Introduction to Computer Programming
Lecture 11.4:

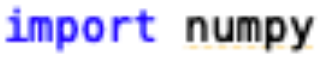
Review :
Imported modules – Numpy & Matplotlib

Hemma Philamore

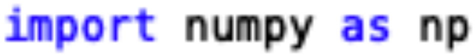
Department of Engineering Mathematics

Import

Import modules using **import** keyword

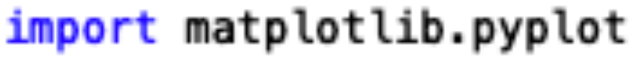


```
import numpy
```



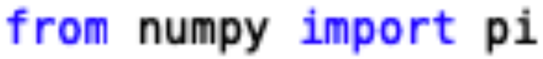
```
import numpy as np
```

Renaming



```
import matplotlib.pyplot
```

Importing sub-modules



```
from numpy import pi
```

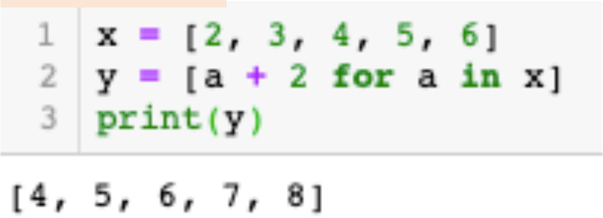
Importing specific functions, variables

Numpy

Numpy Array

Elementwise operation

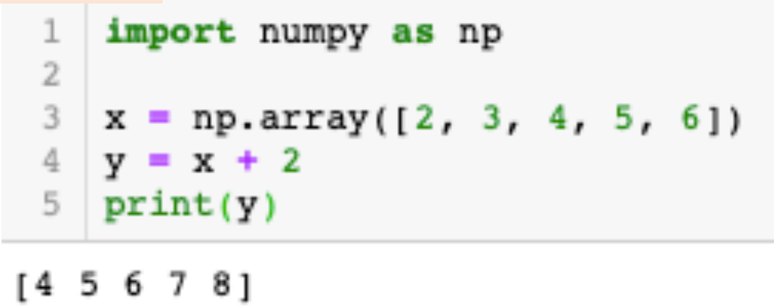
Pure Python



```
x = [2, 3, 4, 5, 6]
y = [a + 2 for a in x]
print(y)
```

```
[4, 5, 6, 7, 8]
```

Numpy

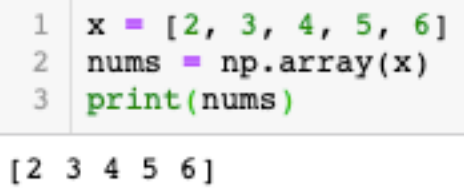


```
import numpy as np

x = np.array([2, 3, 4, 5, 6])
y = x + 2
print(y)
```

```
[4 5 6 7 8]
```

Creating Numpy arrays

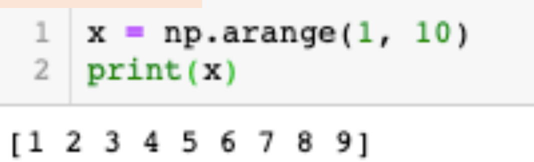
Create 1D array from list



```
x = [2, 3, 4, 5, 6]
nums = np.array(x)
print(nums)
```

```
[2 3 4 5 6]
```

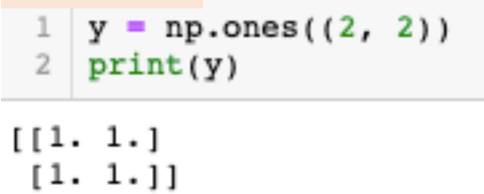
arange



```
x = np.arange(1, 10)
print(x)
```

```
[1 2 3 4 5 6 7 8 9]
```

