In [117…
```
import numpy as np
import pandas as pd
import scipy as sp
```

In [118…
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('ggplot')
```

In [ ]:
```
%%file hw_data.csv
id,sex,weight,height
1,M,190,77
2,F,120,70
3,F,110,68
4,M,150,72
5,O,120,66
6,M,120,60
7,F,140,70
```

Python

1. Finish creating the following function that takes a list and returns the average value.

Add each element in the list to `total` and return `total`

DO NOT use a library function nor `sum()`

In [119…
```
def average(my_list):
    total = 0
    for item in my_list:
        total = (total+item)
        average = total/len(my_list)
    return average

average([1,2,1,4,3,2,5,9])
```

Out[119]:
```
3.375
```

2. Using a Dictionary keep track of the count of numbers (or items) from a list

In [120…
```
def counts(my_list):
    counts = dict()
    for item in my_list:
        counts[item] = counts.get(item, 0) + 1

    return counts
```

```
counts([1,2,1,4,3,2,5,9])
```

Out[120]:
```
{1: 2, 2: 2, 4: 1, 3: 1, 5: 1, 9: 1}
```

3. Using the `counts()` function you created above and the `.split()` function, return a dictionary of most occuring words from the following paragraph. Bonus, remove punctuation from words.

In [121…
```
paragraph_text = '''
For a minute or two she stood looking at the house, and wondering what to do next, whe
The Fish-Footman began by producing from under his arm a great letter, nearly as large
Then they both bowed low, and their curls got entangled together.
Alice laughed so much at this, that she had to run back into the wood for fear of thei
Alice went timidly up to the door, and knocked.
‘There’s no sort of use in knocking,’ said the Footman, ‘and that for two reasons. Fir
‘Please, then,’ said Alice, ‘how am I to get in?’
‘There might be some sense in your knocking,’ the Footman went on without attending to
‘I shall sit here,’ the Footman remarked, ‘till tomorrow—’
At this moment the door of the house opened, and a large plate came skimming out, stra


for char in '-.);’?:‘—(,\n':
    paragraph_text=paragraph_text.replace(char, ' ')
    paragraph_text=paragraph_text.lower()
paragraph_text2 = counts(paragraph_text.split())
sorted(paragraph_text2.items())
```

Out[121]:
```
[('a', 16),
 ('about', 1),
 ('against', 1),
 ('alice', 5),
 ('all', 3),
 ('aloud', 1),
 ('am', 2),
 ('an', 2),
 ('and', 19),
 ('another', 1),
 ('answer', 1),
 ('any', 1),
 ('are', 2),
 ('arm', 1),
 ('as', 4),
 ('at', 7),
 ('attending', 1),
 ('back', 1),
 ('be', 2),
 ('because', 3),
 ('been', 1),
 ('began', 1),
 ('behind', 1),
 ('between', 1),
 ('both', 2),
 ('bowed', 1),
 ('broke', 1),
 ('broken', 1),
 ('but', 2),
 ('by', 3),
 ('called', 1),
 ('came', 2),
 ('can', 1),
 ('certainly', 1),
 ('changing', 1),
 ('considered', 1),
 ('constant', 1),
 ('could', 2),
 ('crash', 1),
 ('crept', 1),
 ('croquet', 2),
 ('curious', 1),
 ('curled', 1),
 ('curls', 1),
 ('decidedly', 1),
 ('dish', 1),
 ('do', 1),
 ('door', 6),
 ('duchess', 2),
 ('entangled', 1),
 ('every', 1),
 ('extraordinary', 1),
 ('eyes', 2),
 ('face', 2),
 ('fear', 1),
 ('felt', 1),
 ('first', 1),
 ('fish', 3),
 ('footman', 10),
 ('footmen', 1),
```

```
('for', 6),
('frog', 2),
('from', 3),
('get', 2),
('going', 1),
('gone', 1),
('got', 1),
('grazed', 1),
('great', 2),
('ground', 1),
('had', 4),
('hair', 1),
('handed', 1),
('have', 1),
('he', 6),
('head', 2),
('heads', 1),
('hear', 1),
('hearing', 1),
('help', 1),
('her', 2),
('here', 1),
('herself', 1),
('him', 3),
('himself', 1),
('his', 6),
('house', 2),
('how', 2),
('howling', 1),
('i', 5),
('if', 3),
('in', 9),
('inside', 2),
('instance', 1),
('into', 3),
('invitation', 2),
('it', 4),
('judging', 1),
('just', 1),
('kettle', 1),
('knock', 1),
('knocked', 1),
('knocking', 2),
('know', 2),
('knuckles', 1),
('large', 3),
('laughed', 1),
('let', 1),
('letter', 1),
('like', 1),
('listen', 1),
('little', 2),
('livery', 3),
('looking', 2),
('loudly', 1),
('low', 1),
('m', 1),
('making', 1),
('might', 3),
('minute', 1),
```

