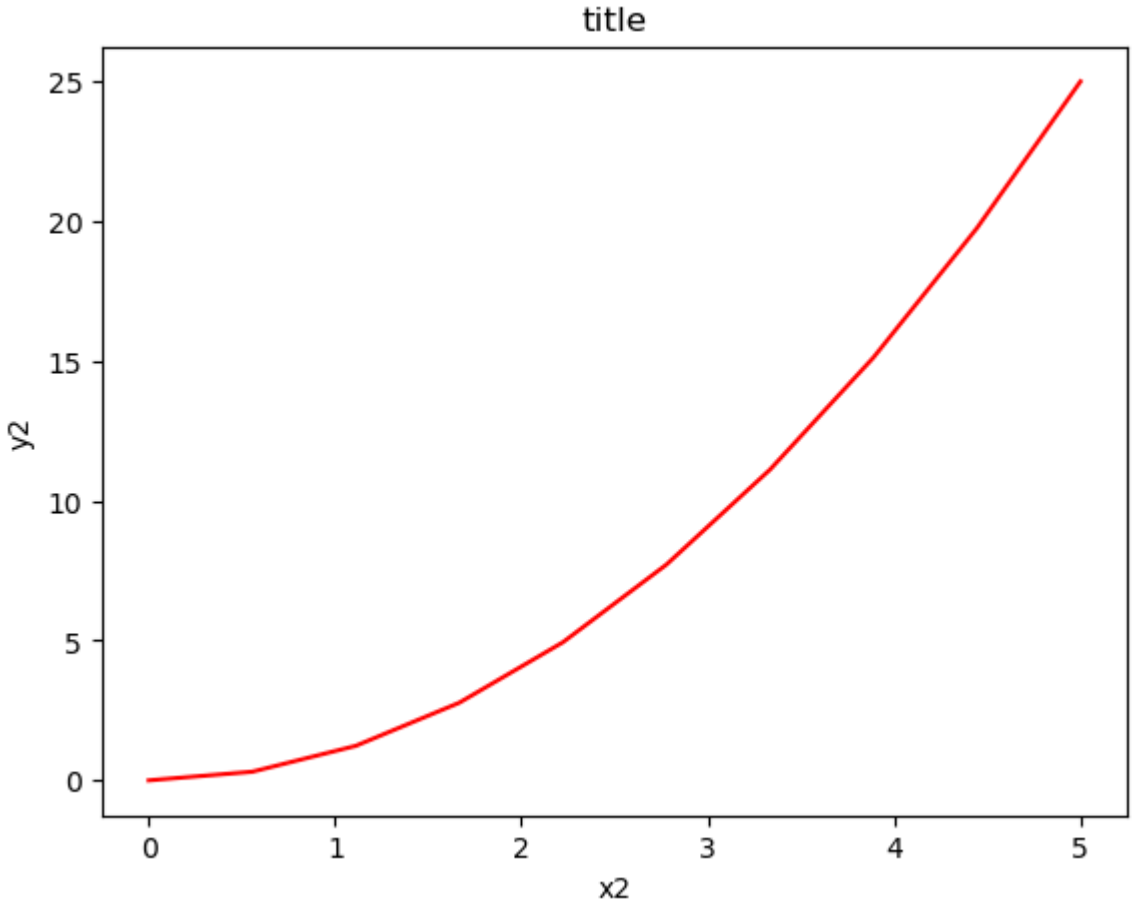

1.0
0.5
0.0
−0.5
−1.0
0
2
4
6
8
10
1.0
0.5
0.0
−0.5
−1.0
0
2
4
6
8
10

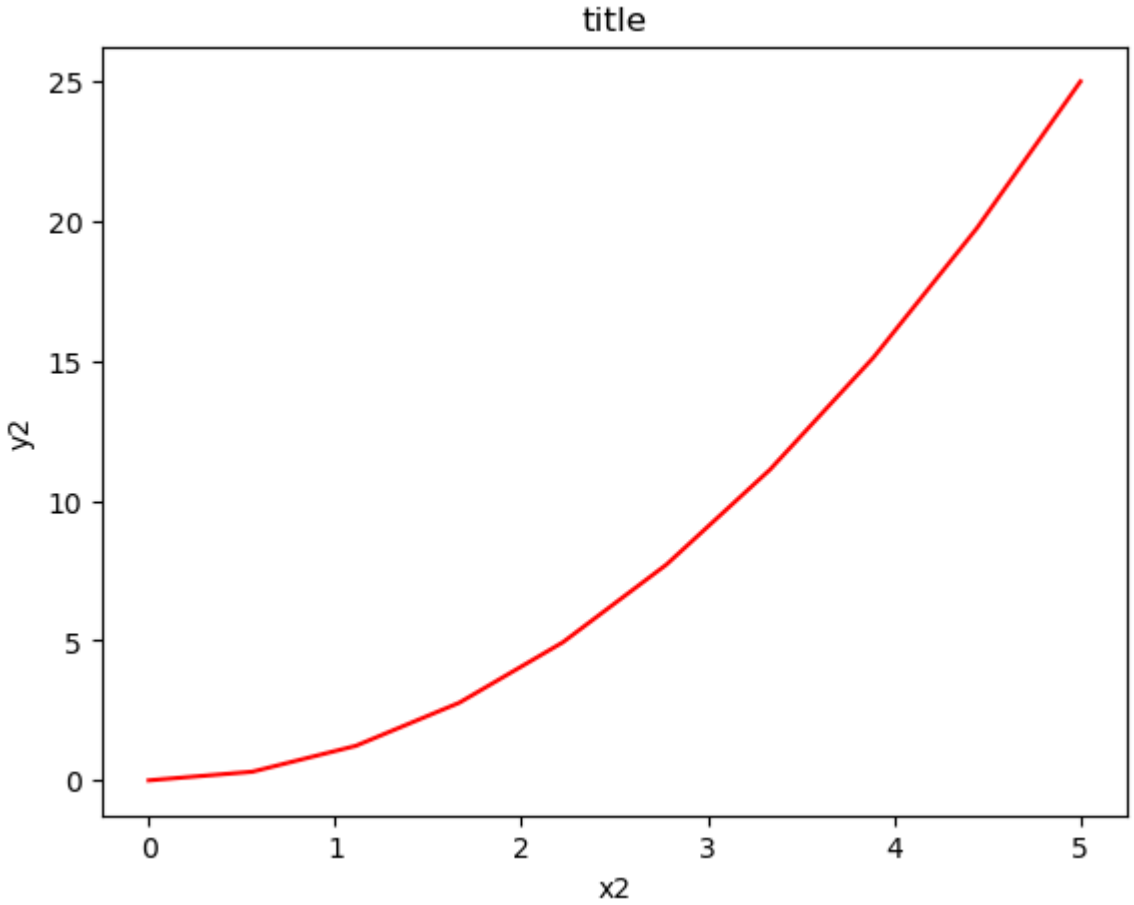

1.0
0.5
0.0
−0.5
−1.0
0
2
4
6
8
10
1.0
0.5
0.0
−0.5
−1.0
0
2
4
6
8
10

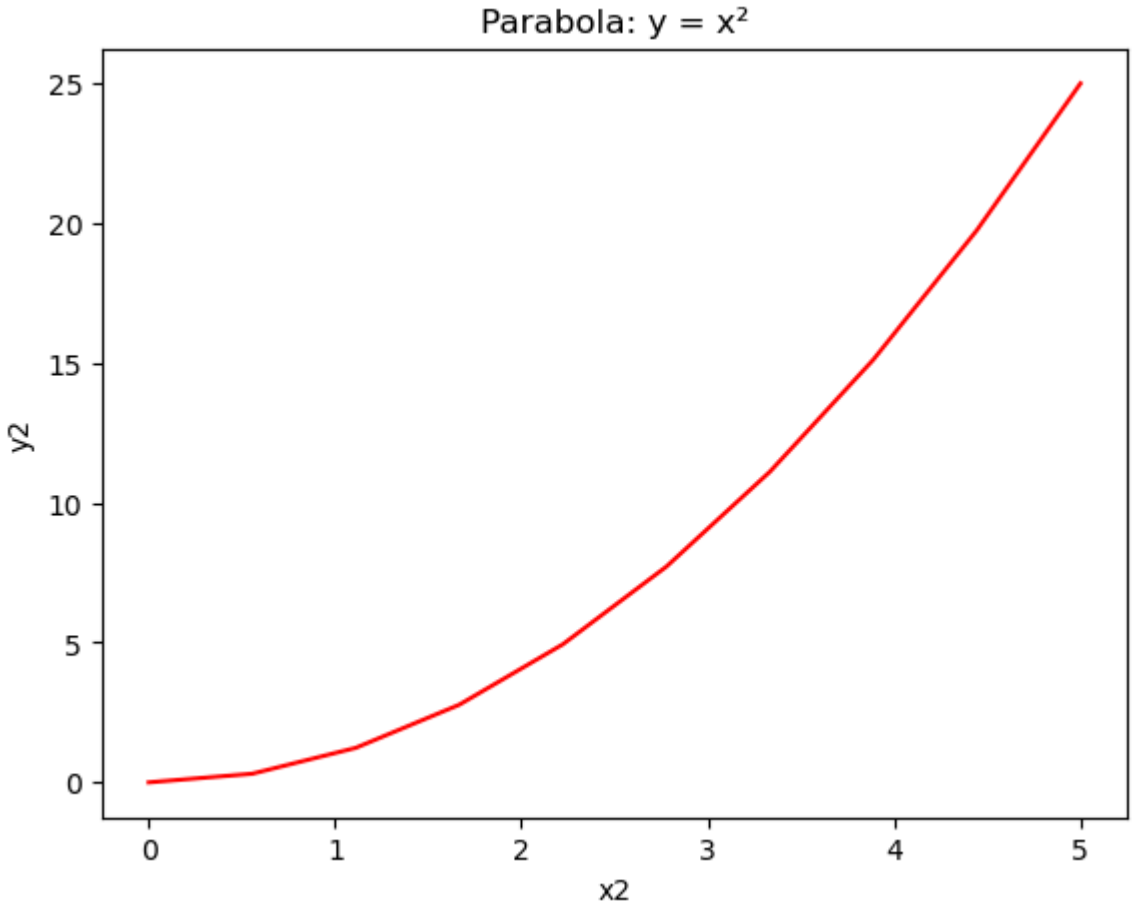


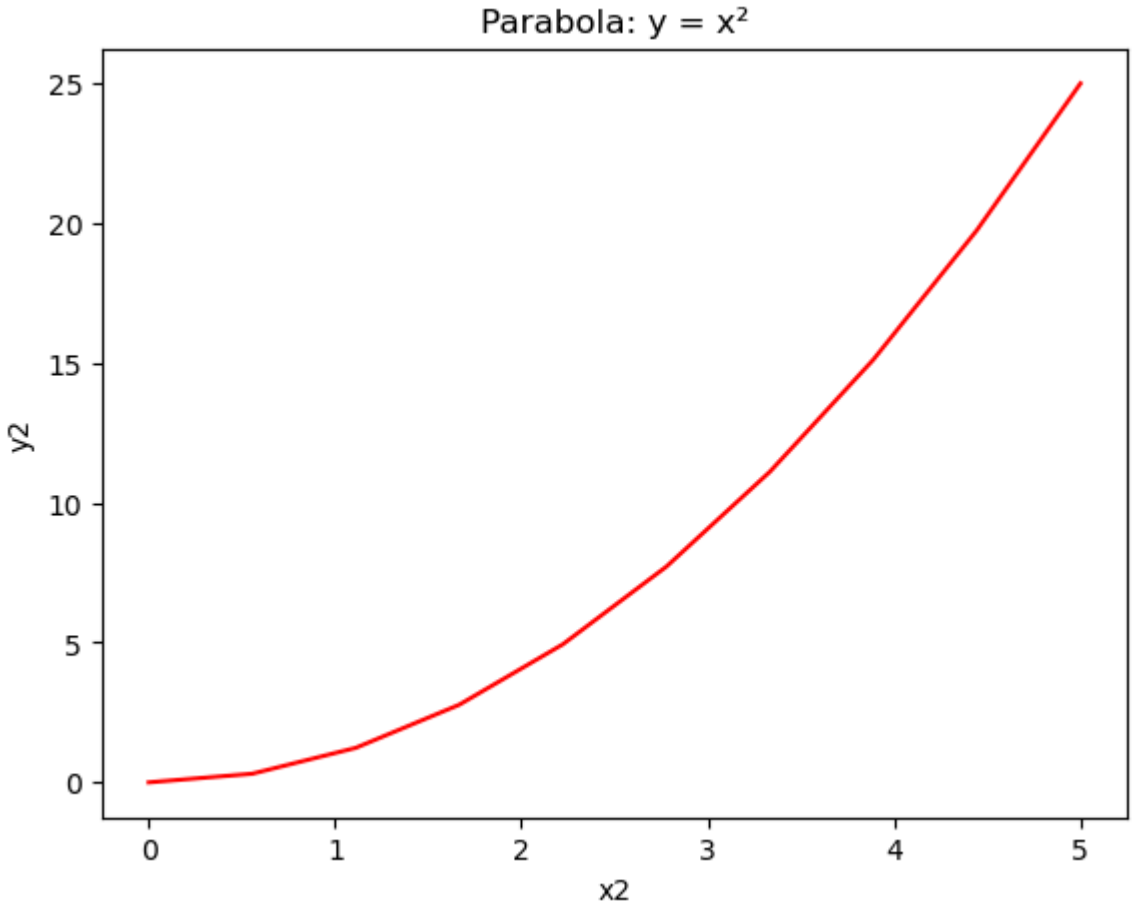
In [78]:
```
# Make sure these two lines are at the top of your notebook
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

# Create a new figure
fig = plt.figure()

# Create data
x2 = np.linspace(0, 5, 10)
y2 = x2 ** 2
```

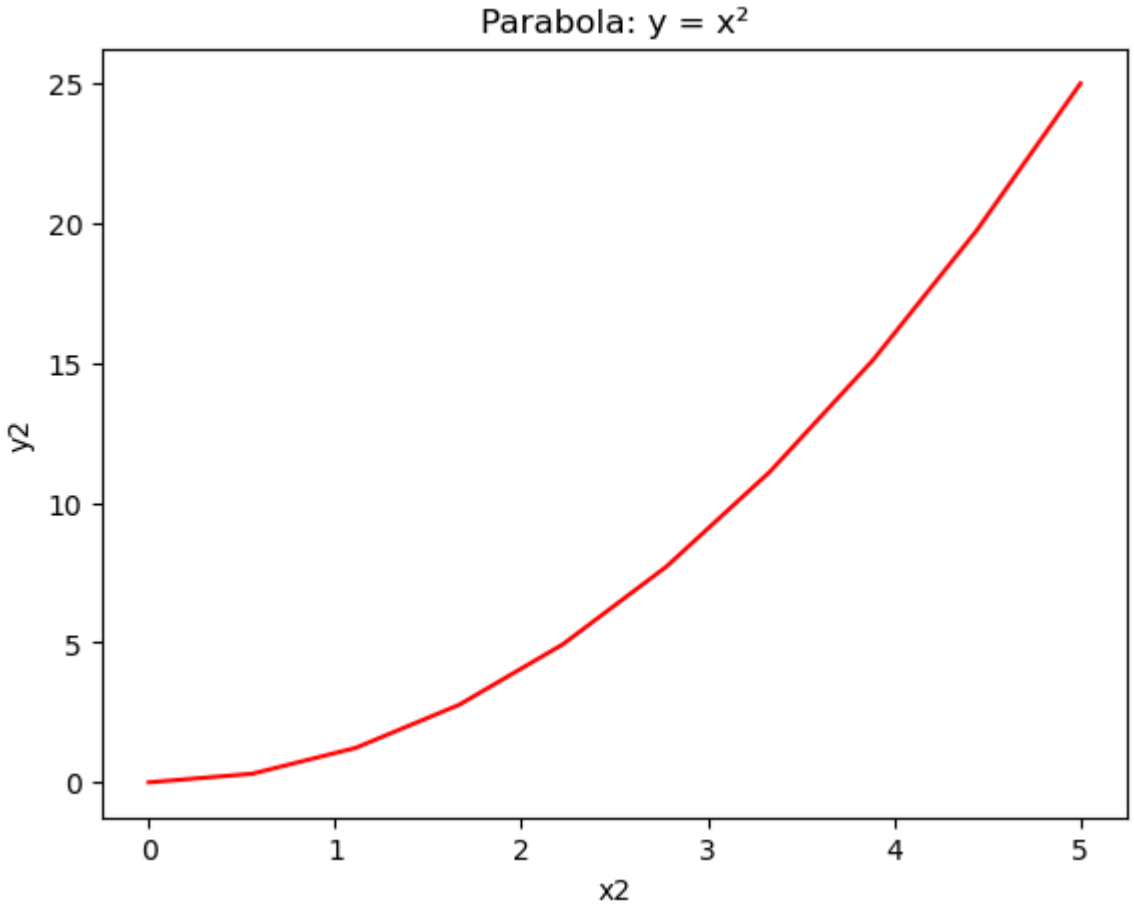
```
# Add axes to the figure: [left, bottom, width, height]
axes = fig.add_axes([0.1, 0.1, 0.8, 0.8])

# Plot the data
axes.plot(x2, y2, 'r')

# Set labels and title
axes.set_xlabel('x2')
axes.set_ylabel('y2')
axes.set_title('Parabola: y = x²')

# Show the plot (optional in Jupyter, but safe to include)
plt.show()
```


