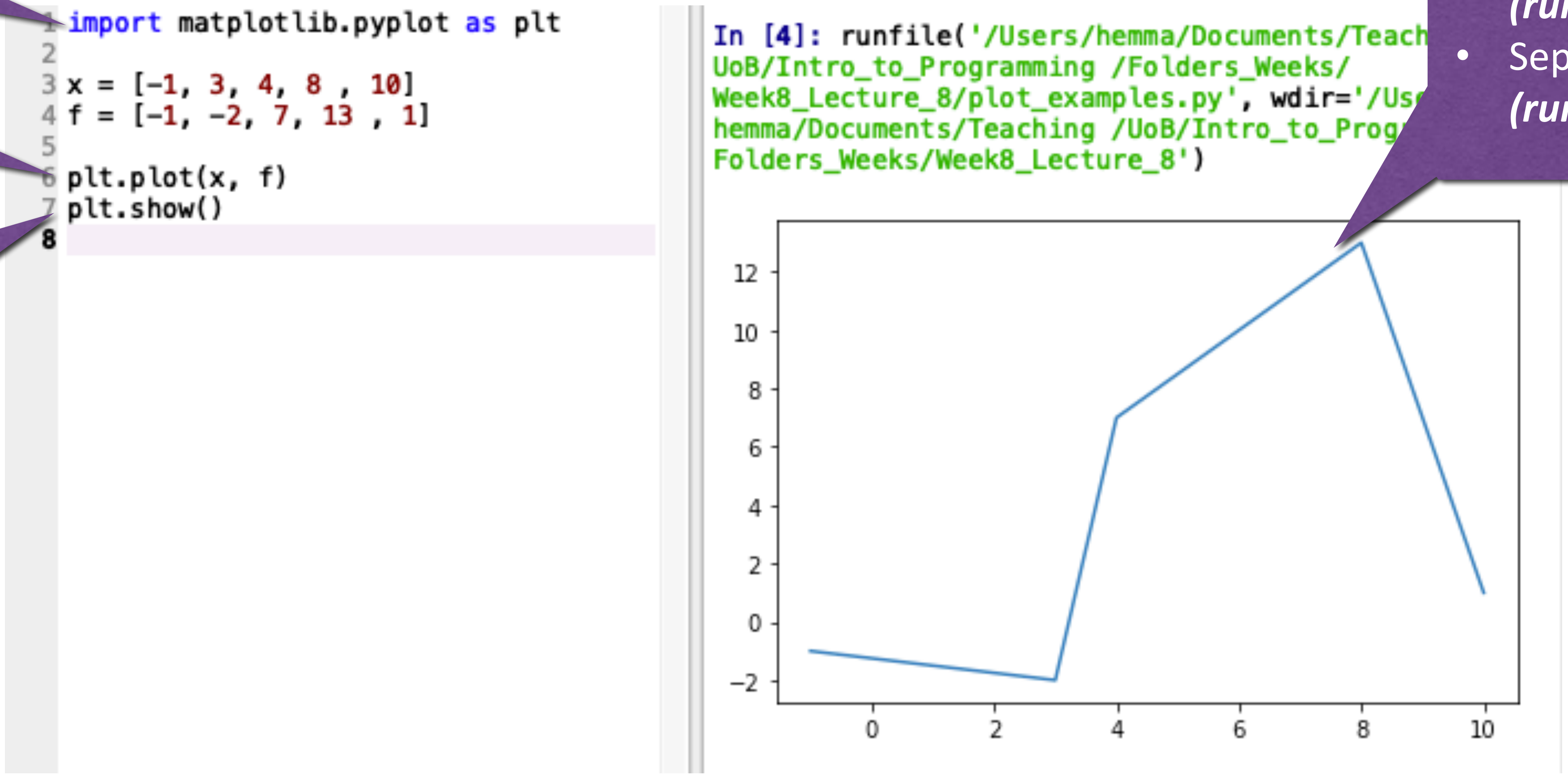
Introduction to Computer Programming Lecture 8.1:

Intro to Matplotlib

Hemma Philamore

Department of Engineering Mathematics

Visualising data



```
import matplotlib.pyplot as plt

x = [-1, 3, 4, 8 , 10]
f = [-1, -2, 7, 13 , 1]

plt.plot(x, f)
plt.show()
```

```
In [4]: runfile('/Users/hemma/Documents/Teach
UoB/Intro_to_Programming /Folders_Weeks/
Week8_Lecture_8/plot_examples.py', wdir='/Us
hemma/Documents/Teaching /UoB/Intro_to_Prog
Folders_Weeks/Week8_Lecture_8')
```

Import the module

Line plot

Required to display plot

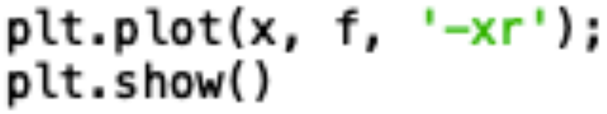
Output displayed in:

- Plots window *(run in Spyder)*
- Separate window *(run in terminal)*

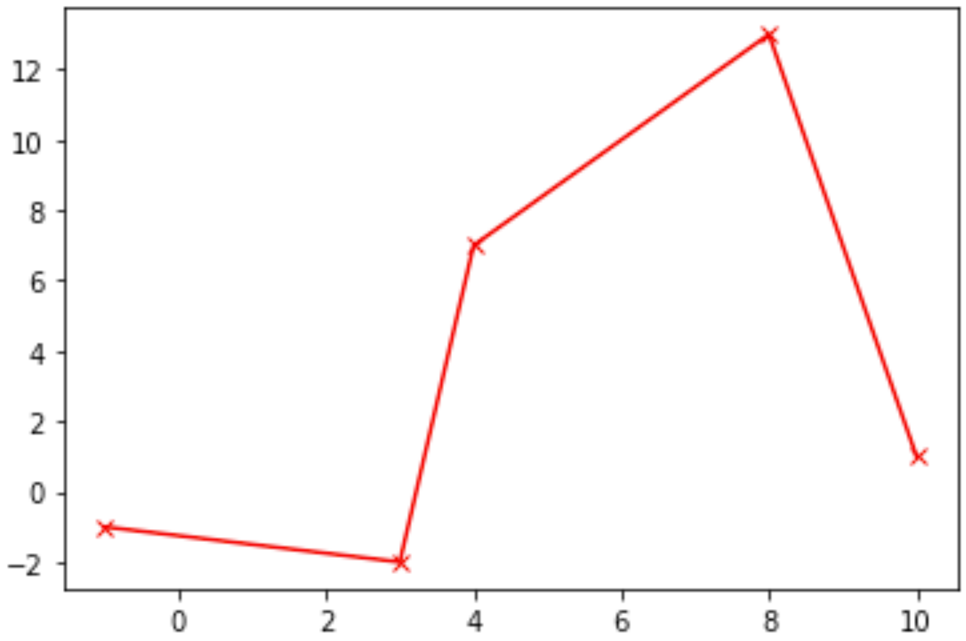
Formatting

Optional format string:

- the colour of the plot (e.g. r = red, k= black)
- the style of the markers (e.g. o = points, * = stars)
- the style of the line (e.g. -- = dashes, . = dots)



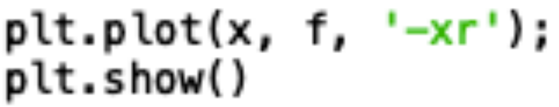
```
plt.plot(x, f, '-xr');
plt.show()
```



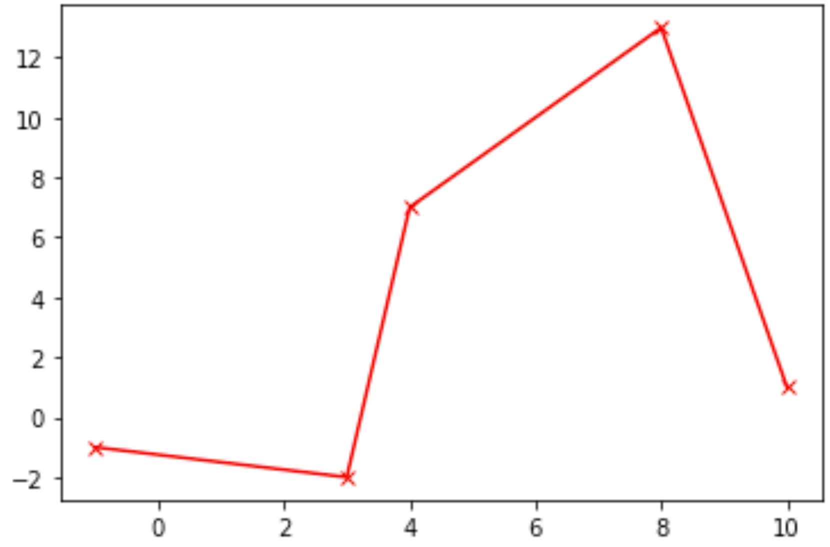
Formatting

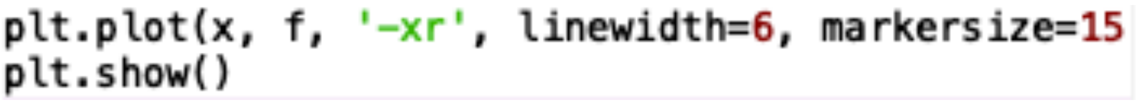
Optional format string:

- the colour of the plot (e.g. r = red, k= black)
- the style of the markers (e.g. o = points, * = stars)
- the style of the line (e.g. -- = dashes, . = dots)

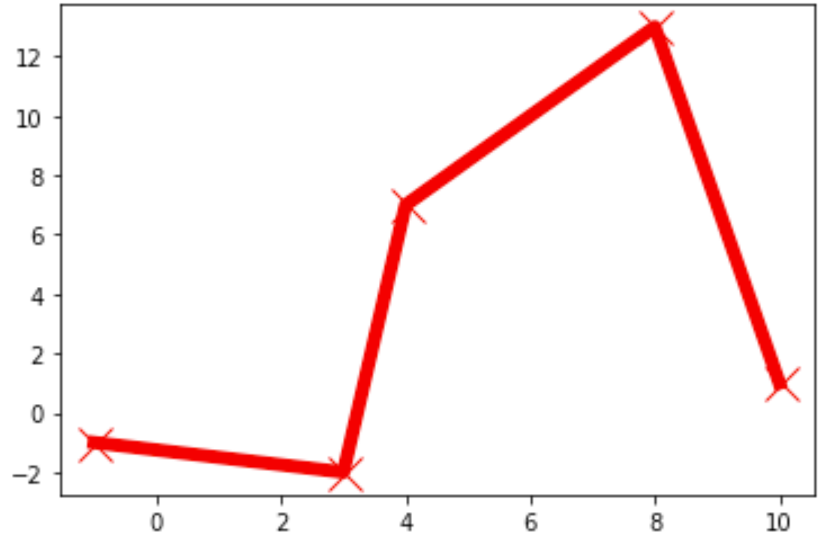


```
plt.plot(x, f, '-xr');
plt.show()
```