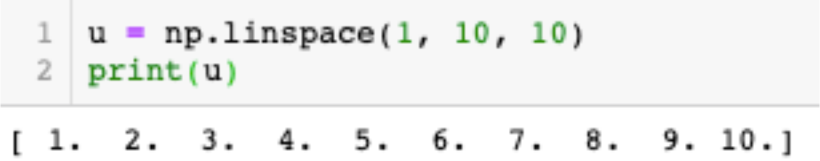
ones



```
y = np.ones((2, 2))
print(y)
```

```
[[1. 1.]
 [1. 1.]]
```

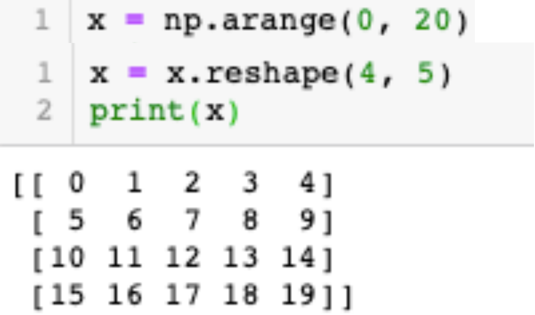
linspace



```
u = np.linspace(1, 10, 10)
print(u)
```

```
[ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
```

Reshape



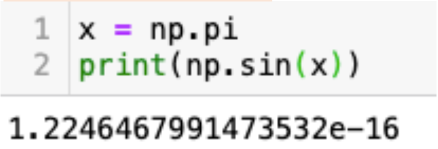
```
x = np.arange(0, 20)
```

```
x = x.reshape(4, 5)
print(x)
```

```
[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]
 [15 16 17 18 19]]
```

Numpy Functionality

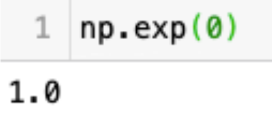
Trigonometry



```
x = np.pi
print(np.sin(x))
```

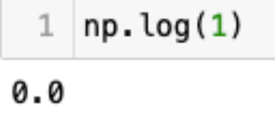
```
1.2246467991473532e-16
```

Exponents and logarithms



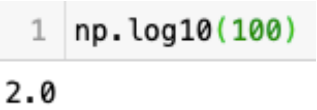
```
np.exp(0)
```

```
1.0
```



```
np.log(1)
```

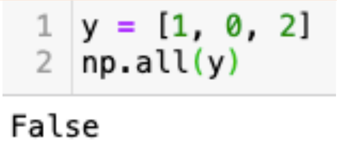
```
0.0
```



```
np.log10(100)
```

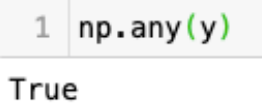
```
2.0
```

Test if **all** or **any** elements in a data structure are non-zero



```
y = [1, 0, 2]
np.all(y)
```

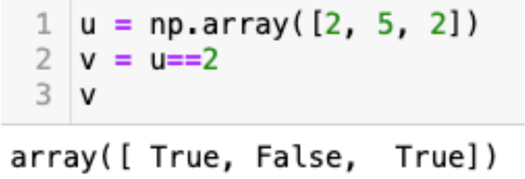
```
False
```



```
np.any(y)
```

```
True
```

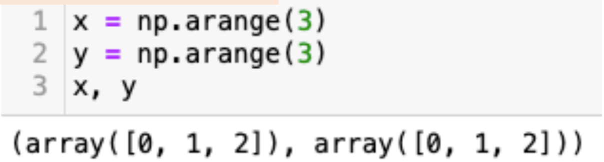
Find instances of a value in an array



```
u = np.array([2, 5, 2])
v = u==2
v
```