```
('moment', 1),
('most', 1),
('much', 1),
('near', 1),
('nearly', 2),
('next', 2),
('no', 2),
('noise', 2),
('nose', 1),
('noticed', 1),
('now', 1),
('of', 9),
('on', 4),
('one', 2),
('only', 2),
('opened', 2),
('or', 2),
('order', 1),
('other', 2),
('otherwise', 1),
('out', 5),
('over', 2),
('peeped', 1),
('perhaps', 1),
('pieces', 2),
('plate', 1),
('play', 2),
('please', 1),
('possibly', 1),
('powdered', 1),
('producing', 1),
('queen', 2),
('questions', 1),
('rapped', 1),
('rate', 1),
('re', 1),
('reasons', 1),
('remarked', 1),
('repeated', 2),
('round', 1),
('run', 1),
('running', 1),
('s', 2),
('said', 3),
('same', 2),
('saying', 1),
('secondly', 1),
('sense', 1),
('shall', 1),
('she', 8),
('side', 1),
('sit', 1),
('sitting', 1),
('skimming', 1),
('sky', 2),
('sneezing', 1),
('so', 2),
('solemn', 2),
('some', 1),
('sort', 1),
```

```
('speaking', 1),
('staring', 1),
('stood', 1),
('straight', 1),
('stupidly', 1),
('such', 1),
('suddenly', 1),
('t', 1),
('that', 3),
('the', 34),
('their', 3),
('then', 3),
('there', 3),
('they', 2),
('this', 4),
('thought', 1),
('till', 1),
('time', 1),
('timidly', 1),
('to', 15),
('together', 1),
('tomorrow', 1),
('tone', 2),
('top', 1),
('trees', 1),
('two', 2),
('uncivil', 1),
('under', 1),
('up', 3),
('us', 1),
('use', 1),
('very', 2),
('was', 8),
('way', 1),
('we', 1),
('went', 2),
('were', 1),
('what', 2),
('when', 2),
('with', 2),
('within', 1),
('without', 1),
('wondering', 1),
('wood', 3),
('words', 1),
('would', 1),
('you', 6),
('your', 1)]
```

4. Read in a file using `open()` and iterated through the file line-by-line write each line from the file to a new file in a `title()`-ized. Create your own file for input

`This is the first line` -> `This Is The First Line`

Hint: There's a function to do this

In [122…
```
%%file module2_readdata.txt
this is the first line
this is the second line
this is the third line
this is the fourth line
this is the fifth line
this is the sixth line
```

```
Overwriting module2_readdata.txt
```

In [123…
```
with open('module2_writedata.txt', 'r') as inp:
    y = inp.read().title()
with open('module2_writedata.txt', 'w') as out:
    out.write(y)

readdata = open('module2_readdata.txt', 'r')
print(readdata.read())
readdata = open('module2_writedata.txt', 'r')
print(readdata.read())
```

```
this is the first line
this is the second line
this is the third line
this is the fourth line
this is the fifth line
this is the sixth line

This Is The First Line
This Is The Second Line
This Is The Third Line
This Is The Fourth Line
This Is The Fifth Line
This Is The Sixth Line
```

Numpy

1. Given a list, find the average using a numpy function.

In [124…
```
simple_list = [1,2,1,4,3,2,5,9]
np.average(simple_list)
```

Out[124]:
```
3.375
```

2. Given two lists of Heights and Weights of individual, calculate the BMI of those individuals, without writing a `for-loop`

In [125…
```
heights = [174, 173, 173, 175, 171]
weights = [88, 83, 92, 74, 77]