title
25
20
15
10
5
0
y2
0
1
2
3
4
5
x2


title
25
20
15
10
5
0
y2
0
1
2
3
4
5
x2


Parabola: y = x²
y2
x2
0
5
10
15
20
25
0
1
2
3
4
5


Parabola: y = x²
y2
x2
0
5
10
15
20
25
0
1
2
3
4
5

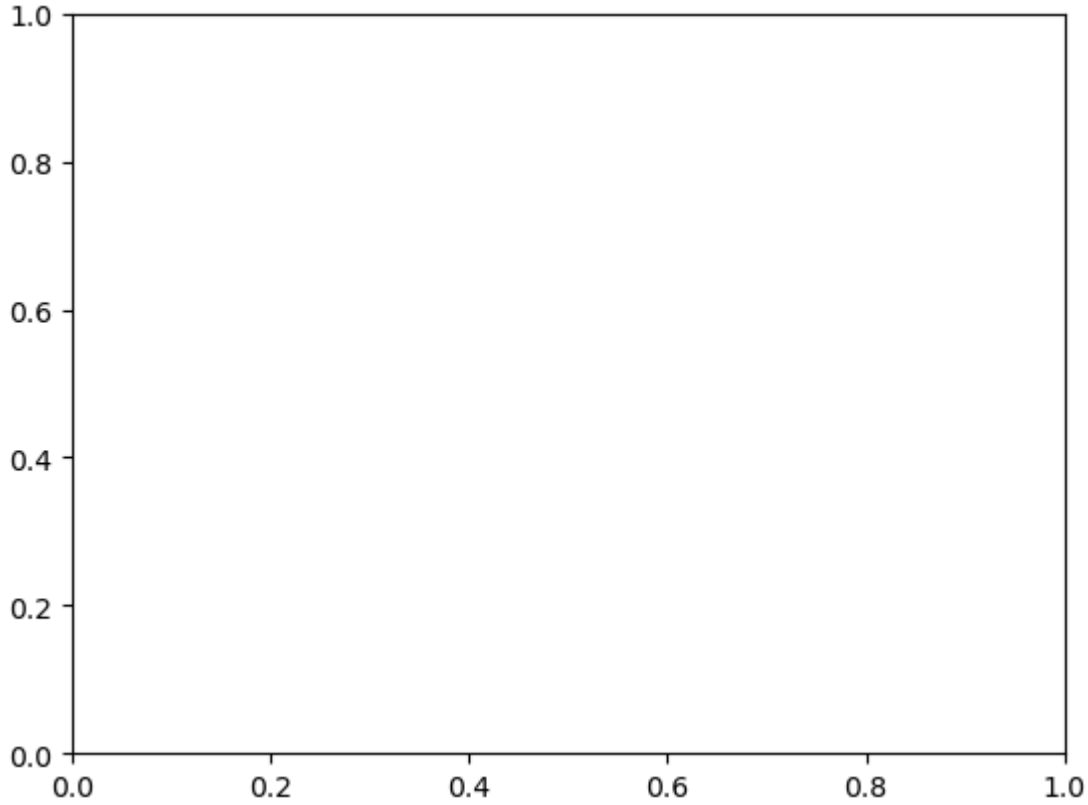


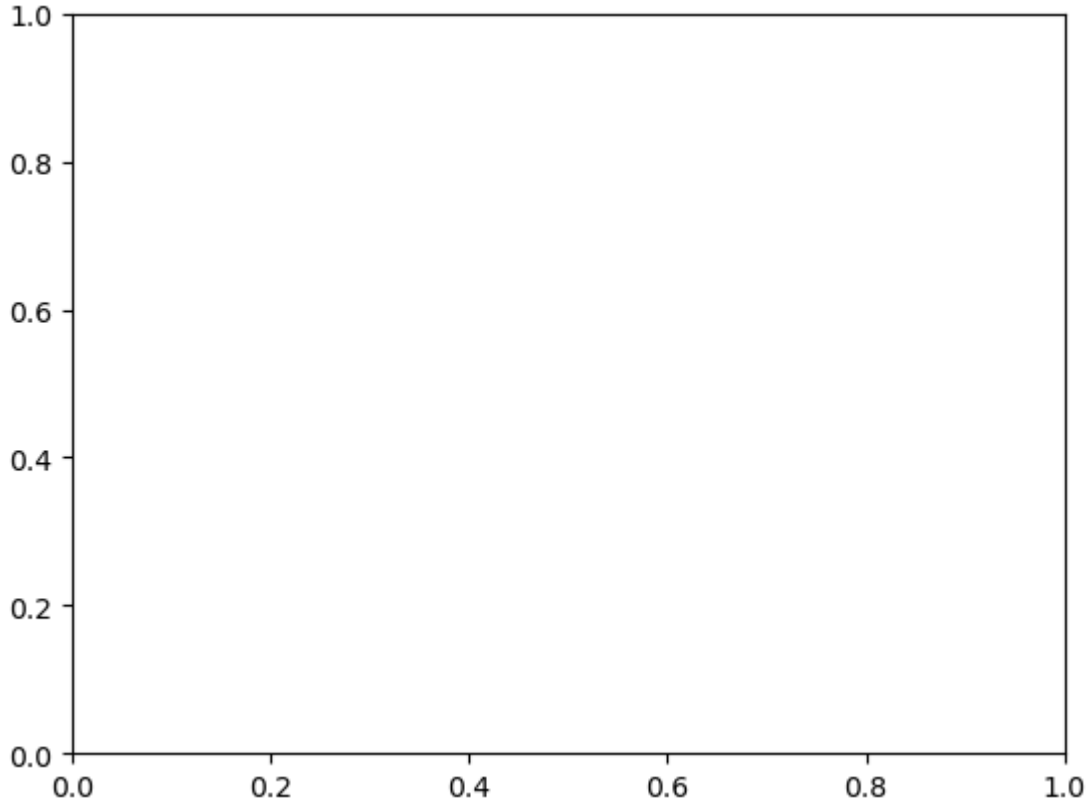
In [83]:
```
pip install matplotlib
```

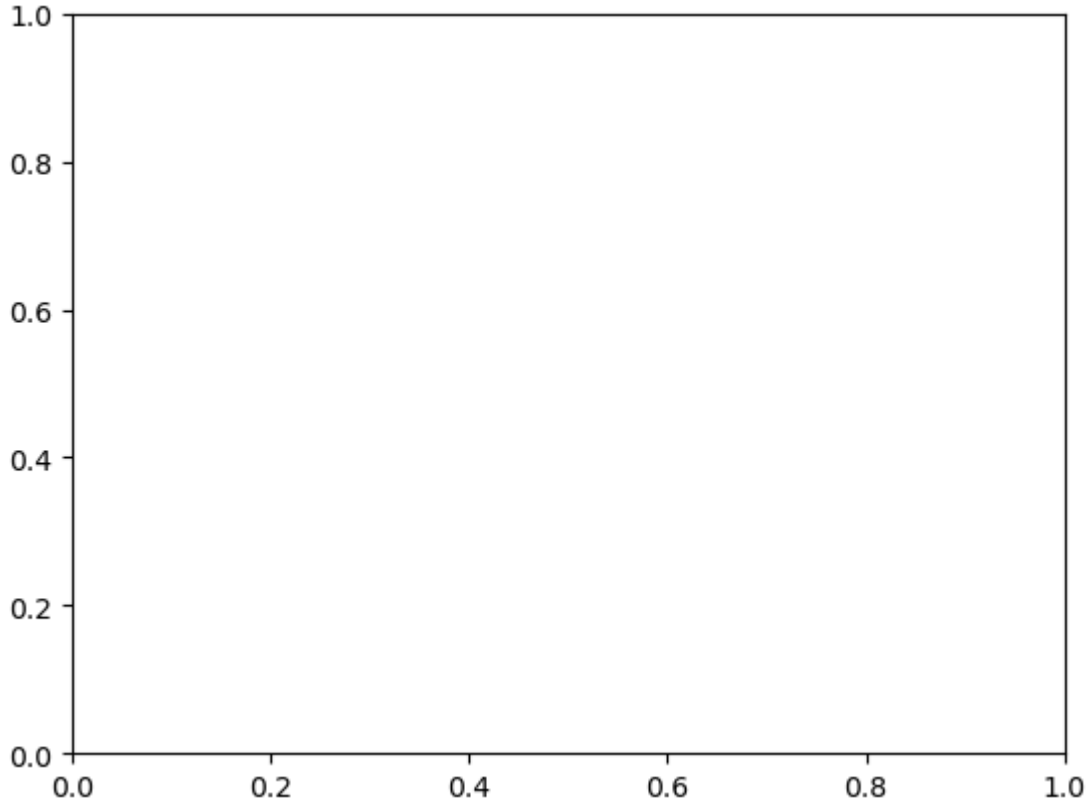
```
Requirement already satisfied: matplotlib in c:\users\santo\anaconda3\lib\site-packages (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: numpy>=1.23 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (2.1.3)
Requirement already satisfied: packaging>=20.0 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (2.9.0.post
0)
Requirement already satisfied: six>=1.5 in c:\users\santo\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib)
(1.17.0)
Note: you may need to restart the kernel to use updated packages.
```

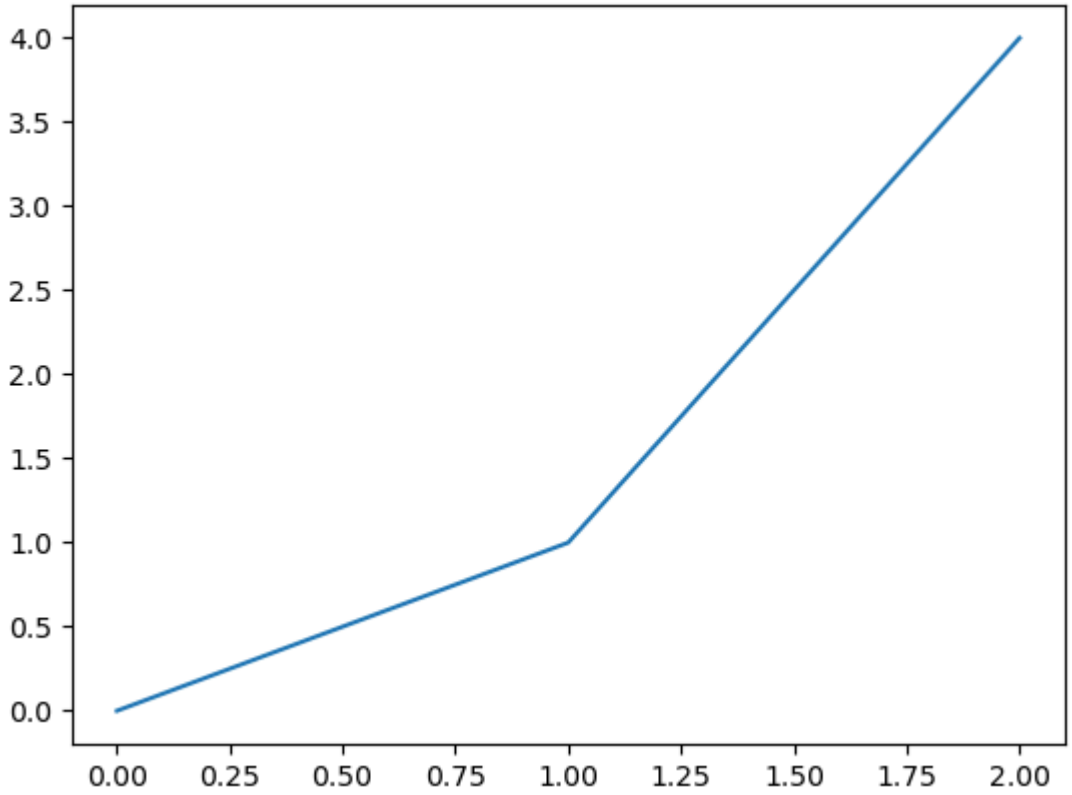
In [85]:

```
%matplotlib inline
import matplotlib.pyplot as plt

# Create a new figure
fig = plt.figure()

# Create default axes in the figure
ax = plt.axes()

# Plot something to test
ax.plot([0, 1, 2], [0, 1, 4])

plt.show()
```