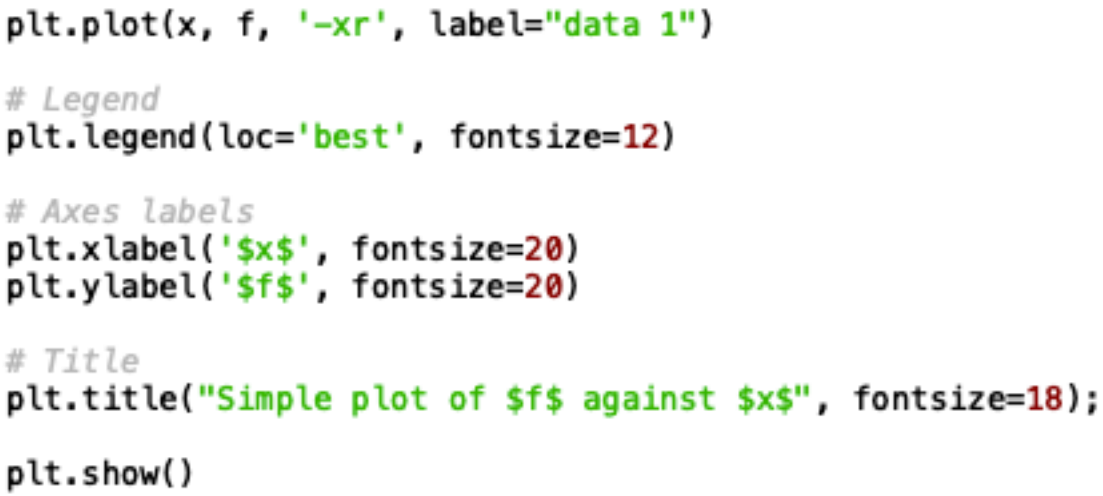




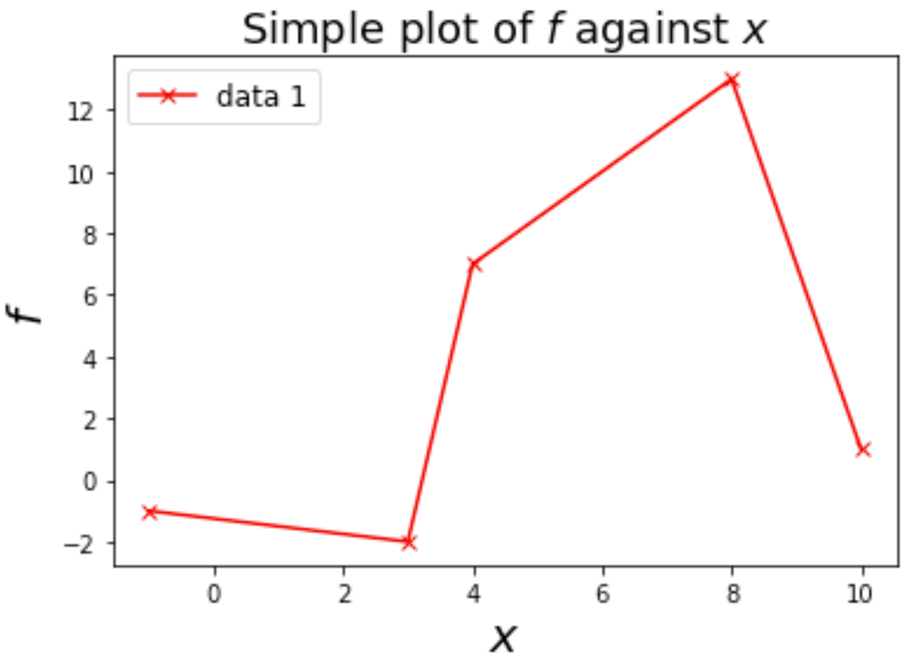
```
plt.plot(x, f, '-xr', linewidth=6, markersize=15
plt.show()
```



Axis Labels, Legend and Title



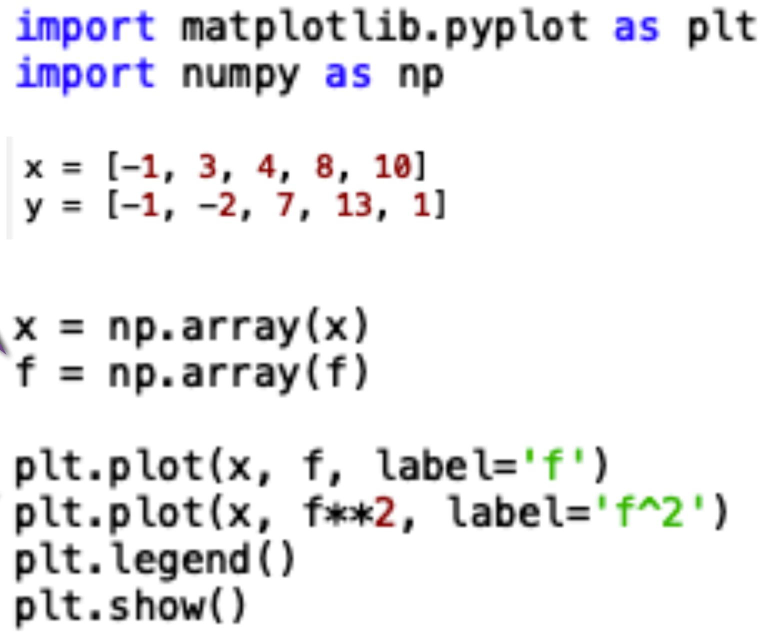
```
plt.plot(x, f, '-xr', label="data 1")

# Legend
plt.legend(loc='best', fontsize=12)

# Axes labels
plt.xlabel('$x$', fontsize=20)
plt.ylabel('$f$', fontsize=20)

# Title
plt.title("Simple plot of $f$ against $x$", fontsize=18);

plt.show()
```