np.divide(weights, heights)
```

Out[125]:
```
array([0.50574713, 0.47976879, 0.53179191, 0.42285714, 0.4502924 ])
```

3. Create an array of length 20 filled with random values (between 0 to 1)

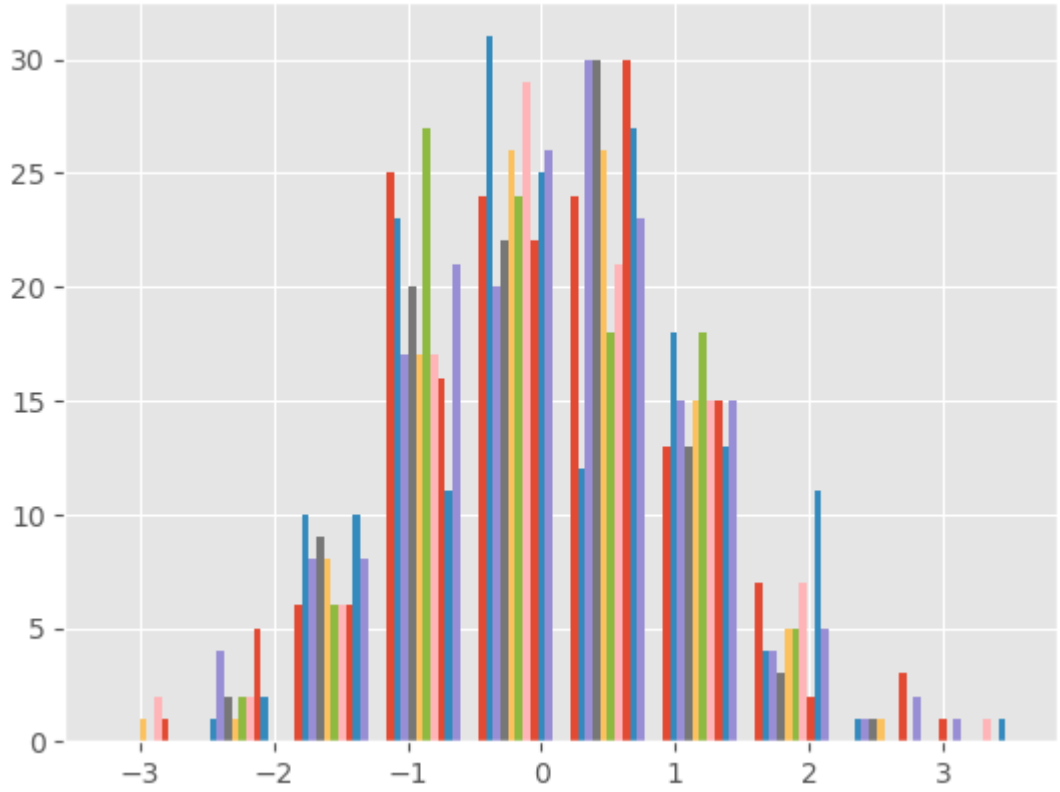
In [126…
```
array1 = np.random.rand(4,5)
print(array1)
```