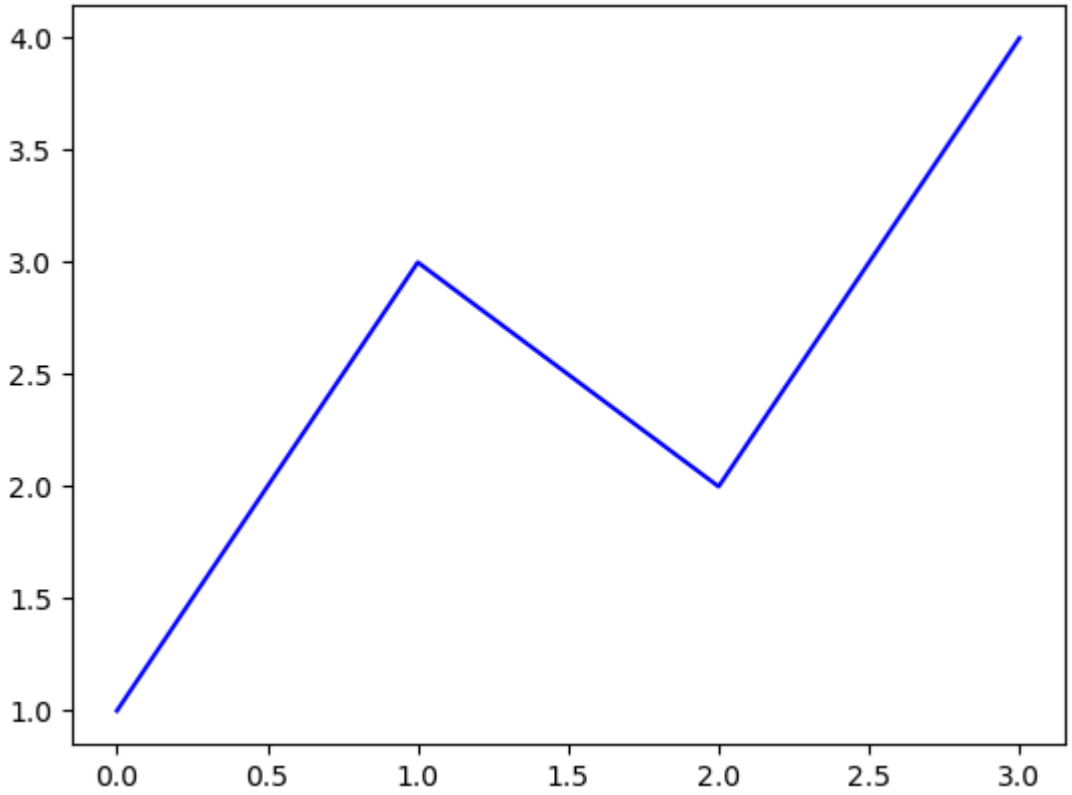

1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


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

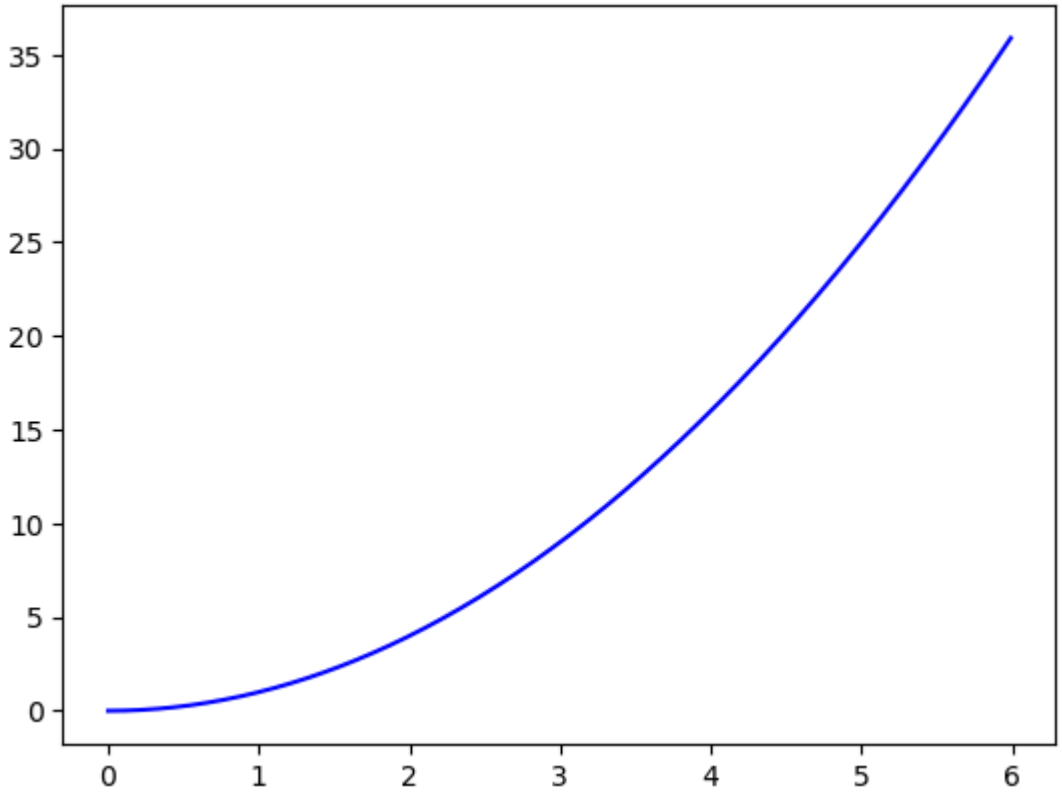

1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0



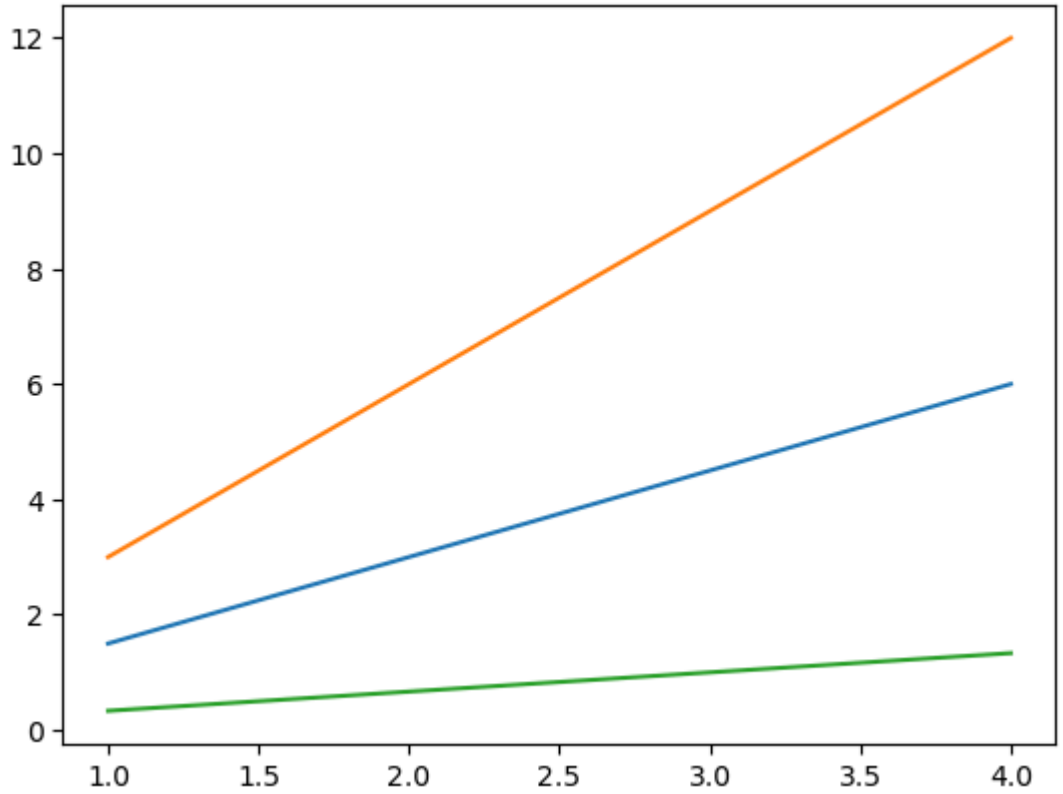
In [86]:
```
plt.plot([1, 3, 2, 4], 'b-')

plt.show( )
```



In [87]:
```
x3 = np.arange(0.0, 6.0, 0.01)

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

plt.show()
```



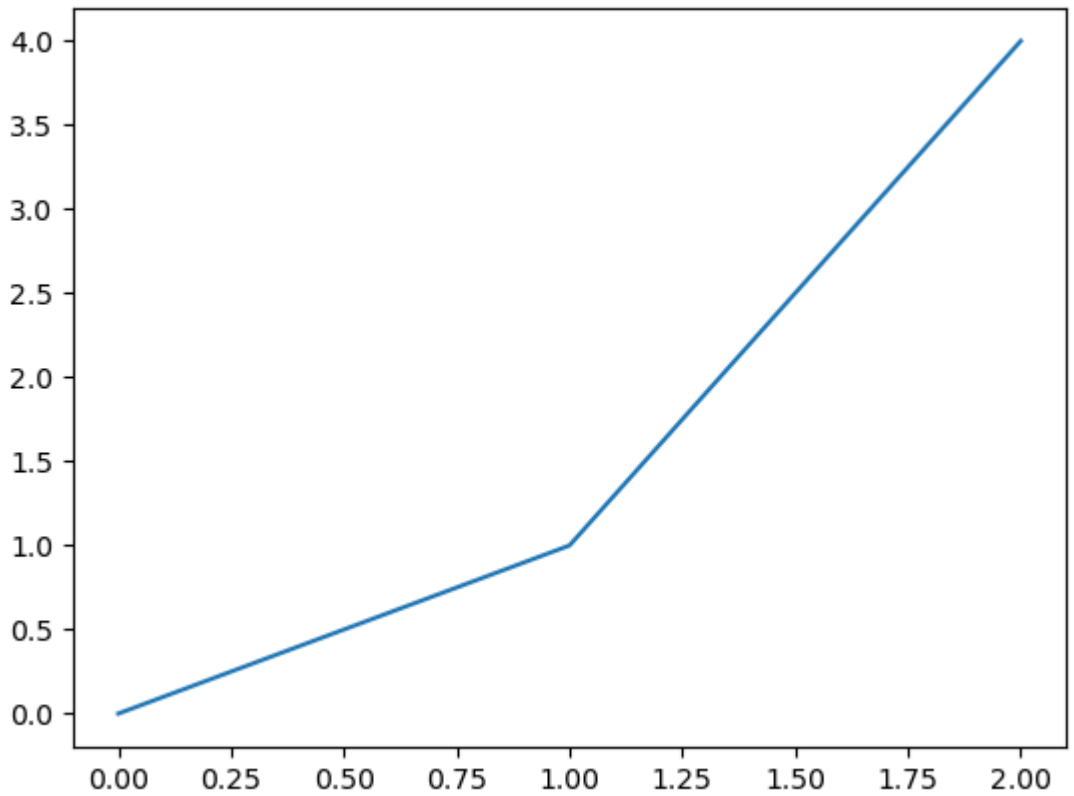
In [88]:
```
x4 = range(1, 5)

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 [89]:
```
# Saving the figure

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

In [90]:
```
# Explore the contents of figure

from IPython.display import Image

Image('plot1.png')
```

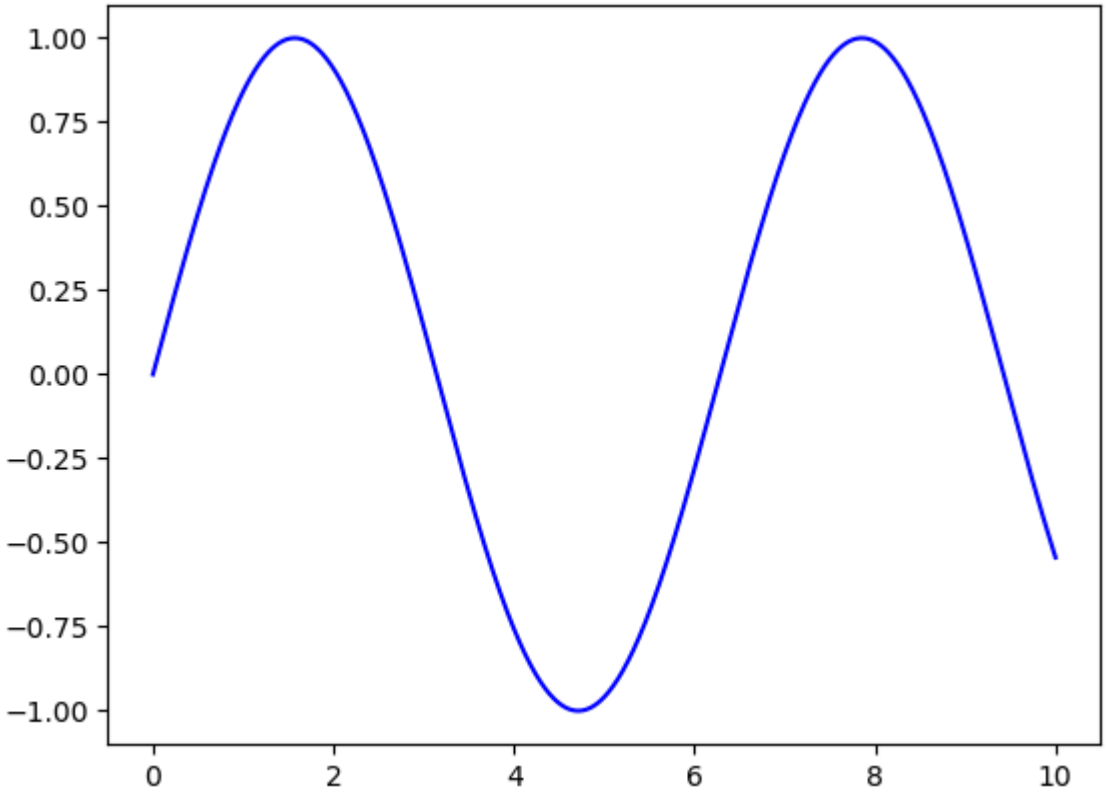
Out[90]:

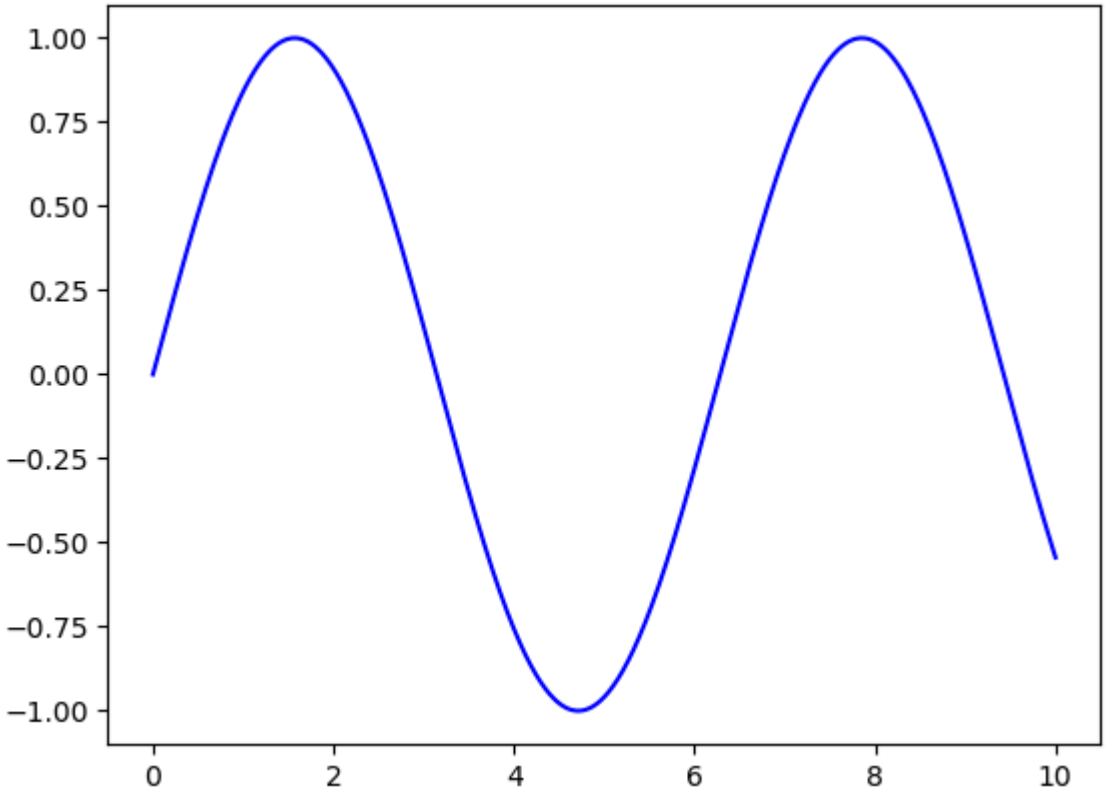


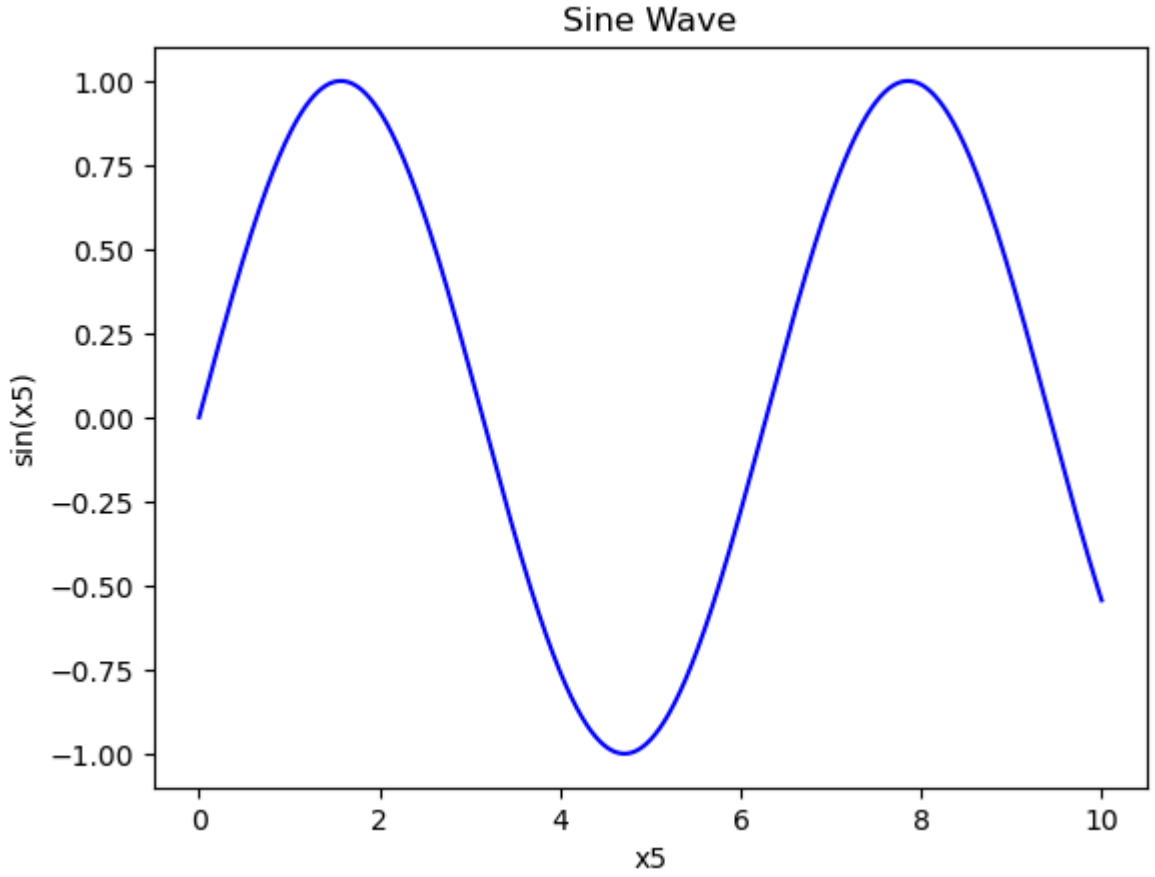
In [91]:
```
# Explore supported file formats


fig.canvas.get_supported_filetypes()
```