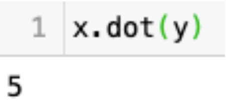
```
array([ True, False,  True])
```

Linear algebra



```
x = np.arange(3)
y = np.arange(3)
x, y
```

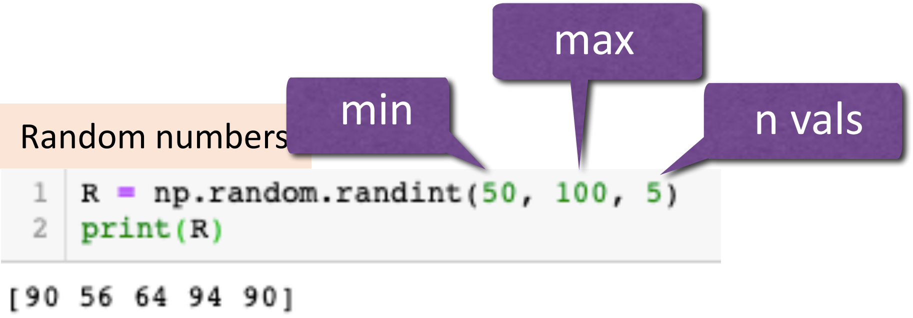
```
(array([0, 1, 2]), array([0, 1, 2]))
```



```
x.dot(y)
```

```
5
```

Random numbers



max

min

n vals

```
R = np.random.randint(50, 100, 5)
print(R)
```

```
[90 56 64 94 90]
```

Q.11.4.A

Create the numpy array:

```
[[6,    8,   10],
 [12, 14, 16],
 [18, 20, 22]]
```

Q.11.4.B

Find sin(x) for each value, x, in the list: $\left[\frac{\pi}{2}, \pi, \frac{\pi}{4}\right]$

Q.11.4.C

Add [6, 4, 2] to each row of the array in your answer to Q.11.4.A to get:

```
[[12,    12,   12],
 [18,    18,   18],
 [24,    24,   24]]
```

Q.11.4.D

Find the dot product of the four elements in the upper left corners of the 3x3 arrays in Q.11.4.A and Q.11.4.C