```
[[0.61122627 0.78005349 0.1911956  0.23397656 0.34987919]
 [0.23280731 0.44555855 0.58956707 0.1481567  0.05363655]
 [0.00320597 0.17639627 0.22268076 0.09053739 0.26645949]
 [0.37783824 0.49145396 0.53938313 0.42077713 0.77259757]]
```

4. Create an array with at least 1000 random numbers from normal distributions (normal). Then, plot a histogram of these values (`plt.hist`).

In [127…
```
array2 = np.random.randn(100,10)
plt.hist(array2)
```

Out[127]:
```
(array([[ 0.,  0.,  6., 25., 24., 24., 13.,  7.,  0.,  1.],
        [ 0.,  1., 10., 23., 31., 12., 18.,  4.,  1.,  0.],
        [ 0.,  4.,  8., 17., 20., 30., 15.,  4.,  1.,  1.],
        [ 0.,  2.,  9., 20., 22., 30., 13.,  3.,  1.,  0.],
        [ 1.,  1.,  8., 17., 26., 26., 15.,  5.,  1.,  0.],
        [ 0.,  2.,  6., 27., 24., 18., 18.,  5.,  0.,  0.],
        [ 2.,  2.,  6., 17., 29., 21., 15.,  7.,  0.,  1.],
        [ 1.,  5.,  6., 16., 22., 30., 15.,  2.,  3.,  0.],
        [ 0.,  2., 10., 11., 25., 27., 13., 11.,  0.,  1.],
        [ 0.,  0.,  8., 21., 26., 23., 15.,  5.,  2.,  0.]]),
 array([-3.28992419, -2.60193553, -1.91394687, -1.22595821, -0.53796955,
         0.1500191 ,  0.83800776,  1.52599642,  2.21398508,  2.90197374,
         3.5899624 ]),
 <a list of 10 BarContainer objects>)
```



Pandas

1. Read in a CSV () and display all the columns and their respective data types

In [128…
```
df = pd.read_csv('hw_data.csv')
df.dtypes
```

Out[128]:
```
id          int64
sex        object
weight      int64
height      int64
dtype: object
```

2. Find the average weight

In [130…
```
df.loc[:, 'weight'].mean()
```

Out[130]:
```
135.71428571428572
```

3. Find the Value Counts on column `sex`

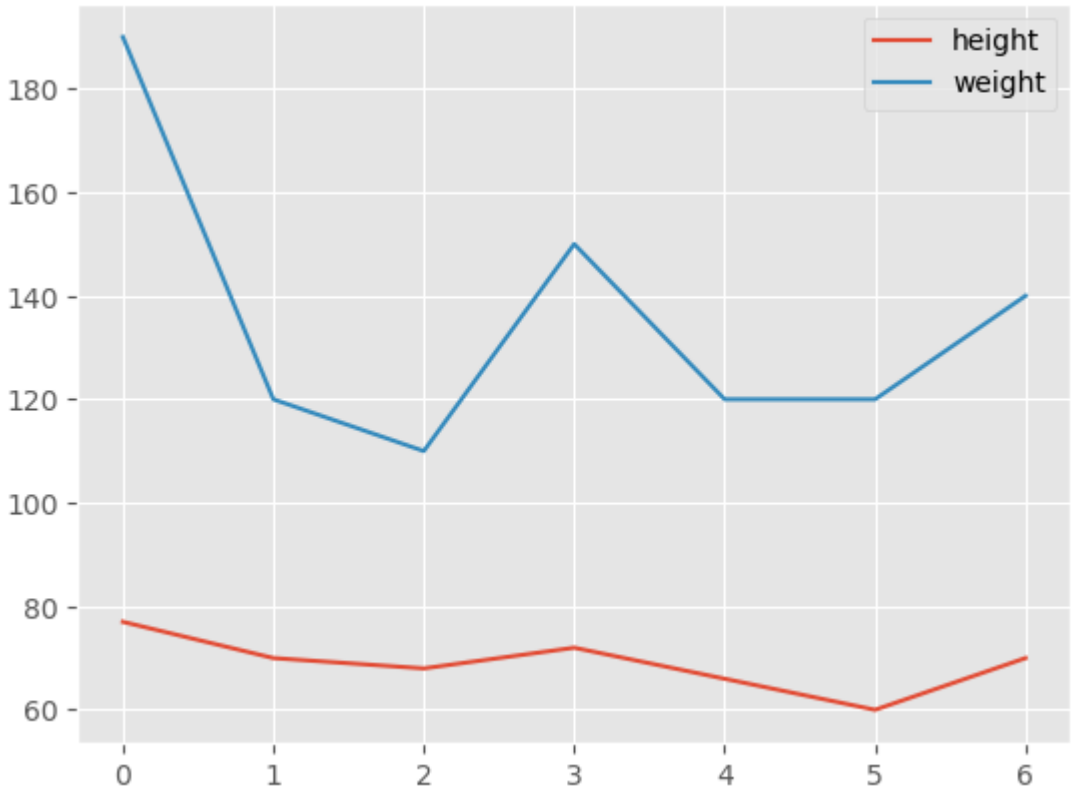
In [131…
```
df['sex'].value_counts()
```

Out[131]:
```
M    3
F    3
O    1
Name: sex, dtype: int64
```

4. Plot Height vs. Weight

In [132…
```
df[['height','weight']].plot()
```

Out[132]:
```
<Axes: >
```



5. Calculate BMI and save as a new column

In [133…
```
df['BMI'] = df['weight']/df['height']
print(df)
```

```
   id sex  weight  height       BMI
0   1   M     190      77  2.467532
1   2   F     120      70  1.714286
2   3   F     110      68  1.617647
3   4   M     150      72  2.083333
4   5   O     120      66  1.818182
5   6   M     120      60  2.000000
6   7   F     140      70  2.000000
```

6. Save sheet as a new CSV file `hw_dataB.csv`

In [115…
```
df.to_csv('hw_dataB.csv')
```

Run the following (Mac)

In [ ]:
```
!cat hw_dataB.csv
```

Run the following (Windows)

In [116…
```
!type hw_dataB.csv
```

```
,id,sex,weight,height,BMI
0,1,M,190,77,2.4675324675324677
1,2,F,120,70,1.7142857142857142
2,3,F,110,68,1.6176470588235294
3,4,M,150,72,2.083333333333335
4,5,O,120,66,1.8181818181818181
5,6,M,120,60,2.0
6,7,F,140,70,2.0
```

In [ ]:
```

```