Out[91]:
```
{'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'}
```

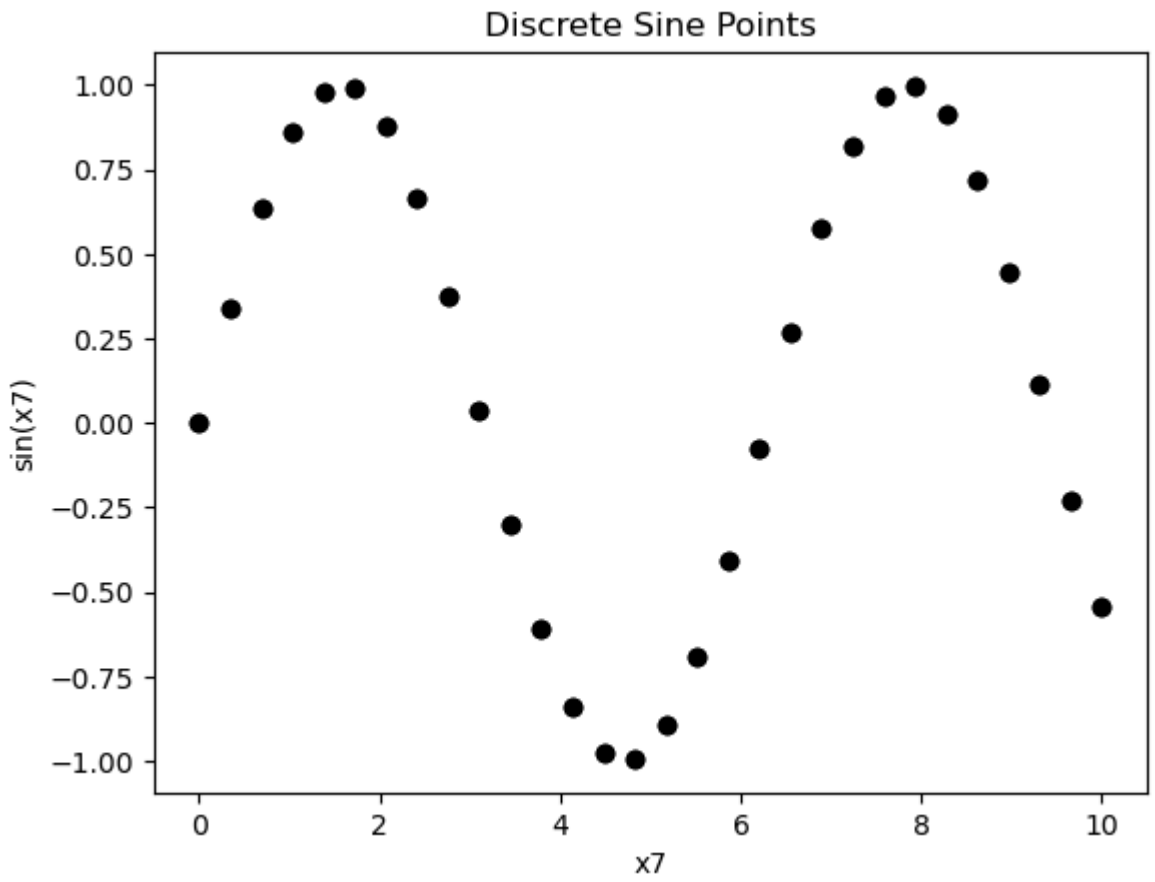
In [94]:
```
# Always include these two lines first in Jupyter Notebook
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

# Create figure and axes
fig = plt.figure()
ax = plt.axes()

# Declare a variable x5
x5 = np.linspace(0, 10, 1000)

# Plot the sinusoid function (sine wave)
ax.plot(x5, np.sin(x5), 'b-')

# Add title and labels (optional but helpful)
ax.set_title('Sine Wave')
ax.set_xlabel('x5')
ax.set_ylabel('sin(x5)')

# Show the plot (optional in Jupyter, but safe to include)
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


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

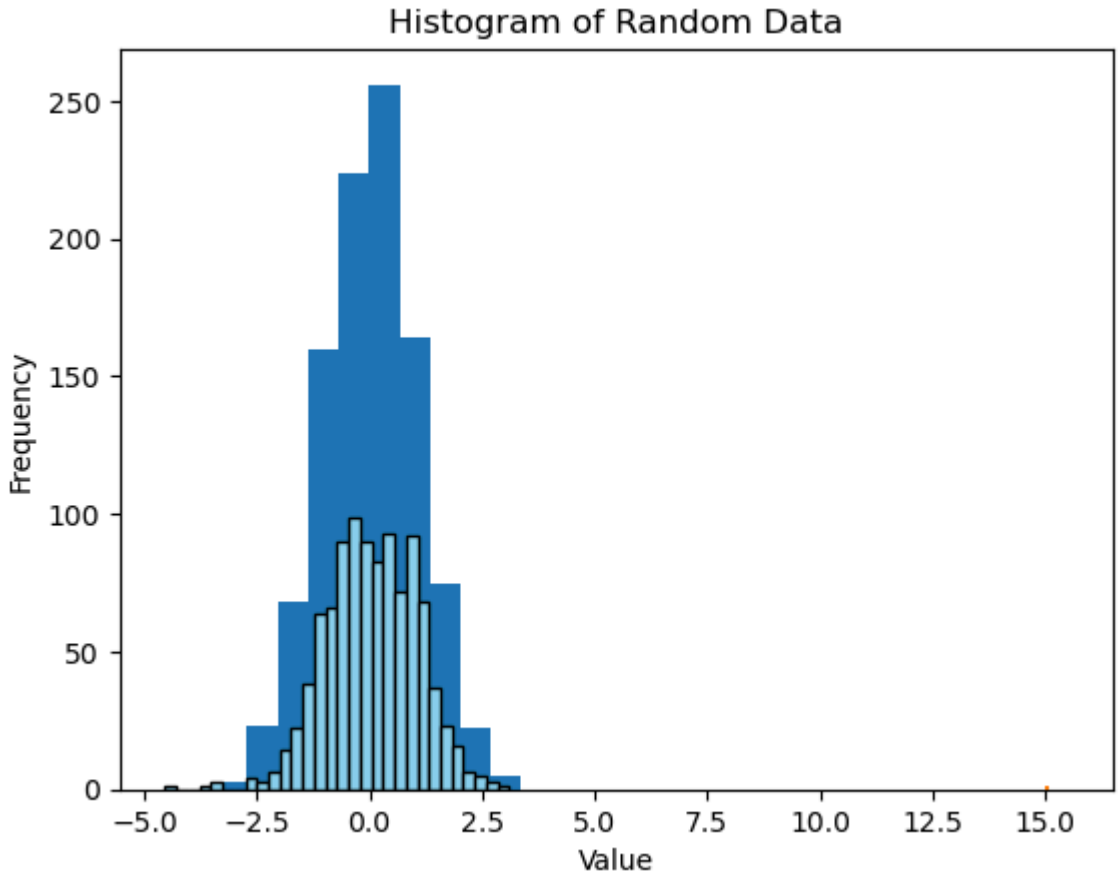


In [96]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

x7 = np.linspace(0, 10, 30)
y7 = np.sin(x7)

plt.plot(x7, y7, 'o', color='black')
plt.title('Discrete Sine Points')
plt.xlabel('x7')
plt.ylabel('sin(x7)')
plt.show()
```



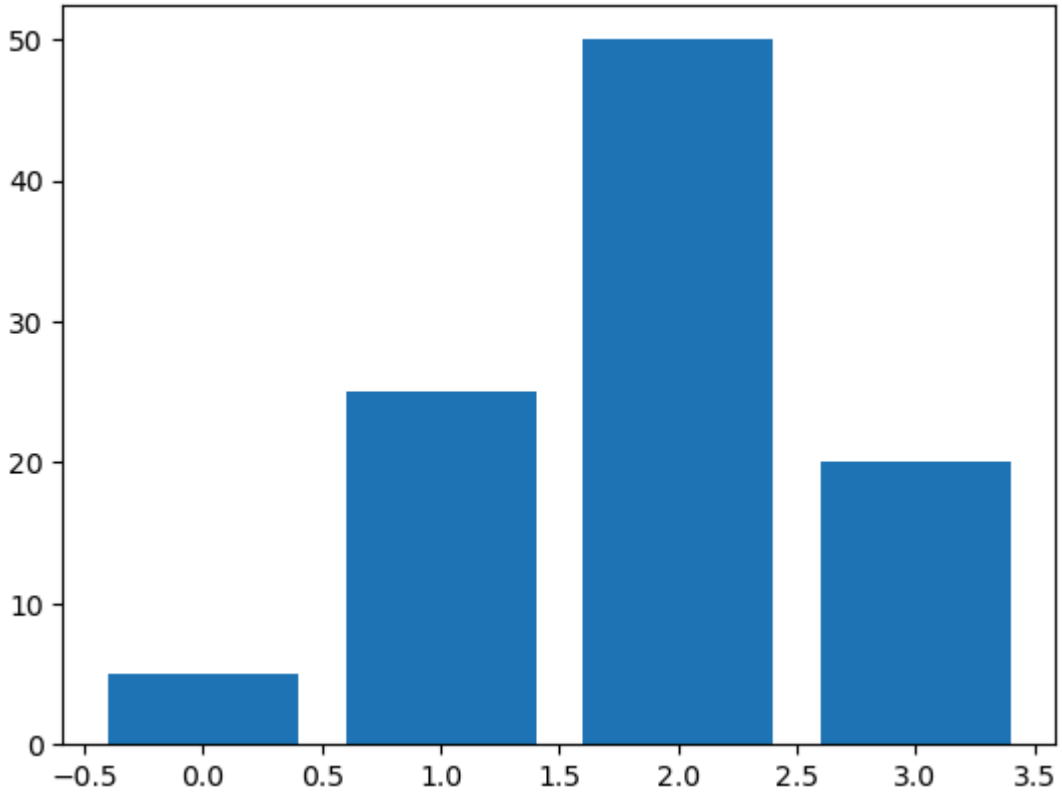
In [99]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

# Generate 1000 random values (from normal distribution)
data1 = np.random.randn(1000)

# Create histogram
plt.hist(data1, bins=30, color='skyblue', edgecolor='black')

# Add labels and title
plt.title('Histogram of Random Data')
plt.xlabel('Value')
```

```
plt.ylabel('Frequency')

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



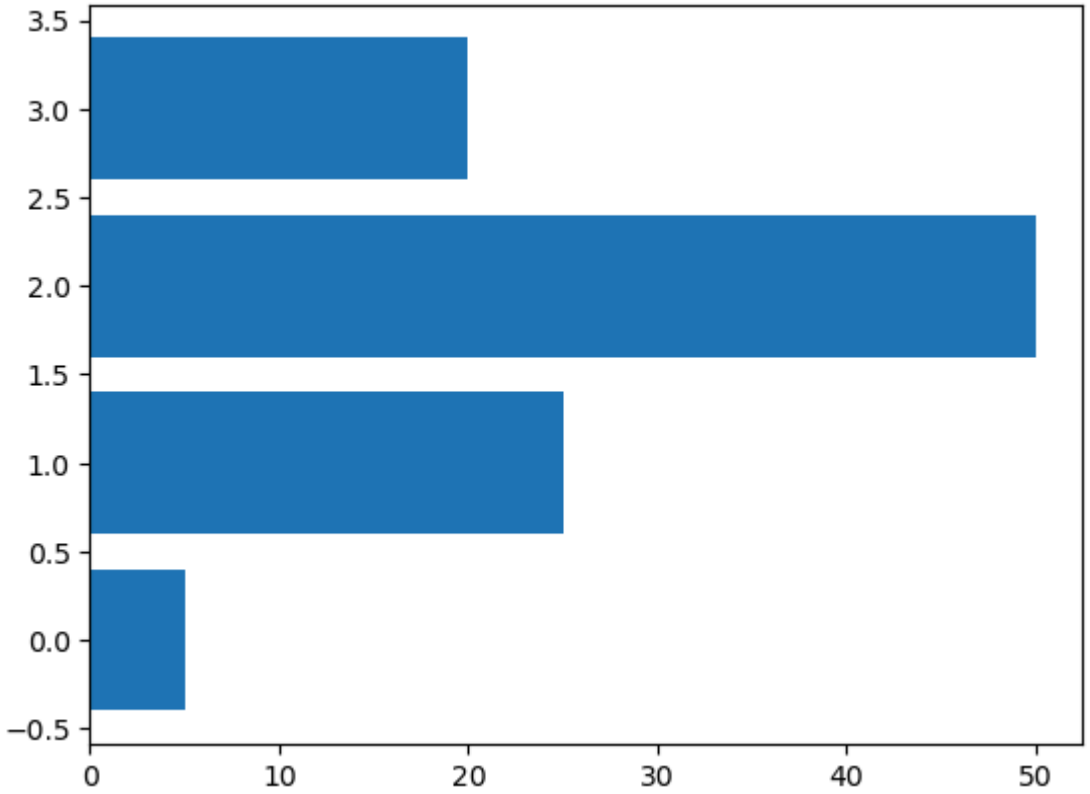
In [100…

```
data2 = [5. , 25. , 50. , 20.]

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

plt.show()
```



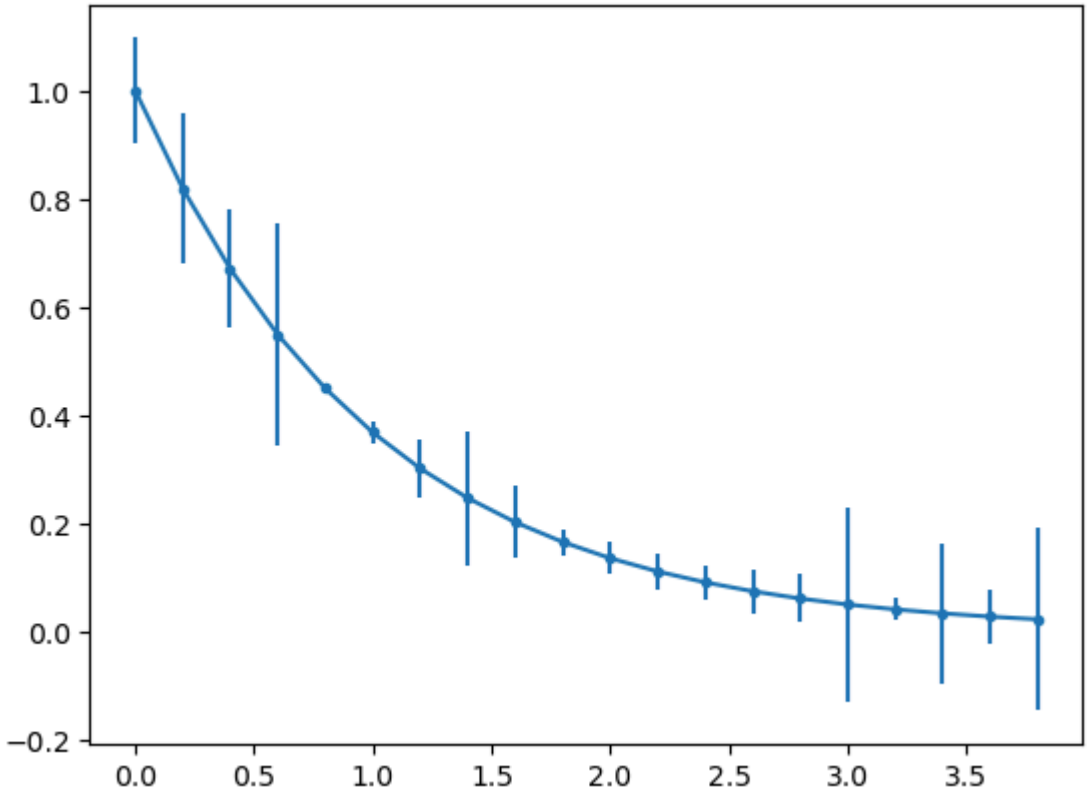
In [101…

```
data2 = [5. , 25. , 50. , 20.]

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

plt.show()
```