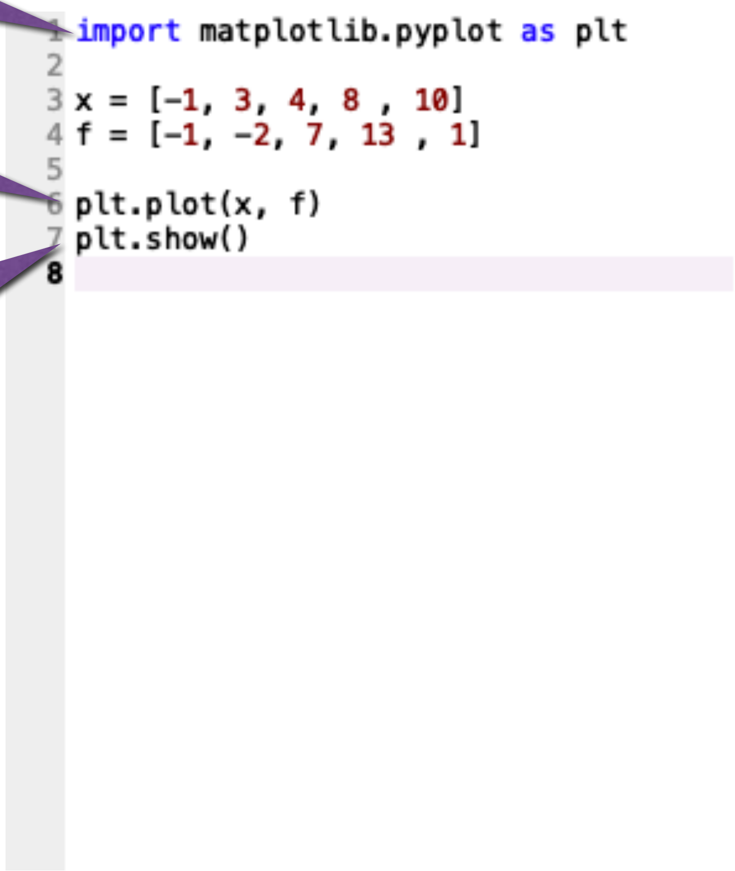
Matplotlib

Visualising data

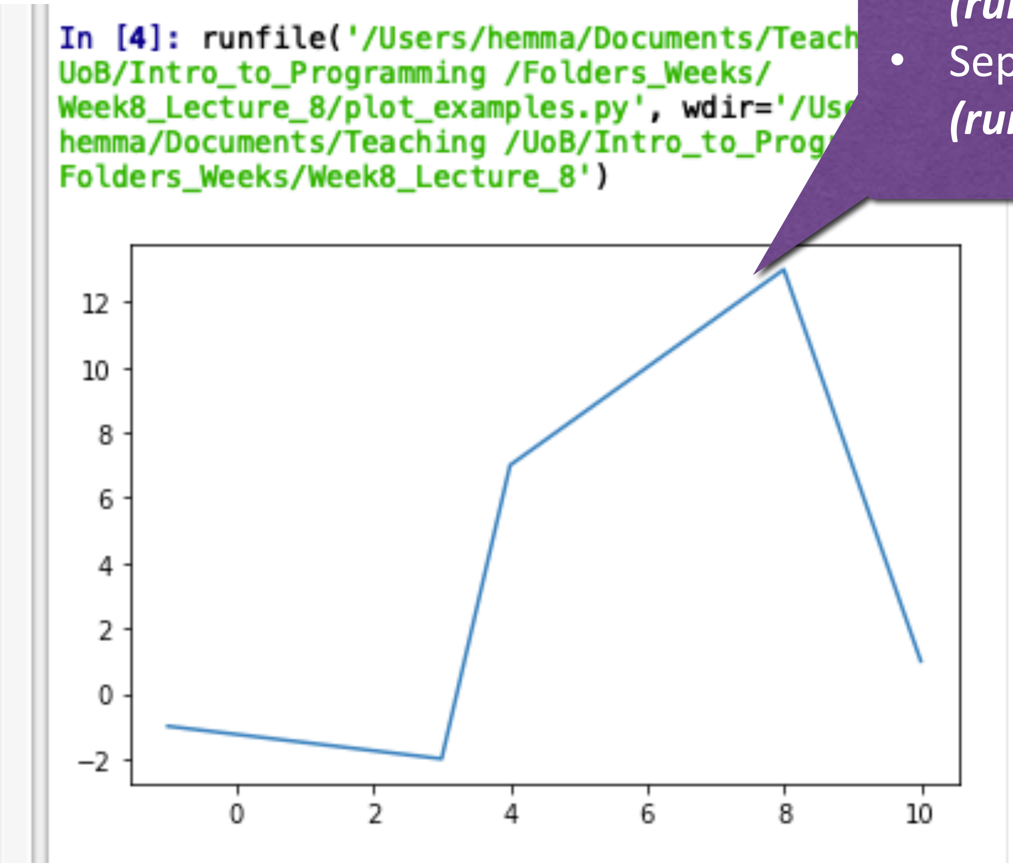
Import the module

Line plot

Required to display plot



```python