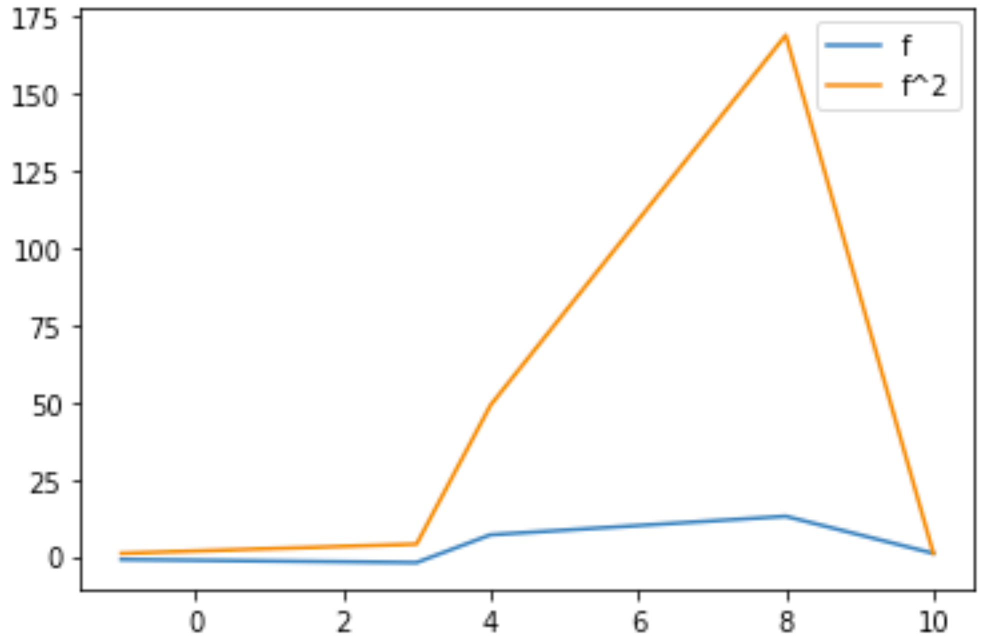
Multiple plots

Converting lists to arrays allows elementwise operations

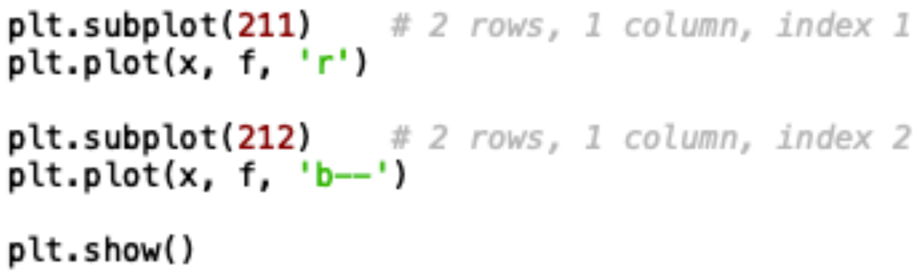
Plots displayed on same axes



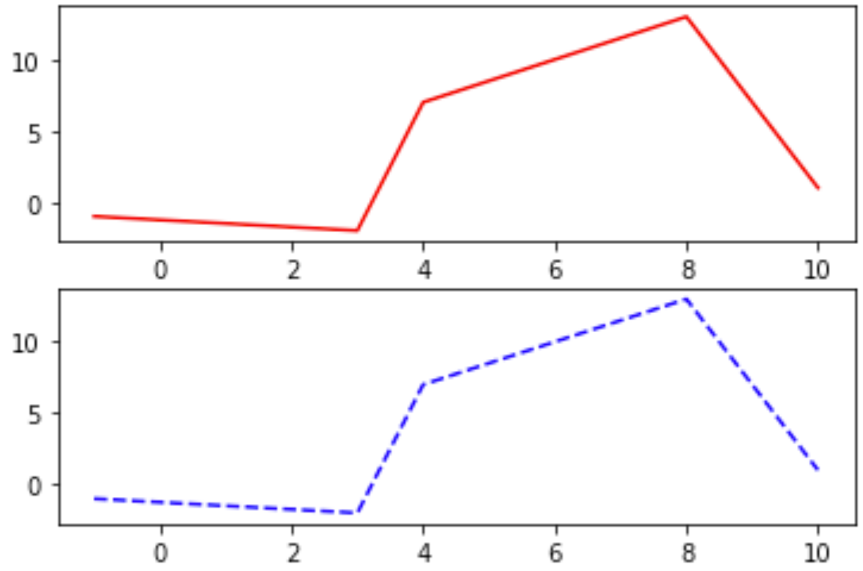
```
import matplotlib.pyplot as plt
import numpy as np

x = [-1, 3, 4, 8, 10]
y = [-1, -2, 7, 13, 1]


x = np.array(x)
f = np.array(f)

plt.plot(x, f, label='f')
plt.plot(x, f**2, label='f^2')
plt.legend()
plt.show()
```



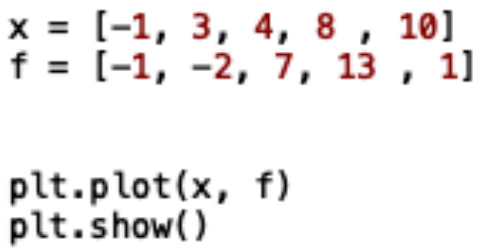
Sub-plots



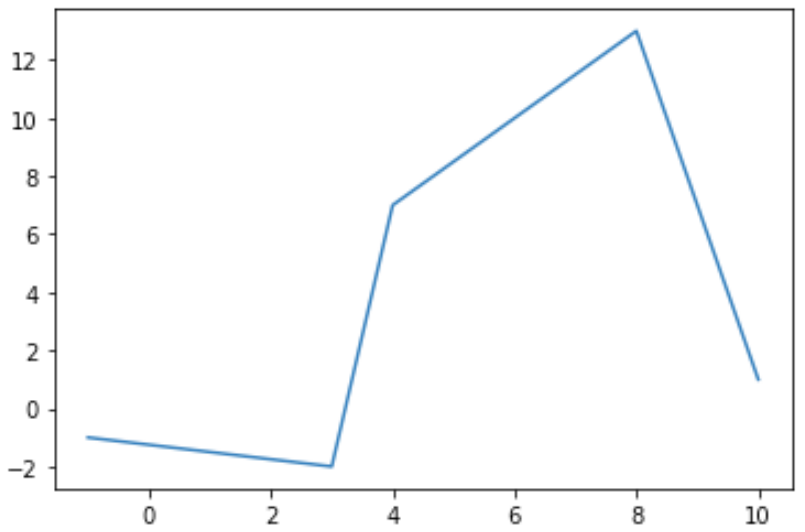
```
plt.subplot(211)    # 2 rows, 1 column, index 1
plt.plot(x, f, 'r')

plt.subplot(212)    # 2 rows, 1 column, index 2
plt.plot(x, f, 'b--')

plt.show()
```