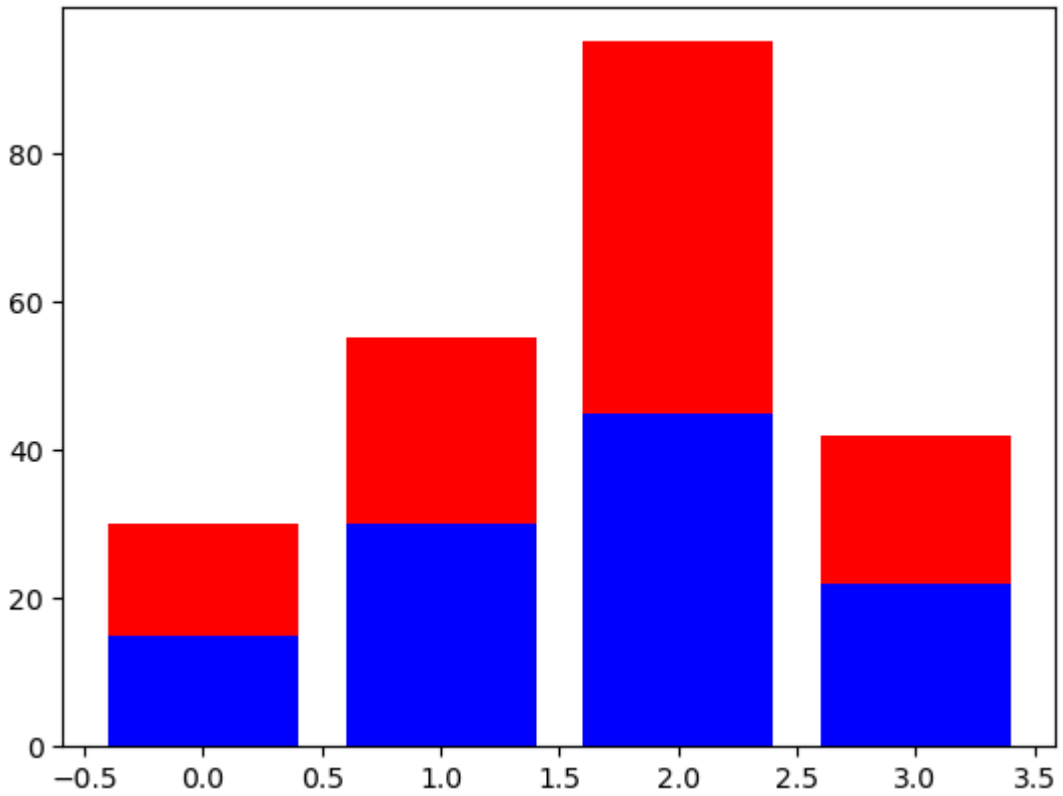
In [102…

```
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();
```



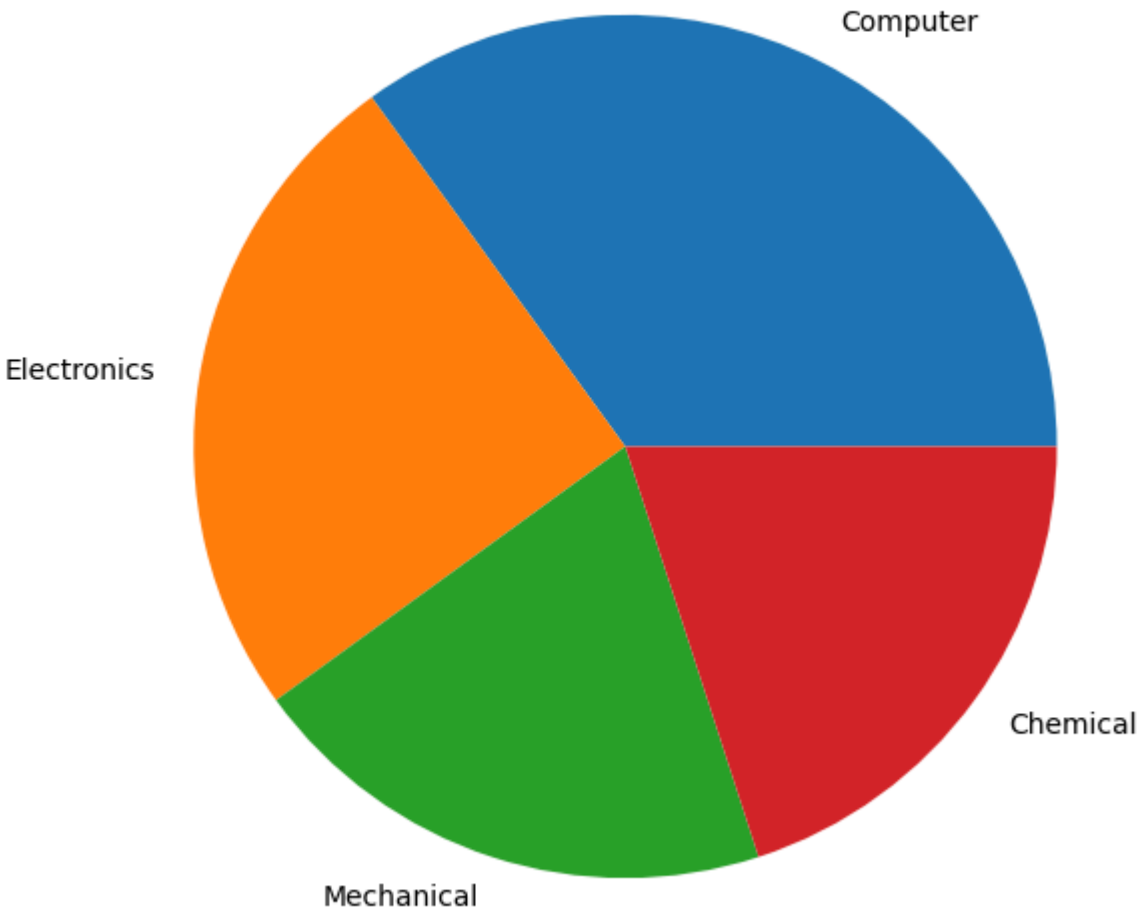
In [103…

```
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()
```



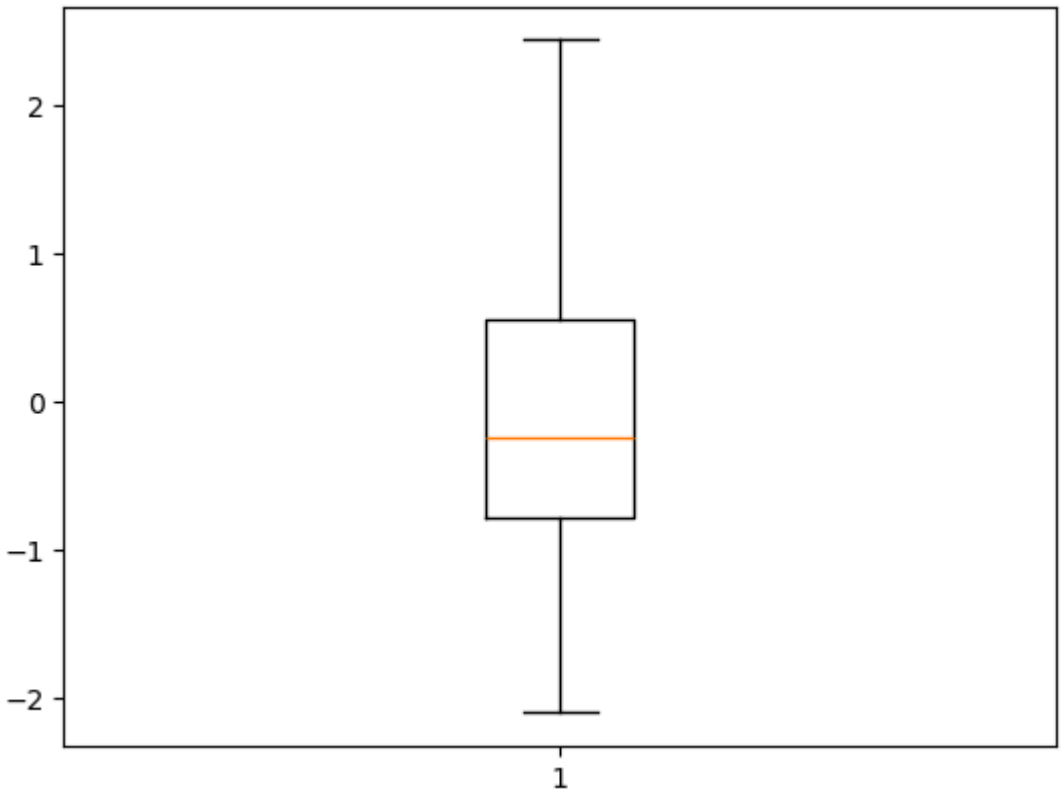
In [104…

```
plt.figure(figsize=(7,7))

x10 = [35, 25, 20, 20]

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

plt.pie(x10, labels=labels);

plt.show()
```



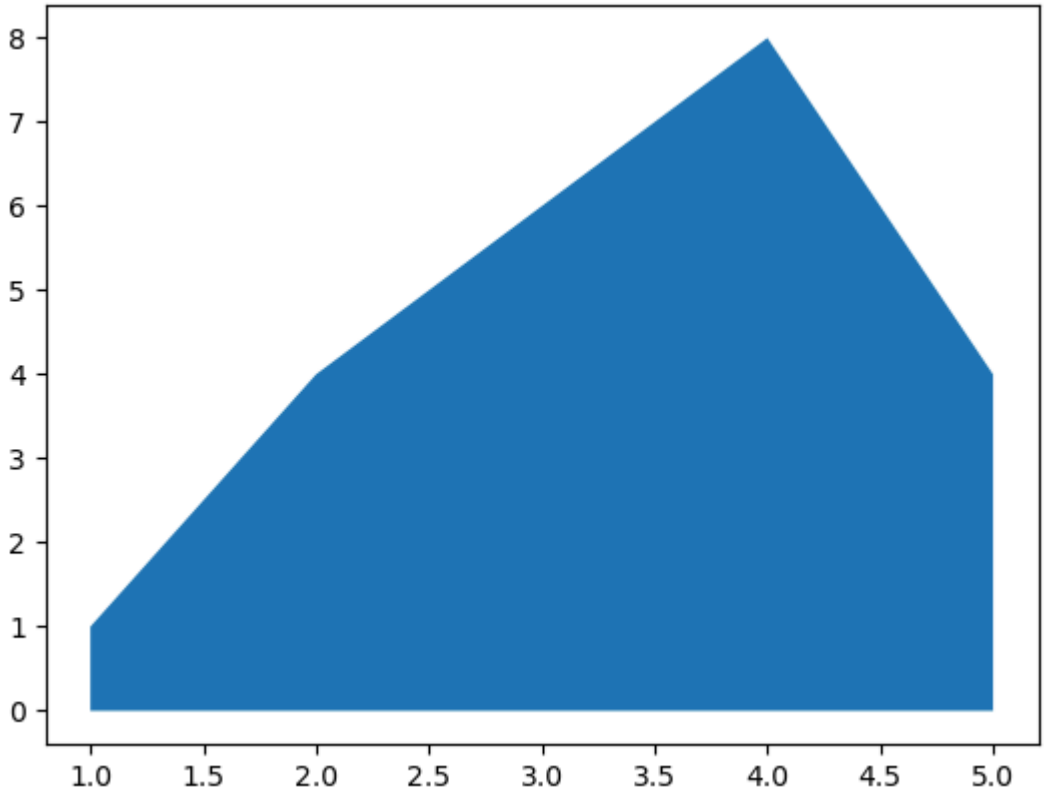
In [105…

```
data3 = np.random.randn(100)

plt.boxplot(data3)

plt.show();
```



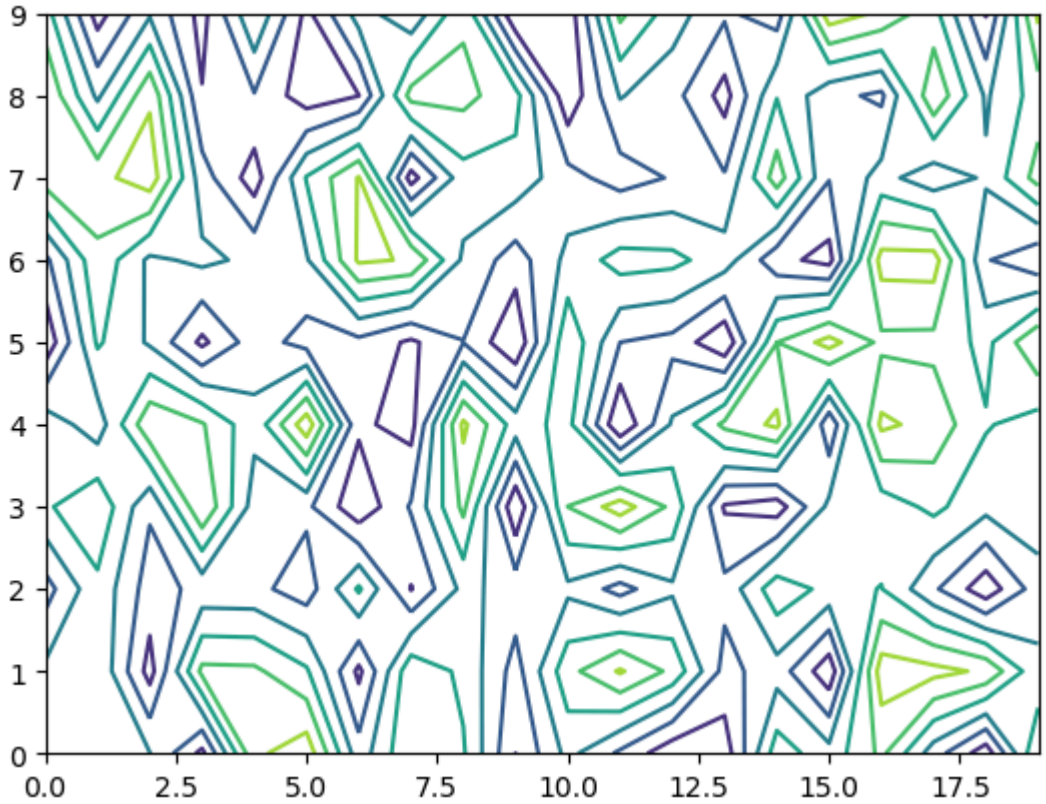
In [106…

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

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



In [107…

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

cp = plt.contour(matrix1)