import matplotlib.pyplot as plt

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

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

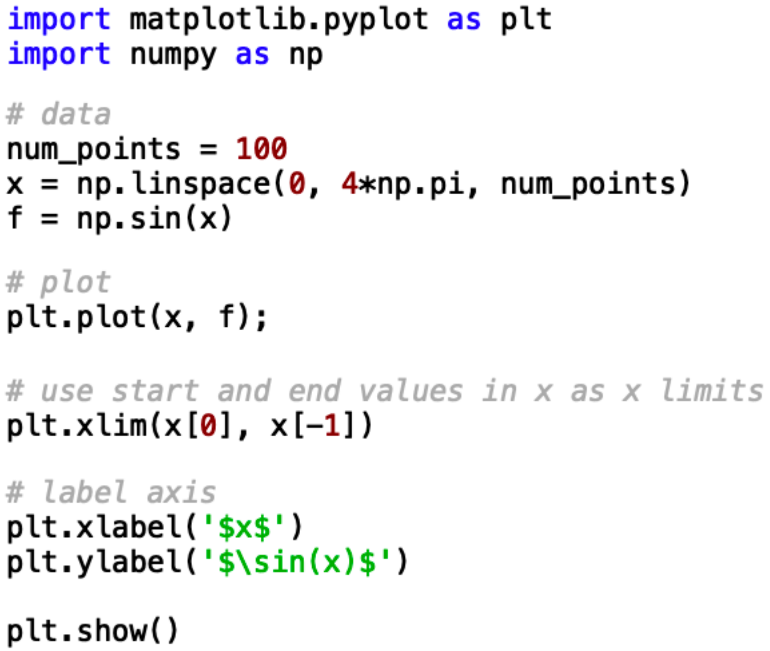