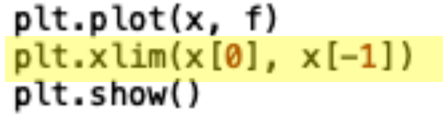


Setting Axis Limits

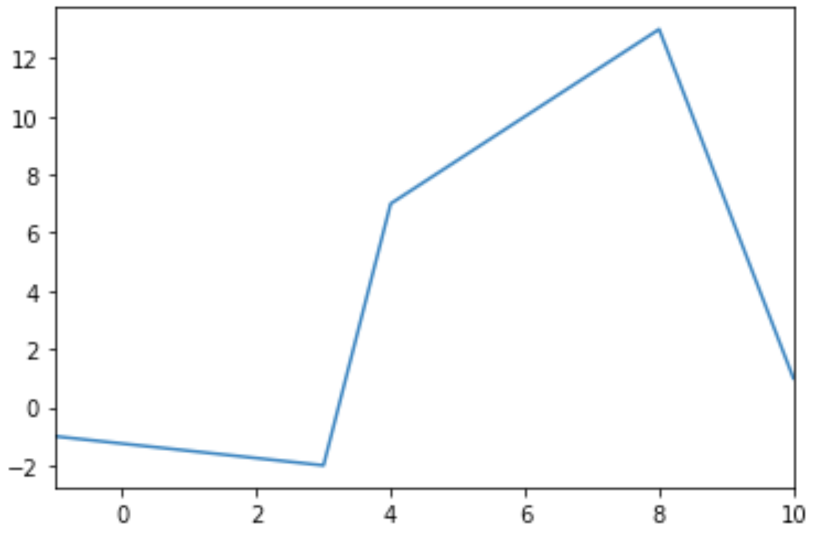


```
x = [-1, 3, 4, 8 , 10]
f = [-1, -2, 7, 13 , 1]


plt.plot(x, f)
plt.show()
```

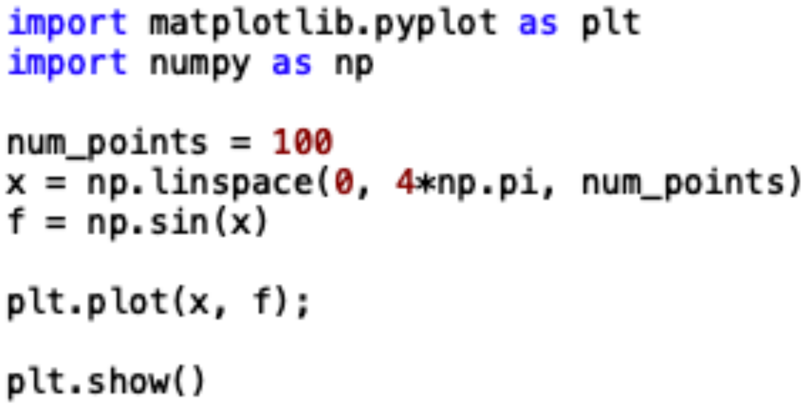




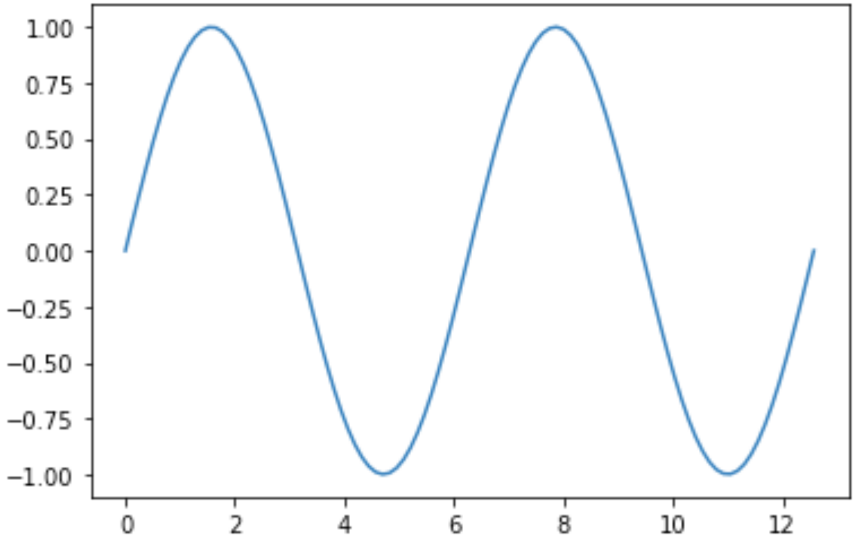
```
plt.plot(x, f)
plt.xlim(x[0], x[-1])
plt.show()
```