plt.show()
```



In [108…
```
# View list of all available styles

print(plt.style.available)
```

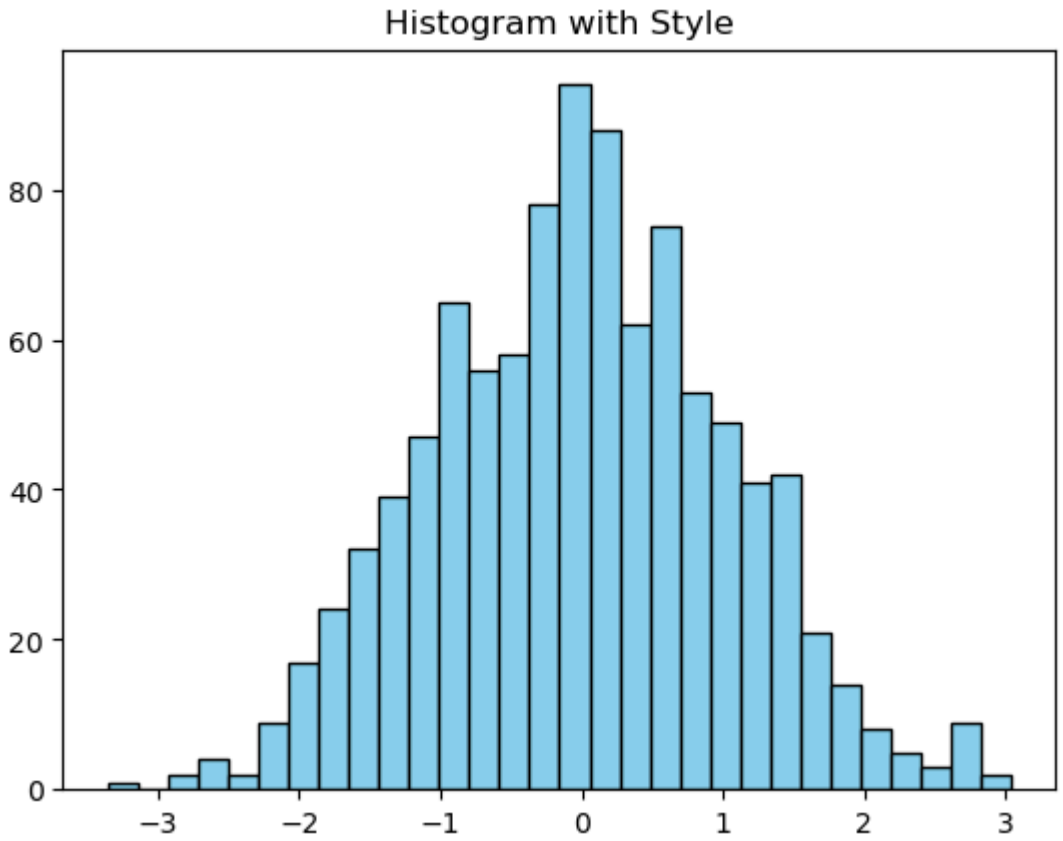
```
['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'bmh', 'classic', 'dark_background', 'fast',
'fivethirtyeight', 'ggplot', 'grayscale', 'petroff10', 'seaborn-v0_8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblind', 'seabo
rn-v0_8-dark', 'seaborn-v0_8-dark-palette',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

In [110…
```
pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\users\santo\anaconda3\lib\site-packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\santo\anaconda3\lib\site-packages (from seaborn) (2.1.3)
Requirement already satisfied: pandas>=1.2 in c:\users\santo\anaconda3\lib\site-packages (from seaborn) (2.2.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\santo\anaconda3\lib\site-packages (from seaborn) (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->se
aborn) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seabor
n) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->s
eaborn) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->s
eaborn) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->sea
born) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->se
aborn) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4
->seaborn) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\santo\anaconda3\lib\site-packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\santo\anaconda3\lib\site-packages (from pandas>=1.2->seaborn) (2025.
2)
Requirement already satisfied: six>=1.5 in c:\users\santo\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib!=
3.6.1,>=3.4->seaborn) (1.17.0)
Note: you may need to restart the kernel to use updated packages.
```

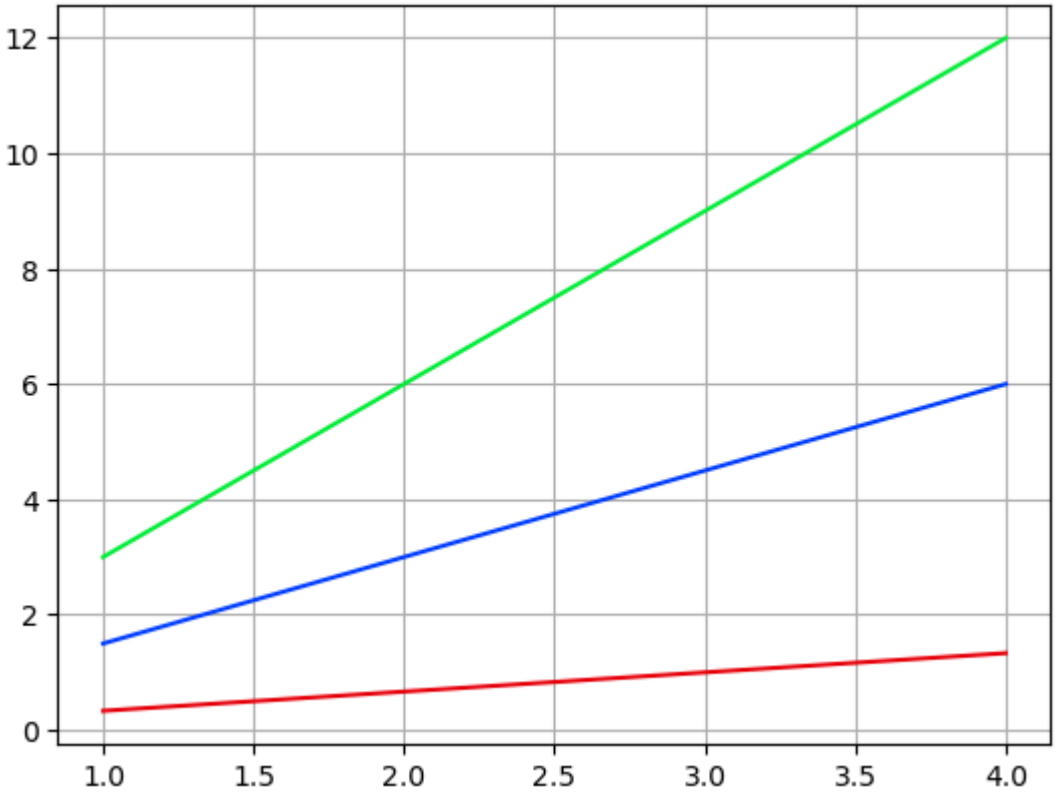
In [111…

```python
import numpy as np
import matplotlib.pyplot as plt

# Use a valid style
plt.style.use('seaborn-v0_8-bright')   # or choose one from plt.style.available

# Generate some data
data = np.random.randn(1000)

# Plot histogram
plt.hist(data, bins=30, color='skyblue', edgecolor='black')
plt.title('Histogram with Style')
plt.show()
```



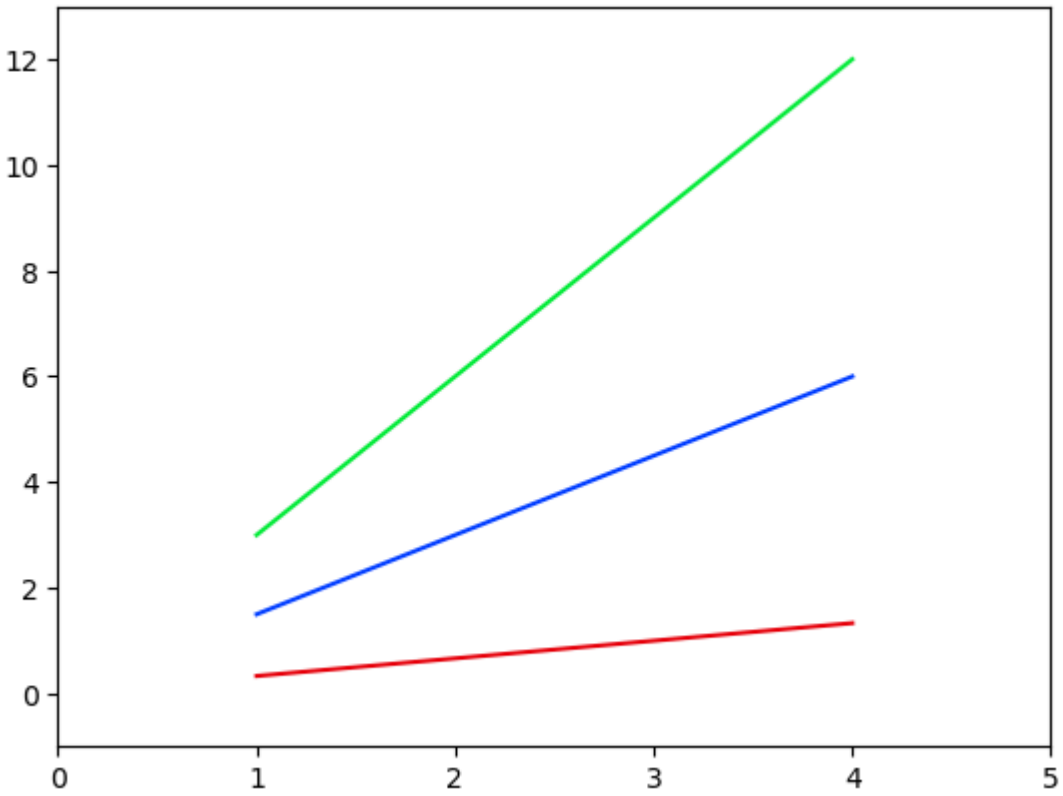
In [112…

```
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 [113…

```
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()
```

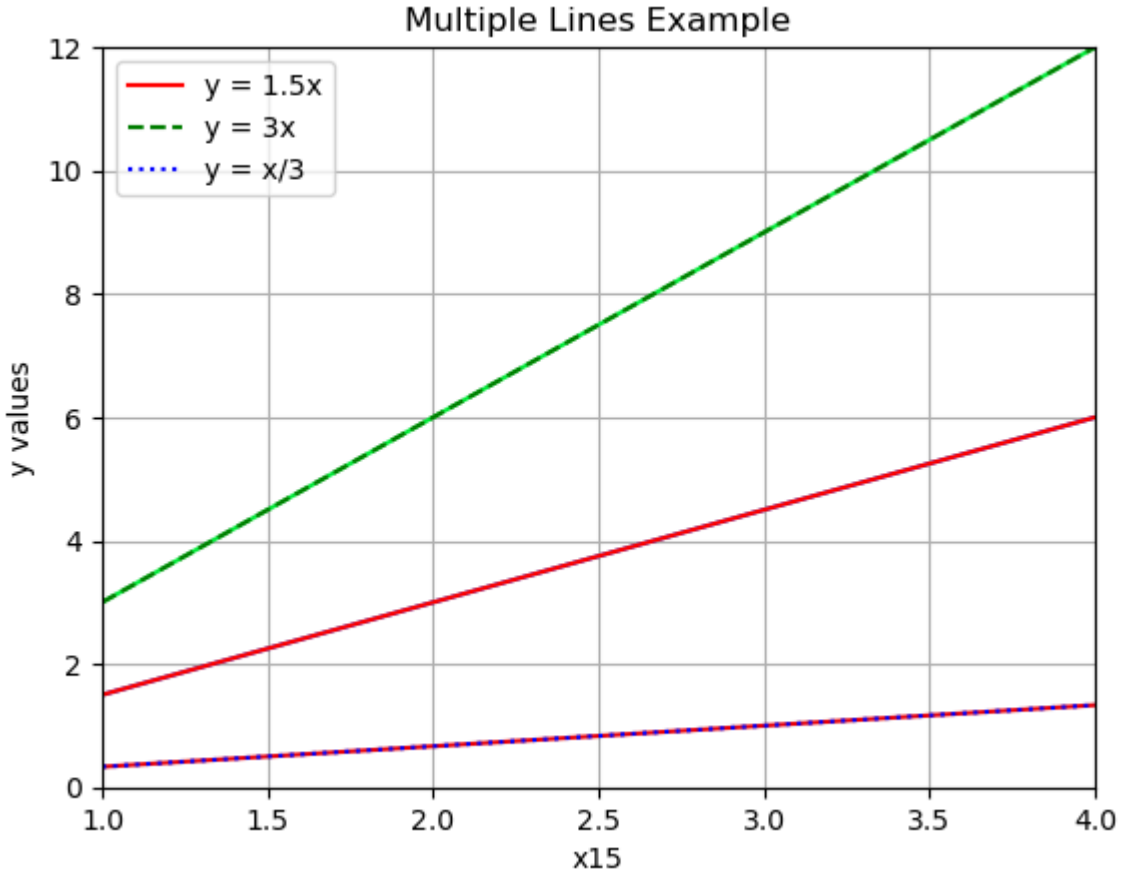


In [117…
```
x15 = np.arange(1, 5)

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[117…
```
(0.0, 12.0)
```

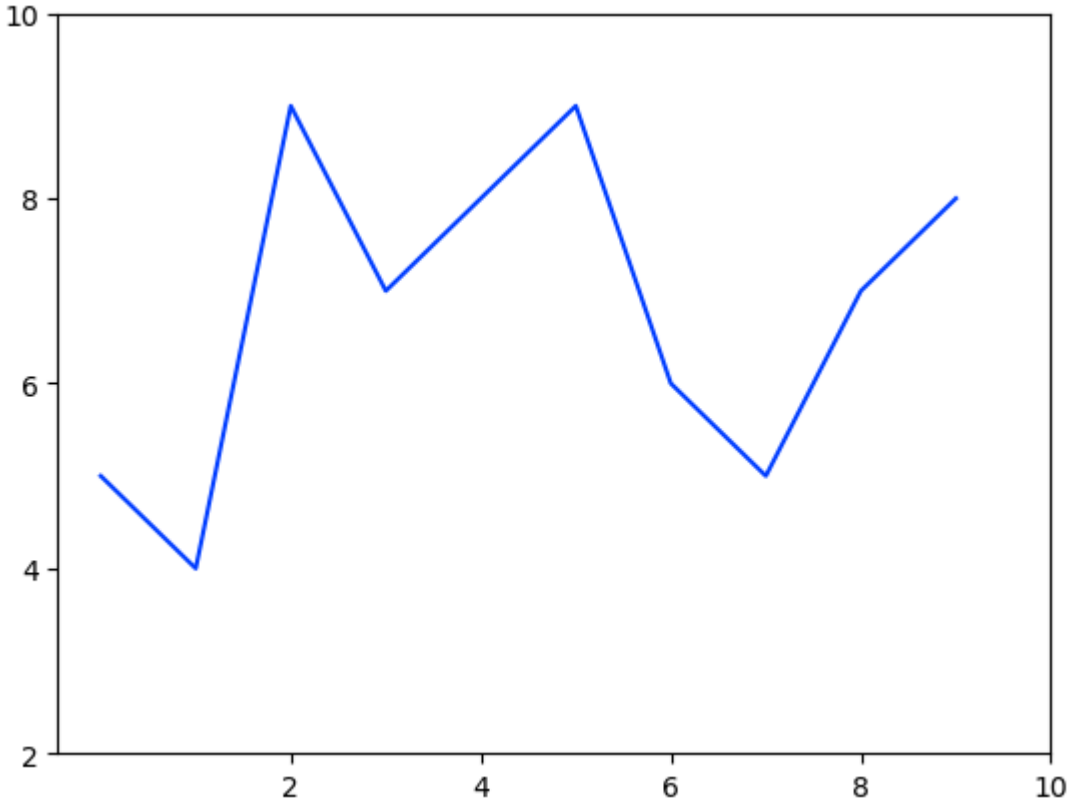
In [115…
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