```
In [4]: runfile('/Users/hemma/Documents/Teach
UoB/Intro_to_Programming /Folders_Weeks/
Week8_Lecture_8/plot_examples.py', wdir='/Use
hemma/Documents/Teaching /UoB/Intro_to_Prog
Folders_Weeks/Week8_Lecture_8')
```

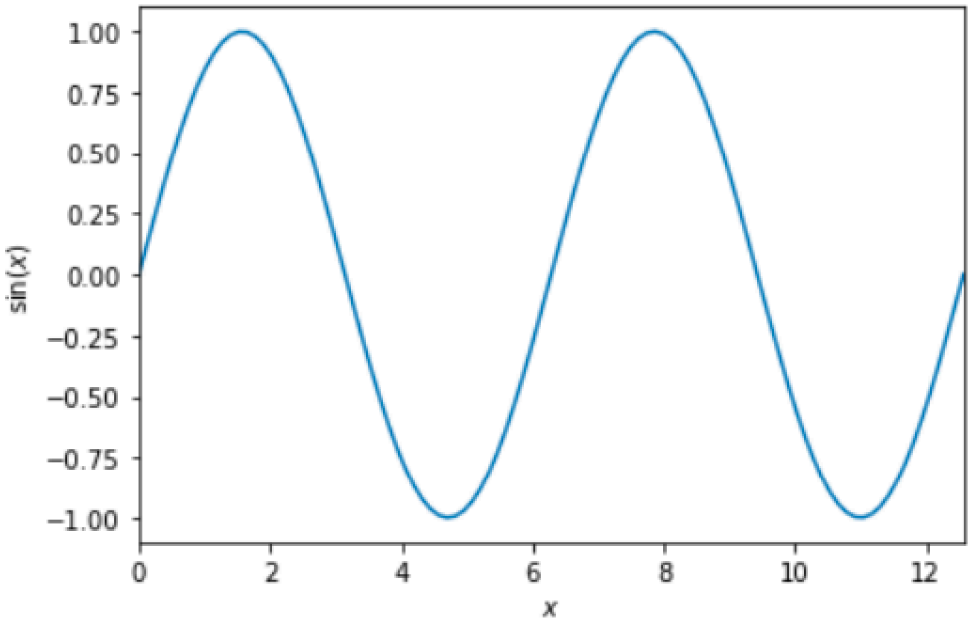
Output displayed in:

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

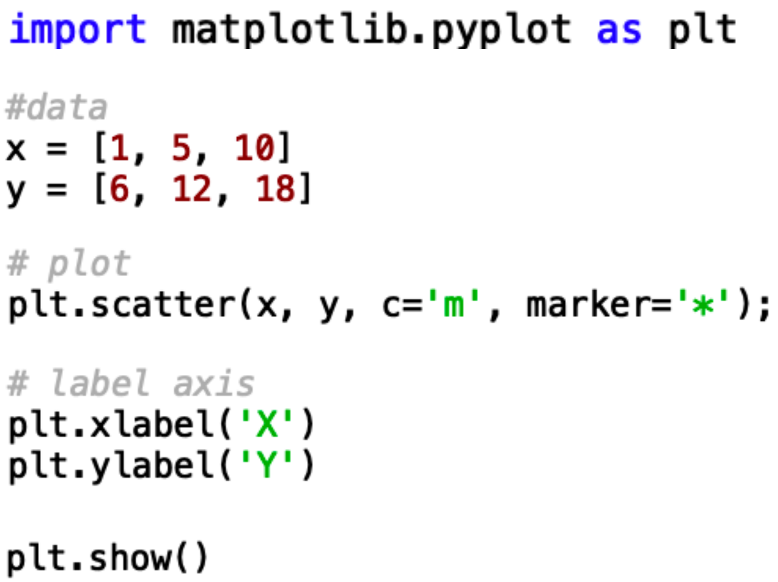
Line plot



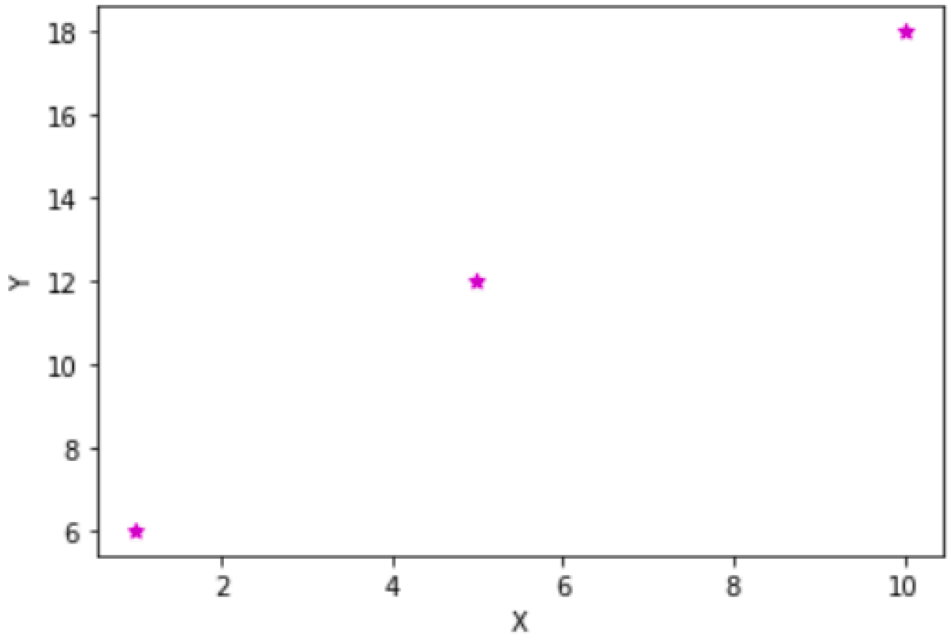
```python
import matplotlib.pyplot as plt
import numpy as np