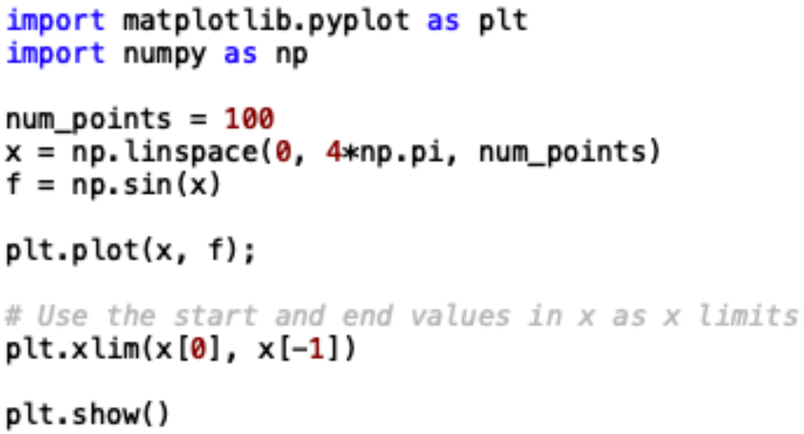


Setting Axis Limits

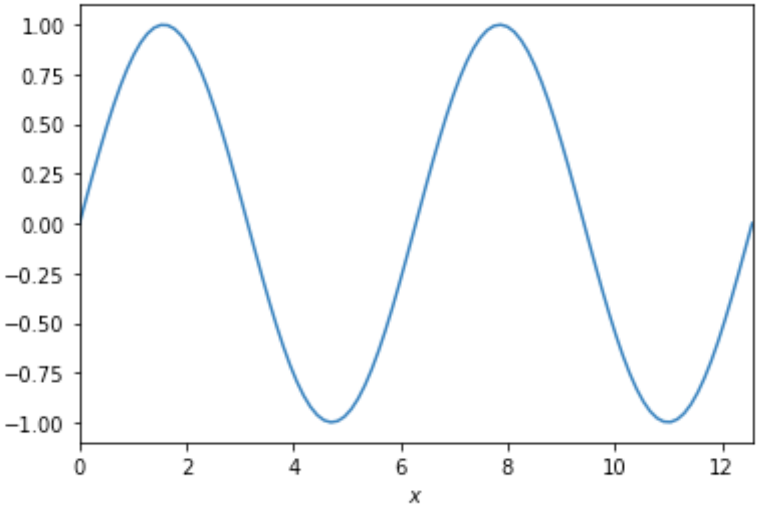


```
import matplotlib.pyplot as plt
import numpy as np

num_points = 100
x = np.linspace(0, 4*np.pi, num_points)
f = np.sin(x)

plt.plot(x, f);

plt.show()
```





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

num_points = 100
x = np.linspace(0, 4*np.pi, num_points)
f = np.sin(x)

plt.plot(x, f);

# Use the start and end values in x as x limits
plt.xlim(x[0], x[-1])

plt.show()
```

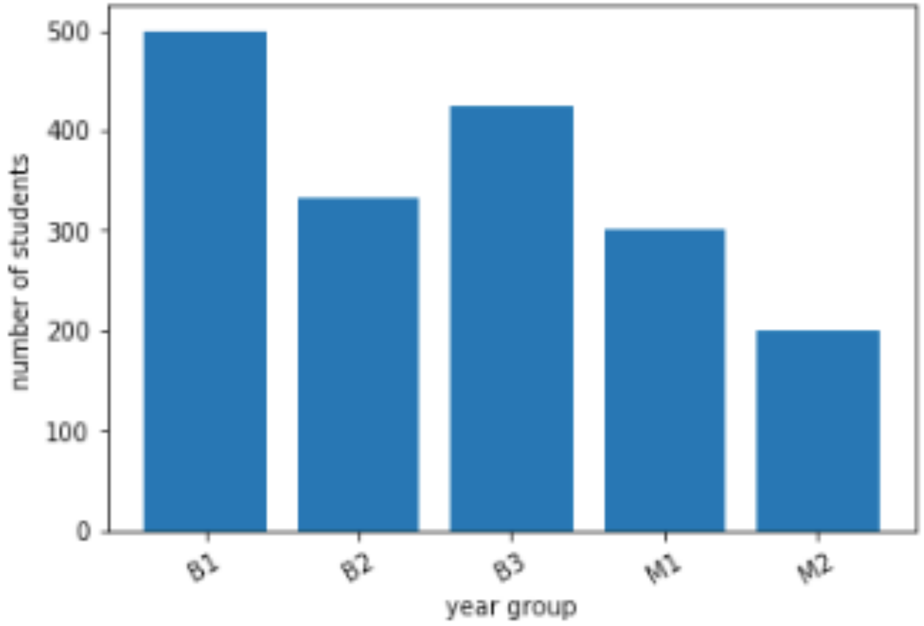


Bar Chart

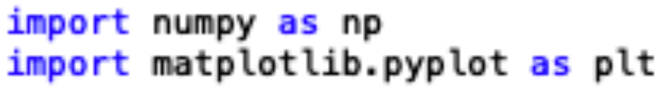
To represent data as a bar chart, for example, the number of students in each year of a degree program:

```
year_groups = ('B1', 'B2', 'B3', 'M1', 'M2')
num_students = (500, 332, 425, 300, 200)
```

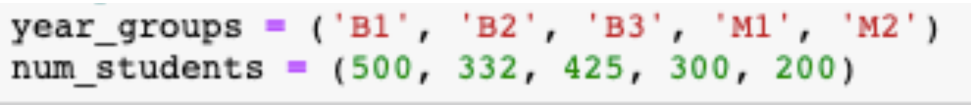
1. Create an array of the position of each bar along the x-axis
2. Produce bar plot of data vs position on x axis
3. Replace the x ticks with the field name
4. Add axis labels



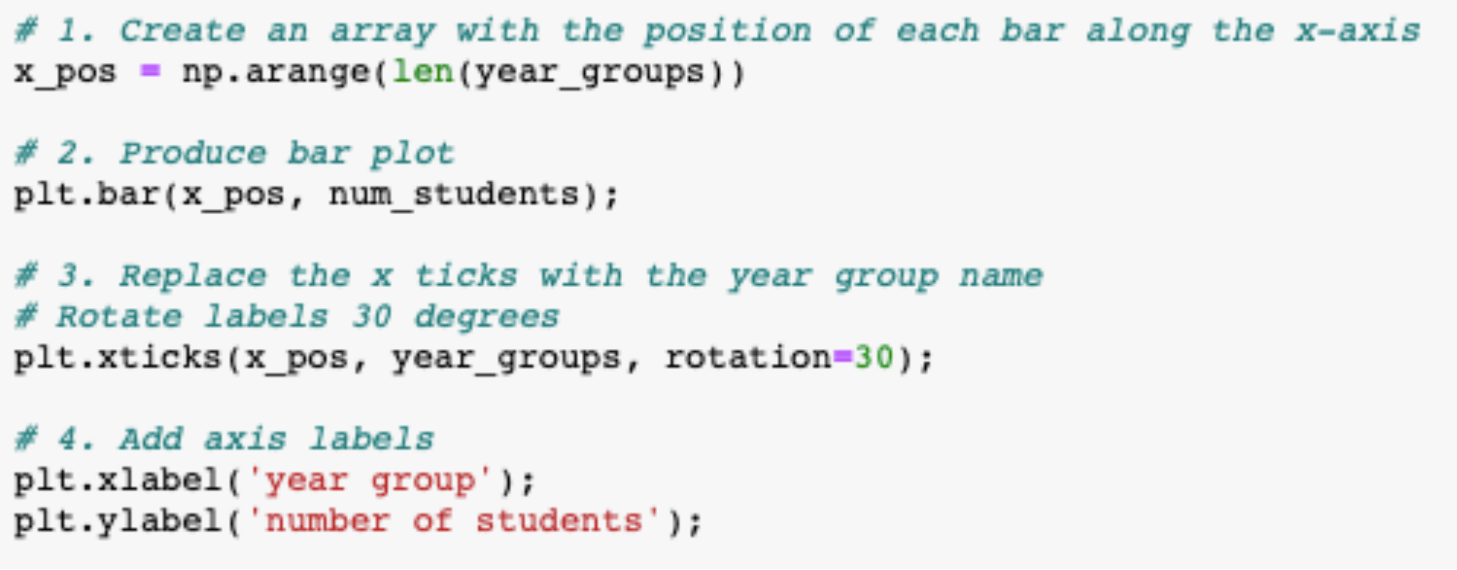
Bar Chart



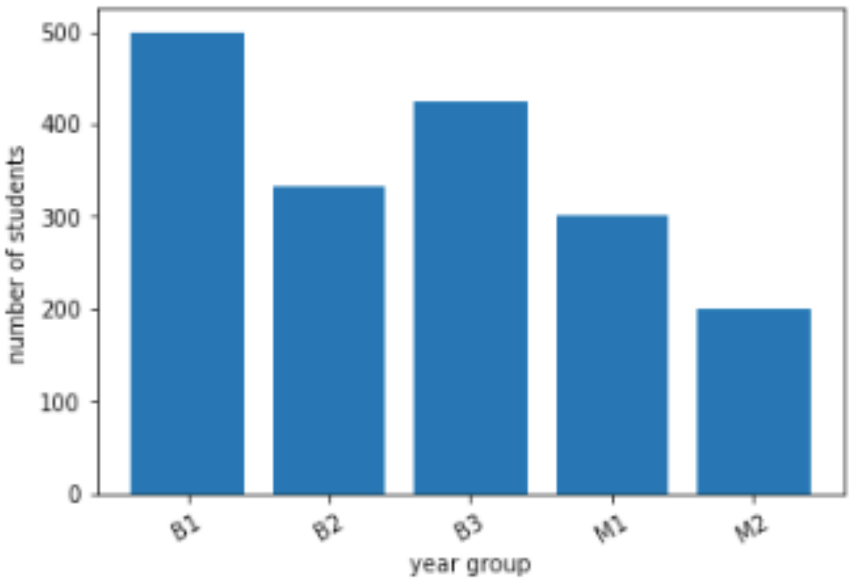
```
import numpy as np
import matplotlib.pyplot as plt
```



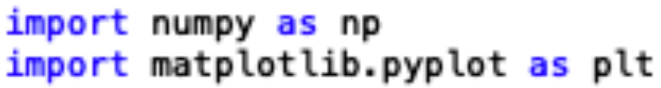
```
year_groups = ('B1', 'B2', 'B3', 'M1', 'M2')
num_students = (500, 332, 425, 300, 200)
```



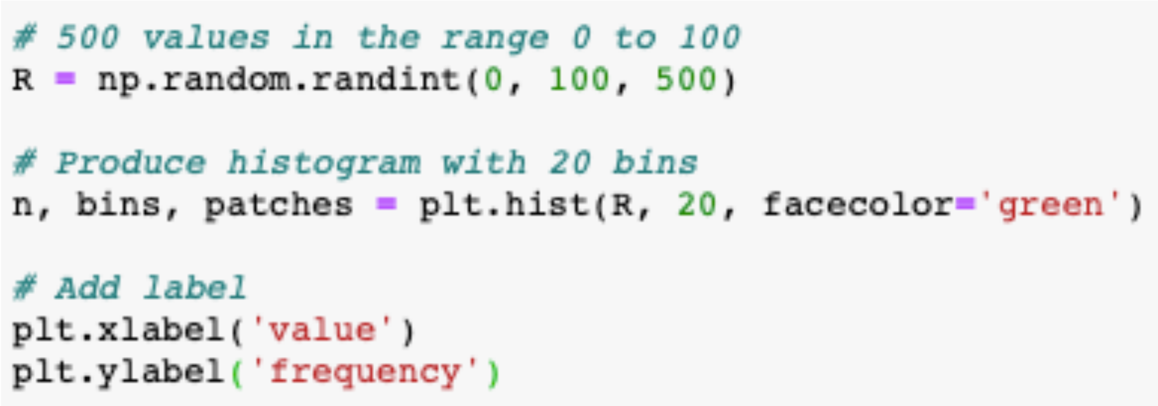
```
# 1. Create an array with the position of each bar along the x-axis
x_pos = np.arange(len(year_groups))

# 2. Produce bar plot
plt.bar(x_pos, num_students);

# 3. Replace the x ticks with the year group name
# Rotate labels 30 degrees
plt.xticks(x_pos, year_groups, rotation=30);

# 4. Add axis labels
plt.xlabel('year group');
plt.ylabel('number of students');
```



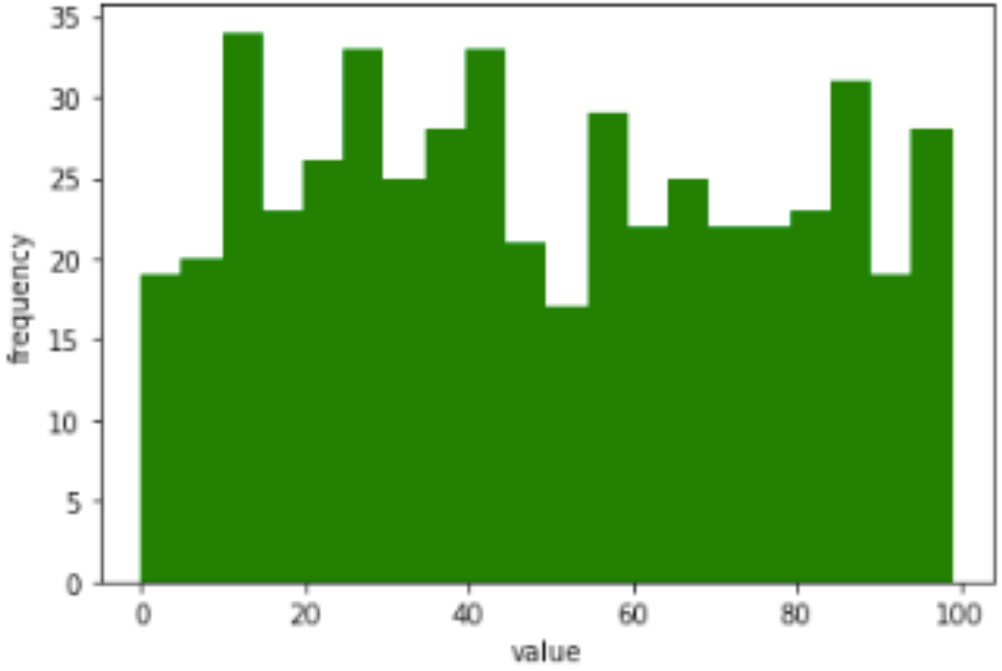
Histogram



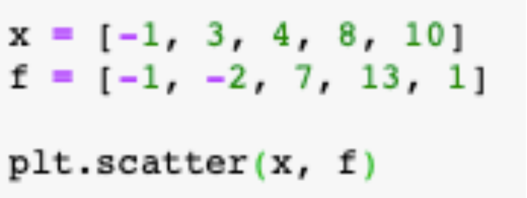
```python
import numpy as np
import matplotlib.pyplot as plt
```



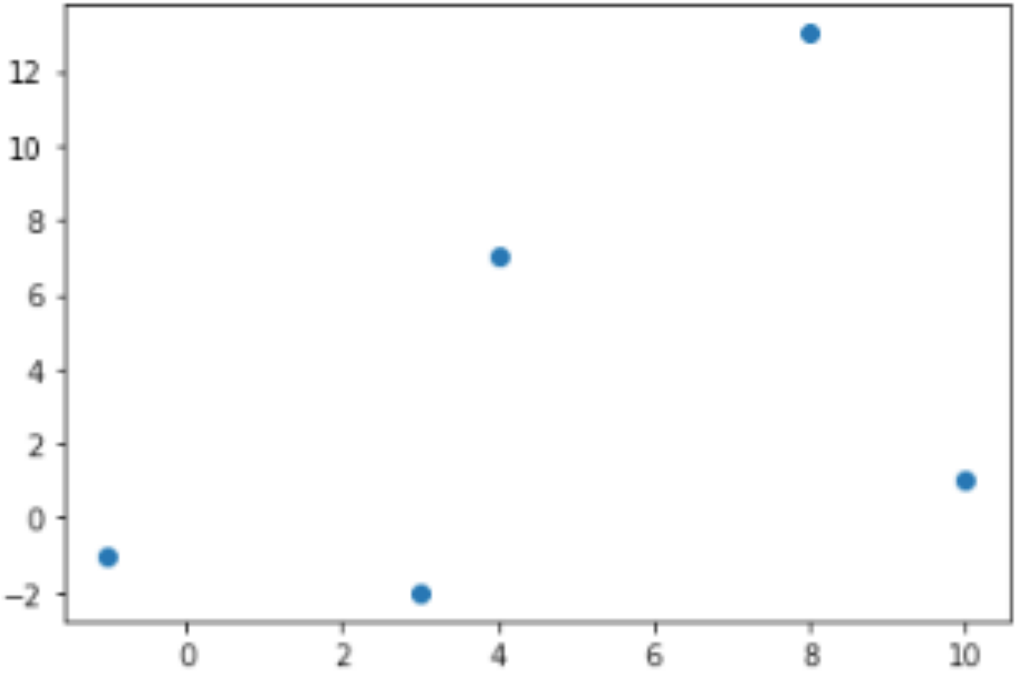
```python
# 500 values in the range 0 to 100
R = np.random.randint(0, 100, 500)

# Produce histogram with 20 bins
n, bins, patches = plt.hist(R, 20, facecolor='green')

# Add label
plt.xlabel('value')
plt.ylabel('frequency')
```



Scatter



```
x = [-1, 3, 4, 8, 10]
f = [-1, -2, 7, 13, 1]

plt.scatter(x, f)
```



12
10
8
6
4
2
0
−2
0
2
4
6
8
10