# Create data
```

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

# Plot three different lines on the same axes
plt.plot(x15, x15 * 1.5, 'r-', label='y = 1.5x')
plt.plot(x15, x15 * 3.0, 'g--', label='y = 3x')
plt.plot(x15, x15 / 3.0, 'b:', label='y = x/3')

# Set x and y axis limits
plt.xlim([1.0, 4.0])
plt.ylim([0.0, 12.0])

# Add labels, title, legend, and grid
plt.xlabel('x15')
plt.ylabel('y values')
plt.title('Multiple Lines Example')
plt.legend()
plt.grid(True)

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



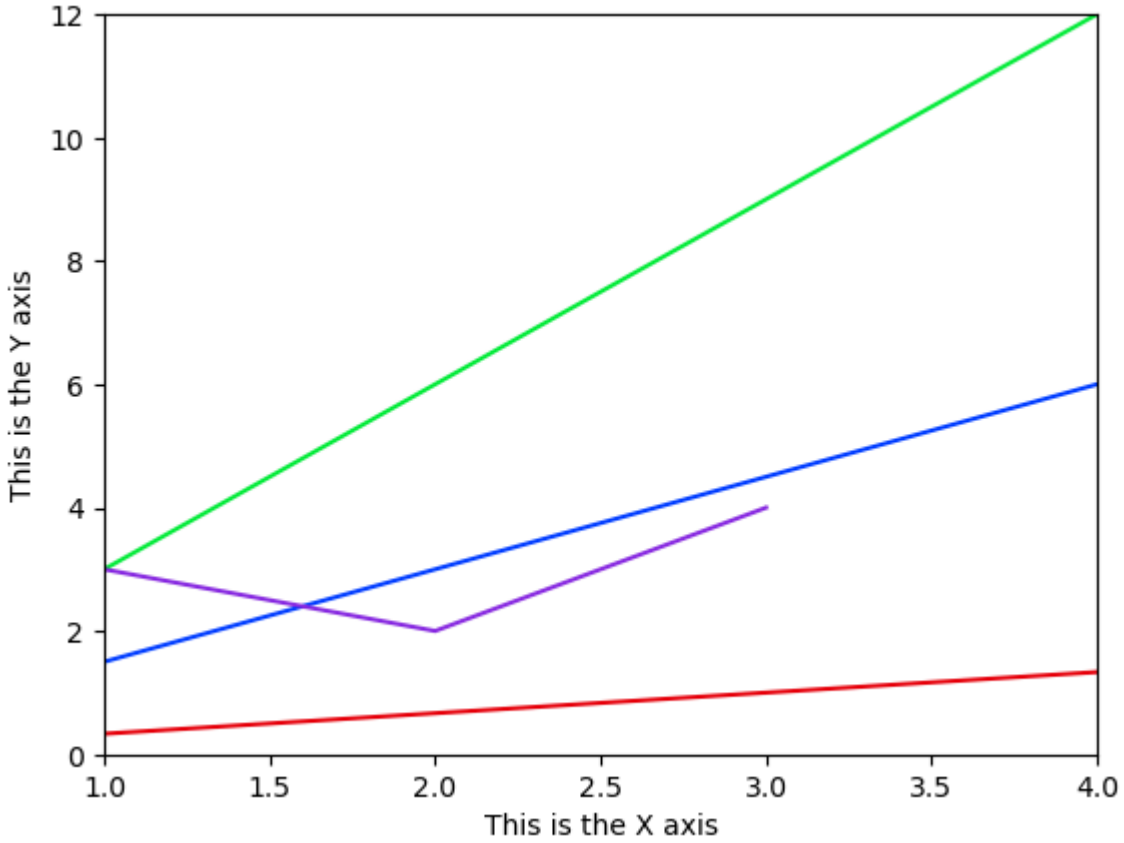
In [116…

```
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()
```



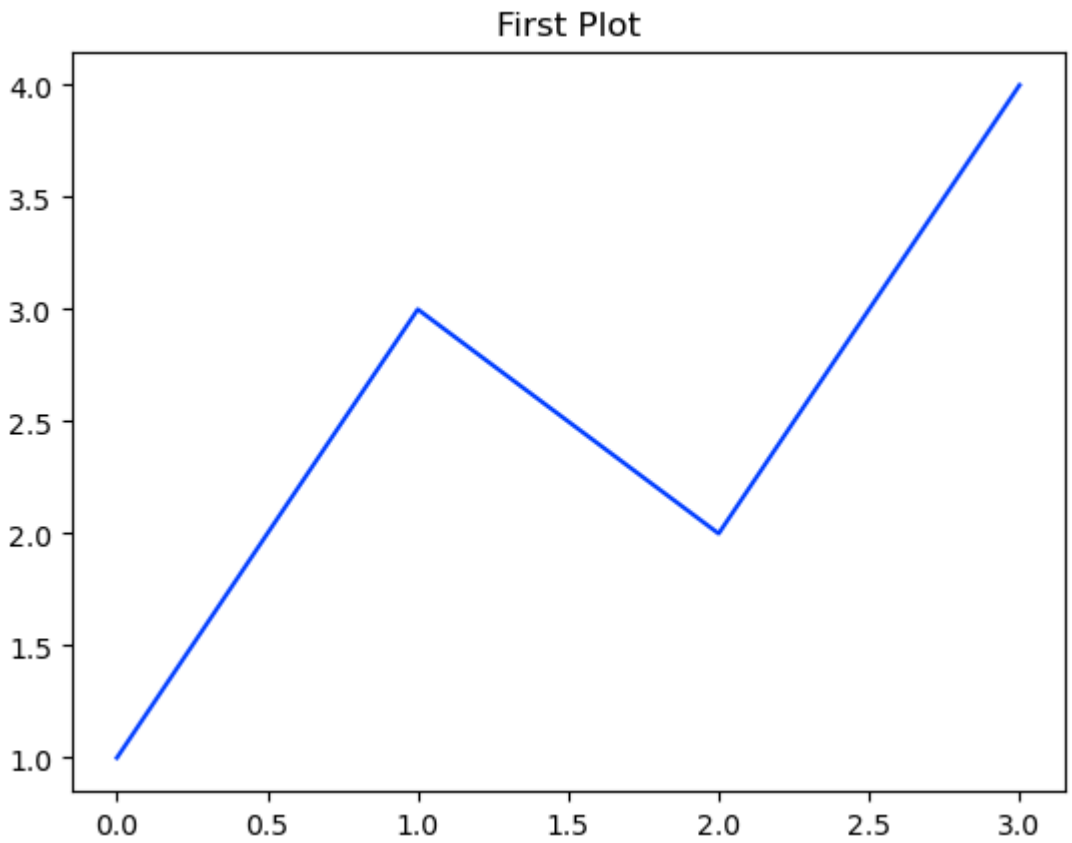
In [118…

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

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

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

plt.show()
```



In [119…

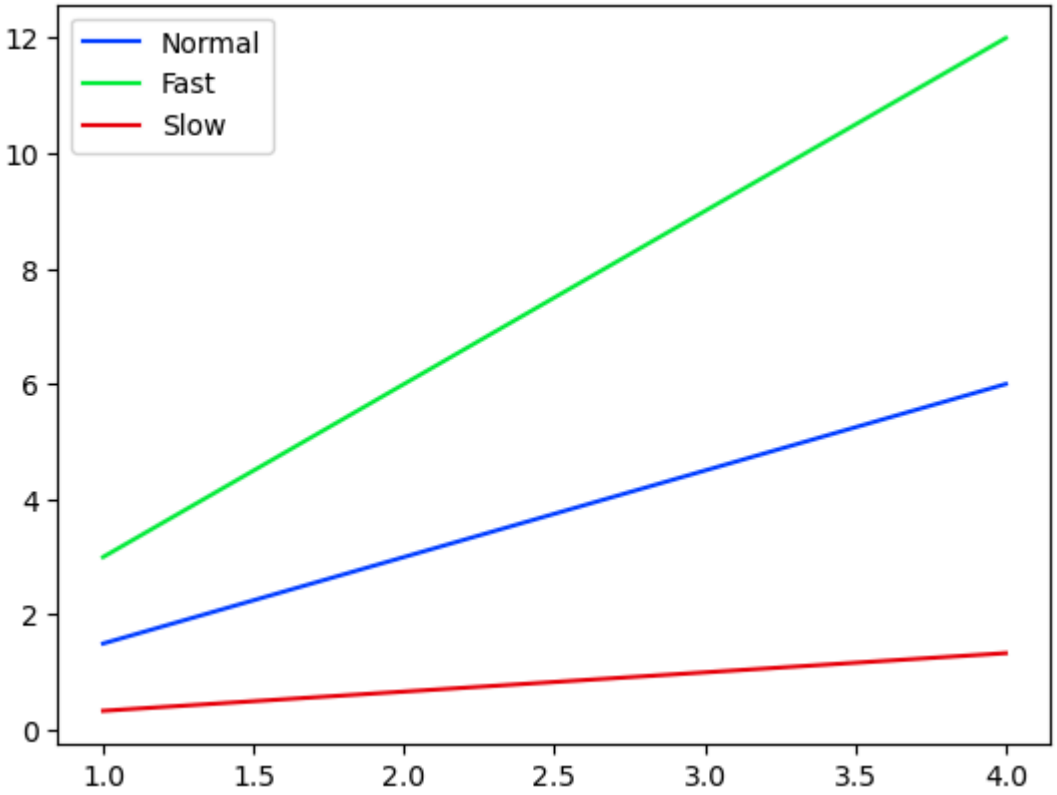
```
plt.plot([1, 3, 2, 4])

plt.title('First Plot')

plt.show()
```

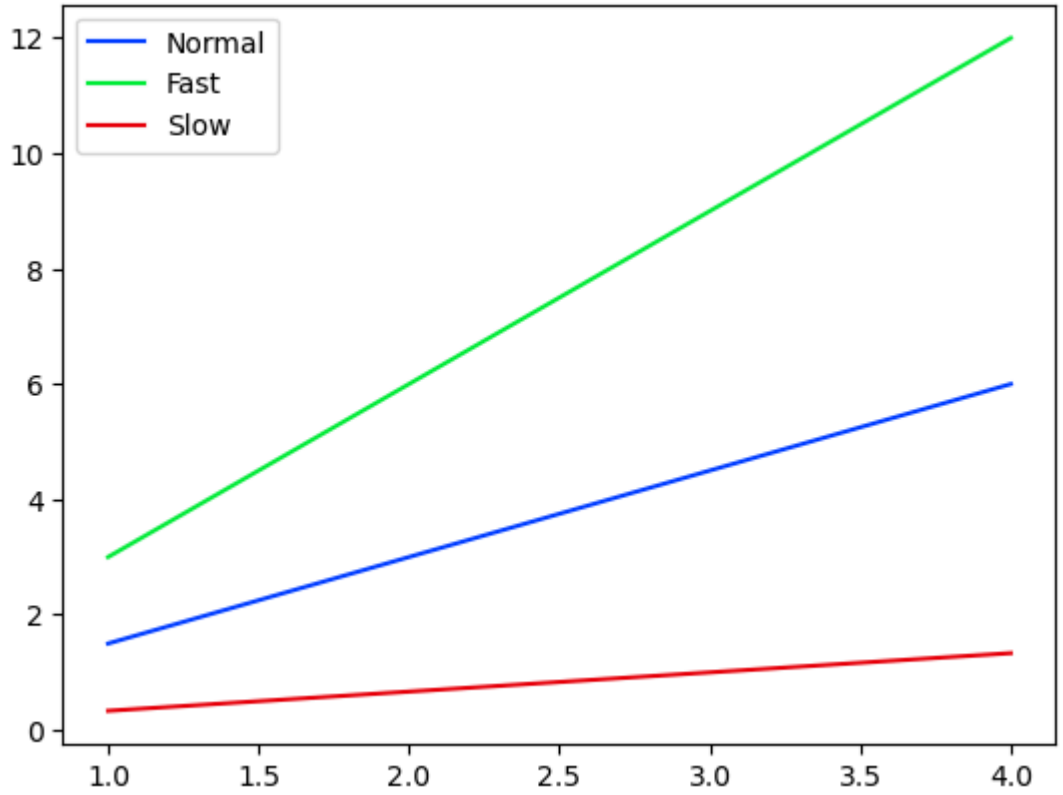


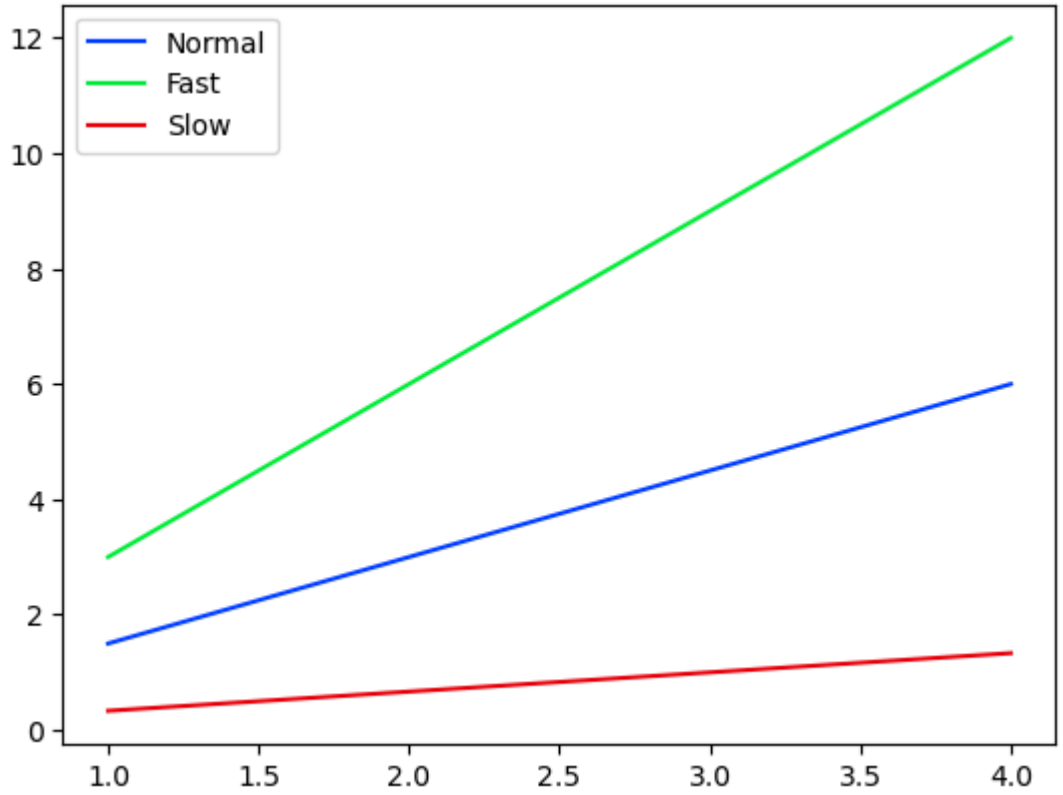
In [123…

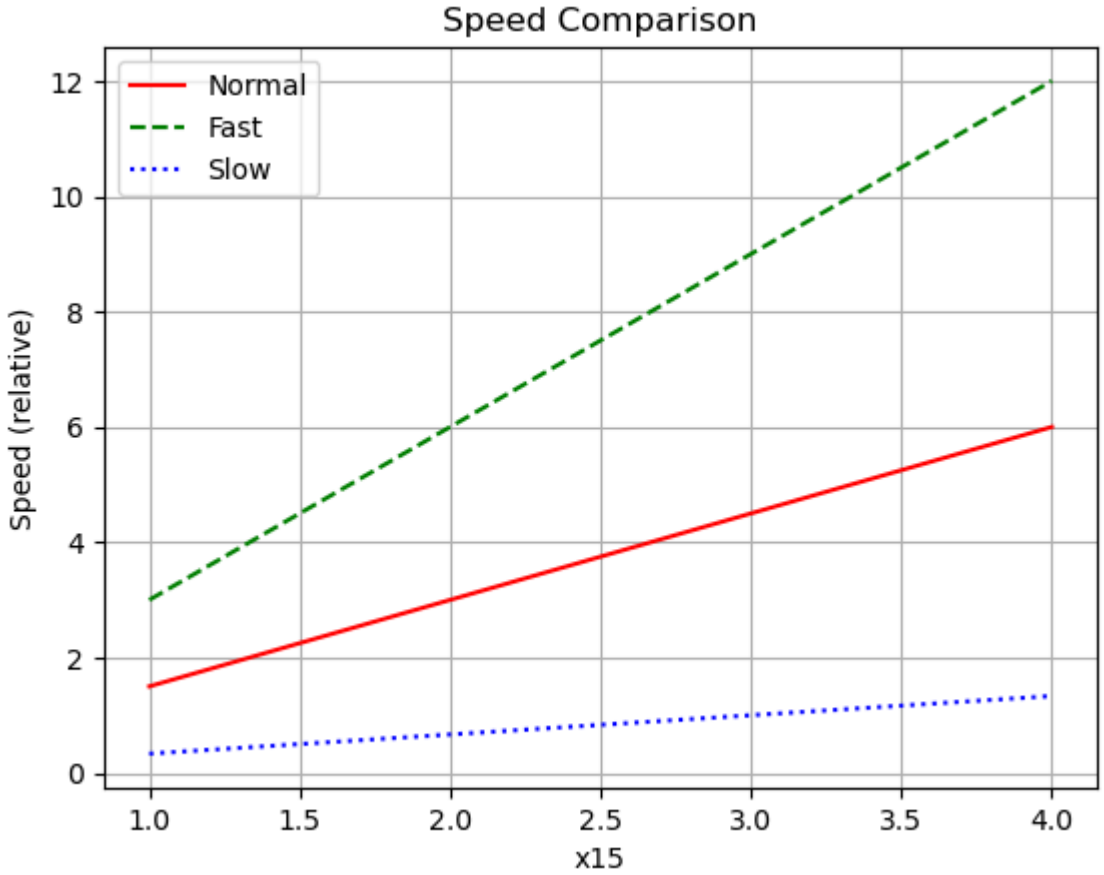
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