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

# plot
plt.plot(x, f);

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

# label axis
plt.xlabel('$x$')
plt.ylabel('$\sin(x)$')

plt.show()
```



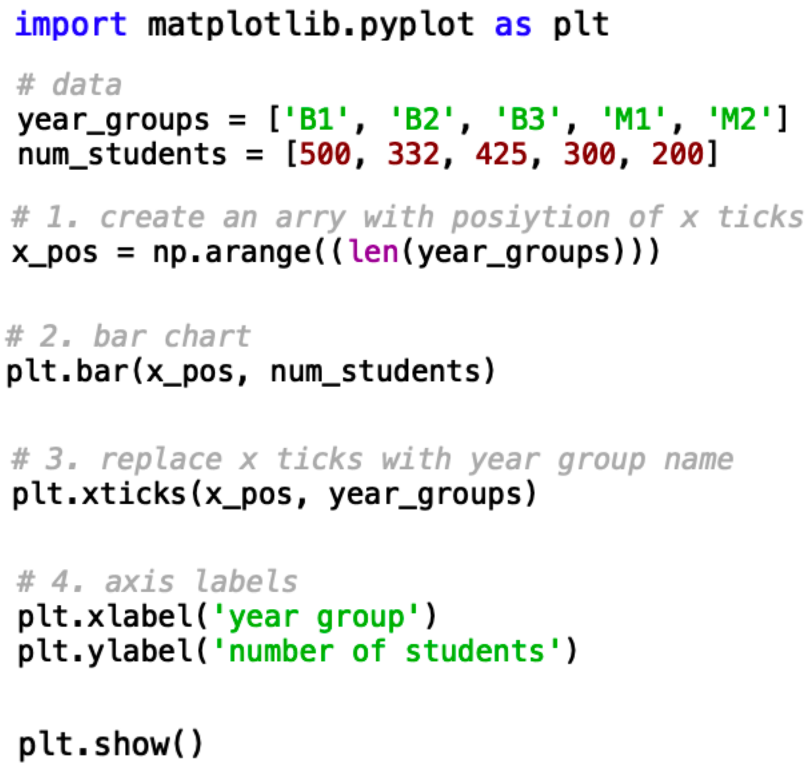
Scatter plot



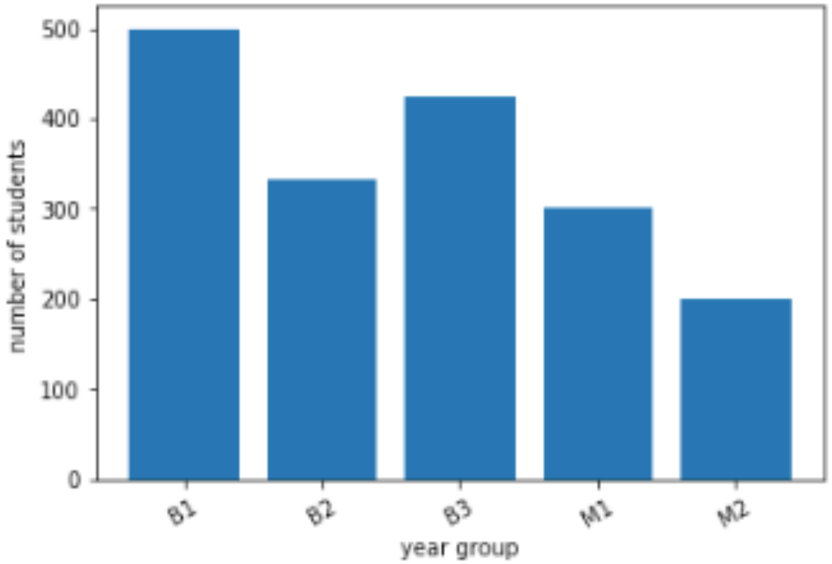
```
import matplotlib.pyplot as plt

#data
x = [1, 5, 10]
y = [6, 12, 18]

# plot
plt.scatter(x, y, c='m', marker='*');

# label axis
plt.xlabel('X')
plt.ylabel('Y')

plt.show()
```



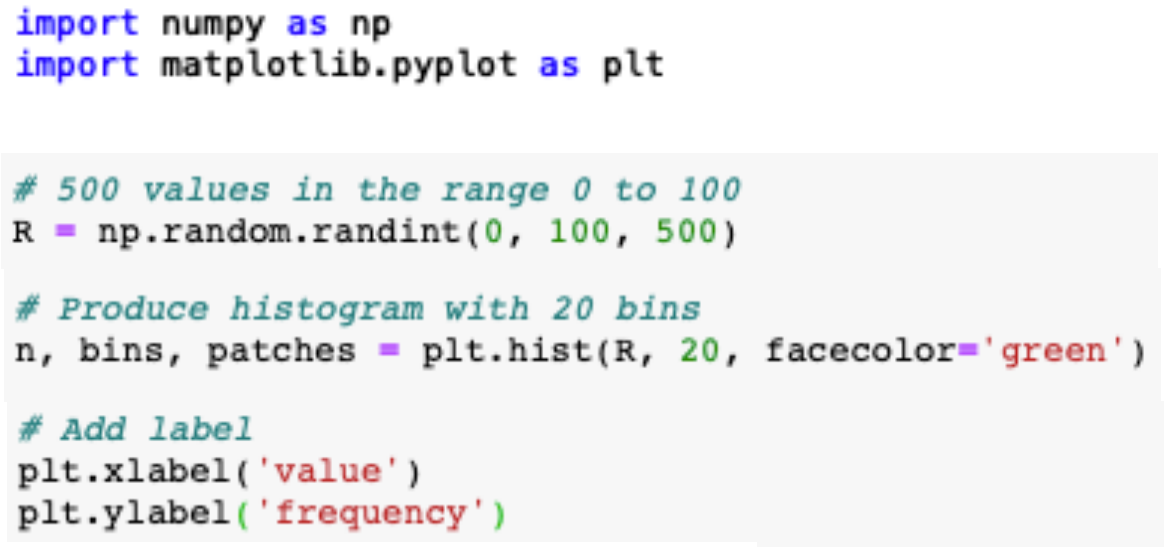
Bar Chart



```
import matplotlib.pyplot as plt

# data
year_groups = ['B1', 'B2', 'B3', 'M1', 'M2']
num_students = [500, 332, 425, 300, 200]

# 1. create an arry with posiytion of x ticks
x_pos = np.arange((len(year_groups)))

# 2. bar chart
plt.bar(x_pos, num_students)

# 3. replace x ticks with year group name
plt.xticks(x_pos, year_groups)

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

plt.show()
```

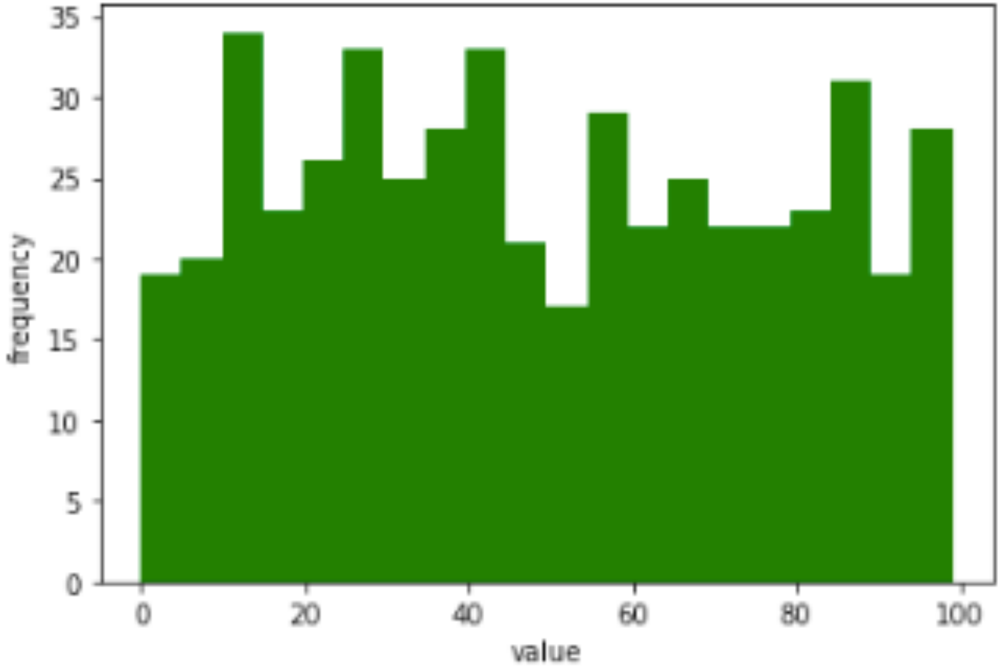


Histogram



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

# 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')
```

```
plt.show()
```



Q.11.4.E

Create a line plot of row 1 against row 0 of the numpy array:

```
[[6,   8,  10],
 [18, 23, 12]]
```

Q.11.4.F

Create a scatter plot of the exponential function, e^{x}, for x in the range 0 to 10 inclusive.

Going further with Numpy, Matplotlib and other python modules

Numpy
Mathematical modelling, scientific computing

Matplotlib
Overlaying plots e.g. error bars, subplots, 3D plotting, simulation, animated plots

Scipy : Numerical routines e.g numerical integration, interpolation, optimization, linear algebra, statistics
Sympy : Symbolic mathematics
Pandas : Data analysis and manipulation