# Create data
x15 = np.arange(1, 5)

# Create a figure and an axes
fig, ax = plt.subplots()

# Plot three lines
ax.plot(x15, x15 * 1.5, 'r-', label='Normal')
ax.plot(x15, x15 * 3.0, 'g--', label='Fast')
ax.plot(x15, x15 / 3.0, 'b:', label='Slow')
```

```
# Add legend (auto-detects labels)
ax.legend()

# Add title and labels
ax.set_title('Speed Comparison')
ax.set_xlabel('x15')
ax.set_ylabel('Speed (relative)')

# Add grid
ax.grid(True)

# Show the figure
plt.show()
```




Normal
Fast
Slow
12
10
8
6
4
2
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0

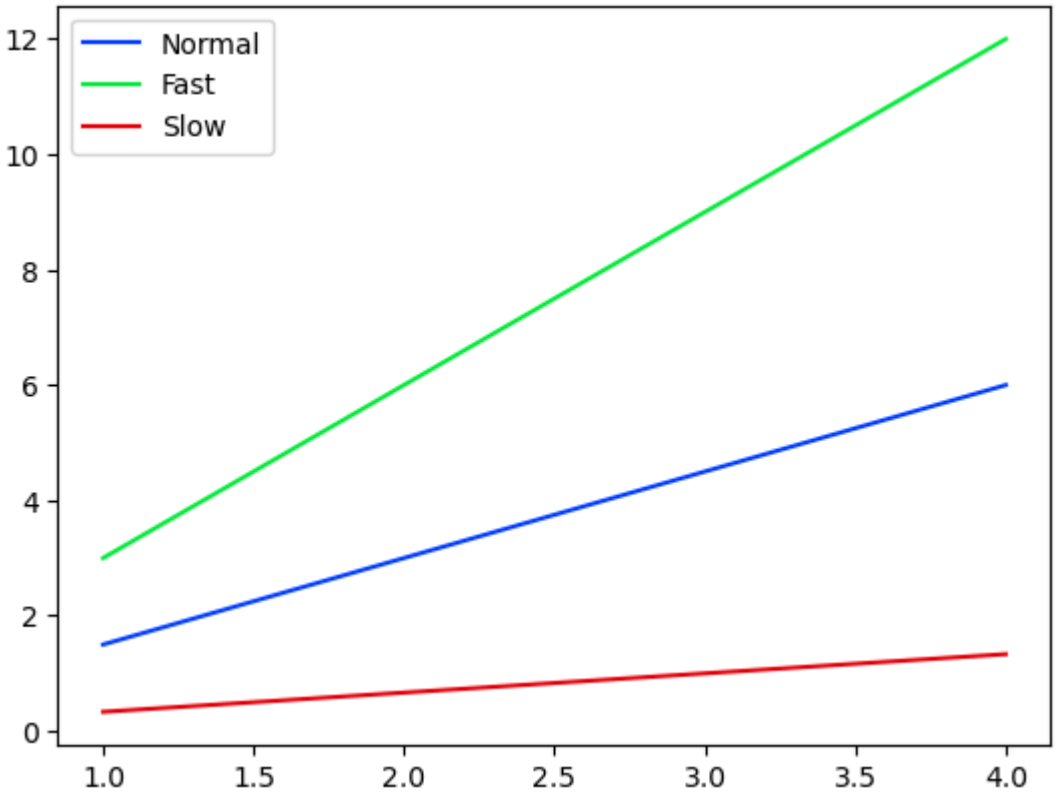

Normal
Fast
Slow
12
10
8
6
4
2
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0

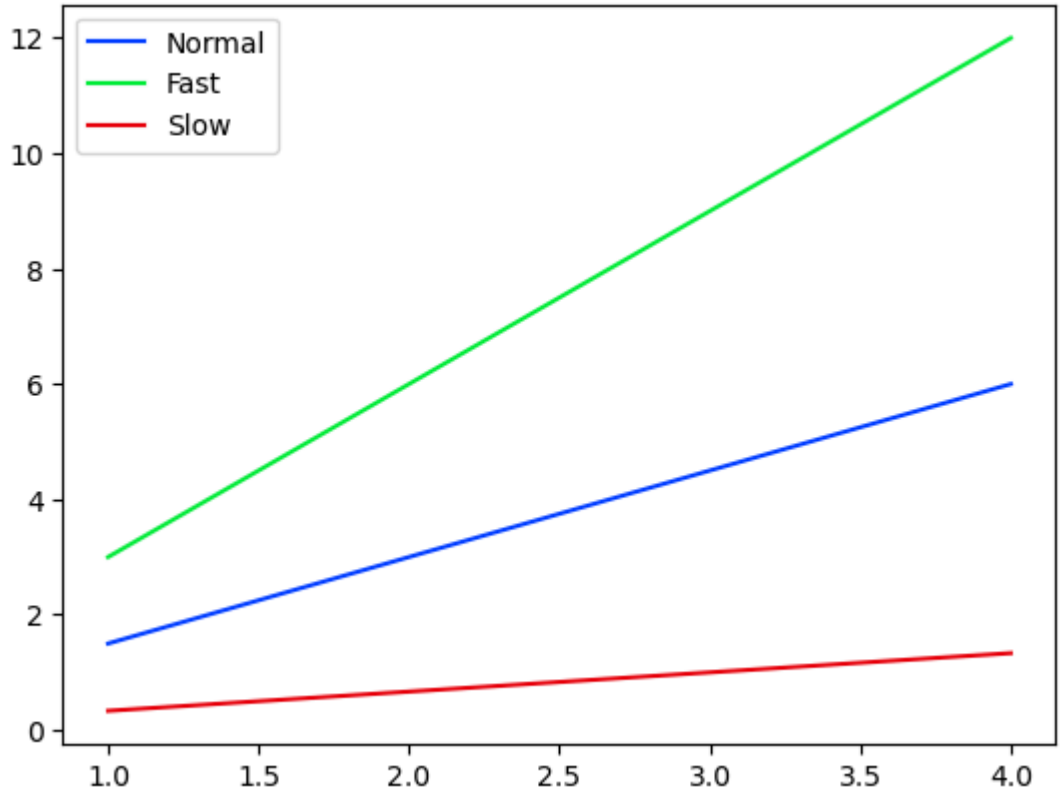


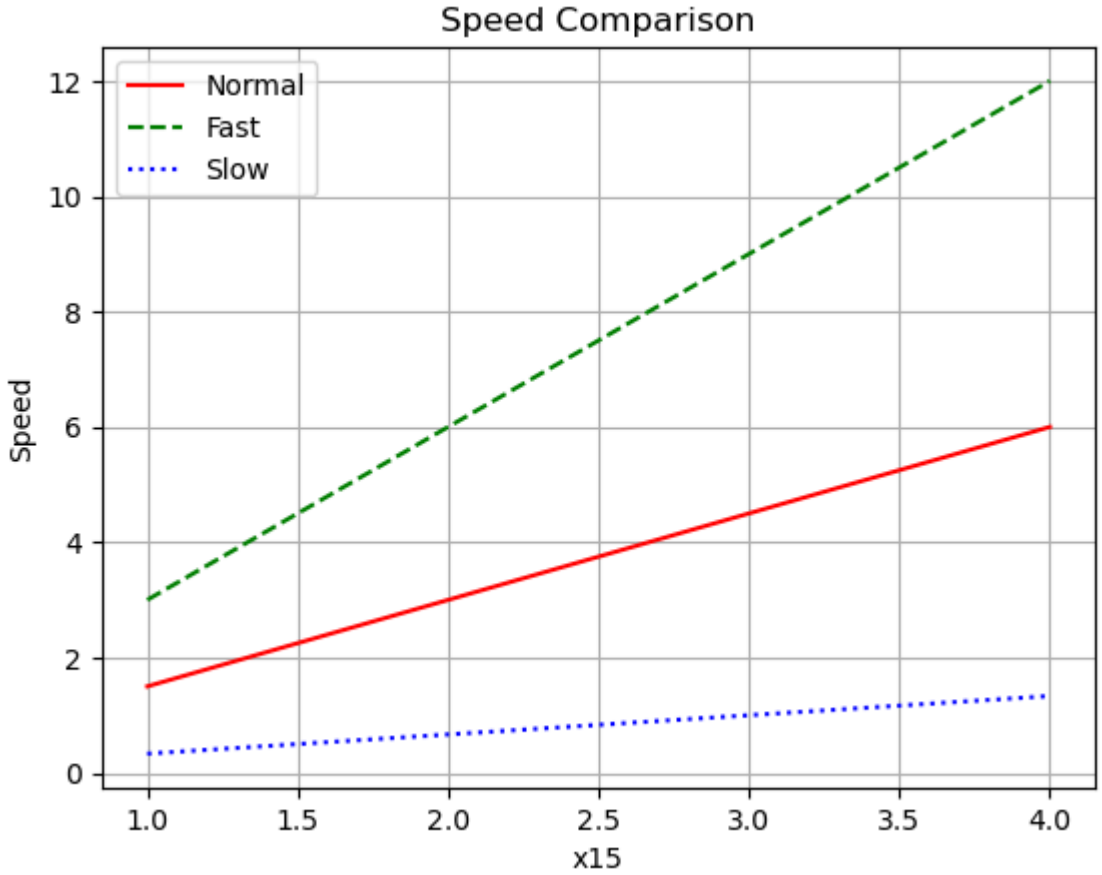
In [126…

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

# Create data
x15 = np.arange(1, 5)

# Create figure and axis
fig, ax = plt.subplots()

# Plot multiple lines with labels
ax.plot(x15, x15 * 1.5, 'r-', label='Normal')
ax.plot(x15, x15 * 3.0, 'g--', label='Fast')
```

```
ax.plot(x15, x15 / 3.0, 'b:', label='Slow')

# Show legend
ax.legend()

# Add title and labels
ax.set_title('Speed Comparison')
ax.set_xlabel('x15')
ax.set_ylabel('Speed')

# Add grid
ax.grid(True)

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

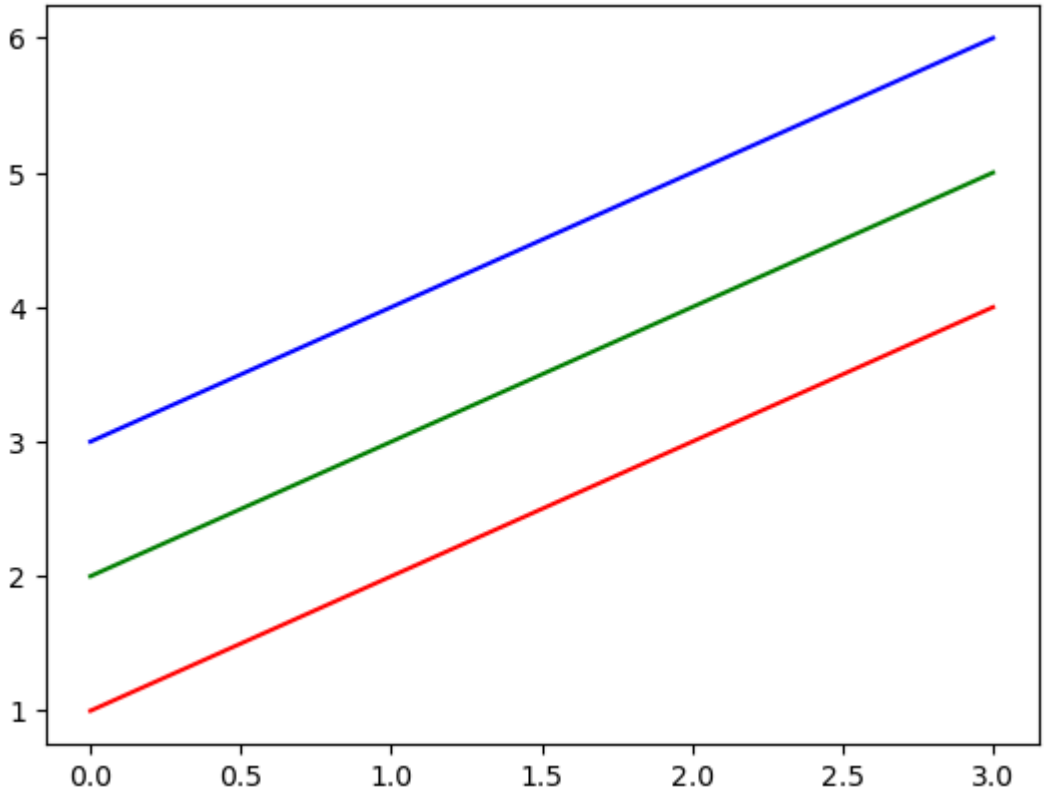



Normal
Fast
Slow
12
10
8
6
4
2
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0



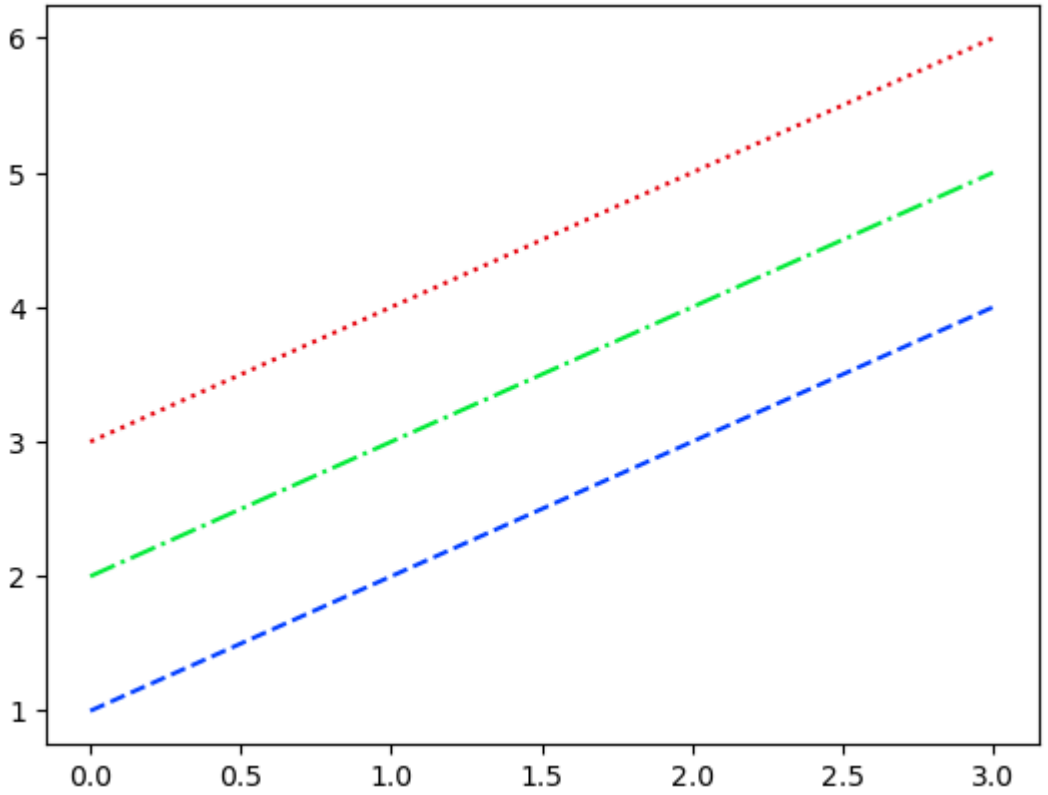
In [127…

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

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

plt.show()
```



In [128…

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

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

plt.show()
```



In [ ]: