In [1]:

```
import numpy as np
import pandas as pd
import scipy as sp
```

In [2]:

```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('ggplot')
```

In [3]:

```
%%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
```

```
Writing hw_data.csv
```

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 [23]:

```
def average(my_list):
    total = 0
    for item in my_list:
        total = total + item

    total = total / len(my_list)

    return total

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

Out[23]:

```
3.375
```

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

In [26]:

```
def counts(my_list):
    counts = dict()
    for item in my_list:
        counts[item] = counts.get(item, 0) + 1
    #based on stackexchange discussion, counts.get(item, 0) + 1 sets a
    #isn't in the dictionary/list of number so the first time it sees

    return counts

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

Out[26]:

```
{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 [30]:

```
paragraph_text = '''
For a minute or two she stood looking at the house, and wondering what
The Fish-Footman began by producing from under his arm a great letter,
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
Alice went timidly up to the door, and knocked.
‘There’s no sort of use in knocking,’ said the Footman, ‘and that for
‘Please, then,’ said Alice, ‘how am I to get in?’
‘There might be some sense in your knocking,’ the Footman went on with
‘I shall sit here,’ the Footman remarked, ‘till tomorrow—’
At this moment the door of the house opened, and a large plate came sk

input = str.split(paragraph_text) #split separates string into list de

from collections import Counter

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

    sorted_counts = dict(Counter(counts).most_common()) #counter libra

    return sorted_counts


counts(input)
```

```
 'because': 3,
 'by': 3,
 'large': 3,
 'that': 3,
 'all': 3,
 'their': 3,
 'this': 3,
 'for': 3,
 'into': 3,
 'up': 3,
 'said': 3,
 'I': 3,
 'might': 3,
 'For': 2,
 'or': 2,
 'two': 2,
 'looking': 2,
 'what': 2,
 'when': 2,
 'came': 2,
```

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 [62]:

```
import os
directory = os.listdir('/Users/spencerdouglas/Desktop/mlnm-personal')
path = directory[2]

with open(path, 'r') as input, open('Output.txt', 'w') as output:
    for line in input:
        newline = line.title()
        output.write(newline)

with open('Output.txt', 'r') as file:
    print(file.read())
```

```
This Is My First Line
This Is My Second Line
This Is My Third Line
This Is My Fourth Line
This Is My Fifth Line
```

Numpy

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

In [63]:

```
simple_list = [1,2,1,4,3,2,5,9]

np.mean(simple_list)
```

Out[63]:

```
3.375
```

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

In [69]:
```
heights = [174, 173, 173, 175, 171] #centimeters
weights = [88, 83, 92, 74, 77] #KG

height_array = np.array(heights)
weight_array = np.array(weights)

#BMI formula from https://www.cdc.gov/nccdphp/dnpao/growthcharts/train
# (kg / m**2) convert height to m by height/100

meters = height_array / 100

bmi = weight_array / (meters **2)

print(bmi)
```

```
[29.06592681 27.73229978 30.73941662 24.16326531 26.33288875]
```

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

In [70]:
```
array = np.random.rand(20)
array
```

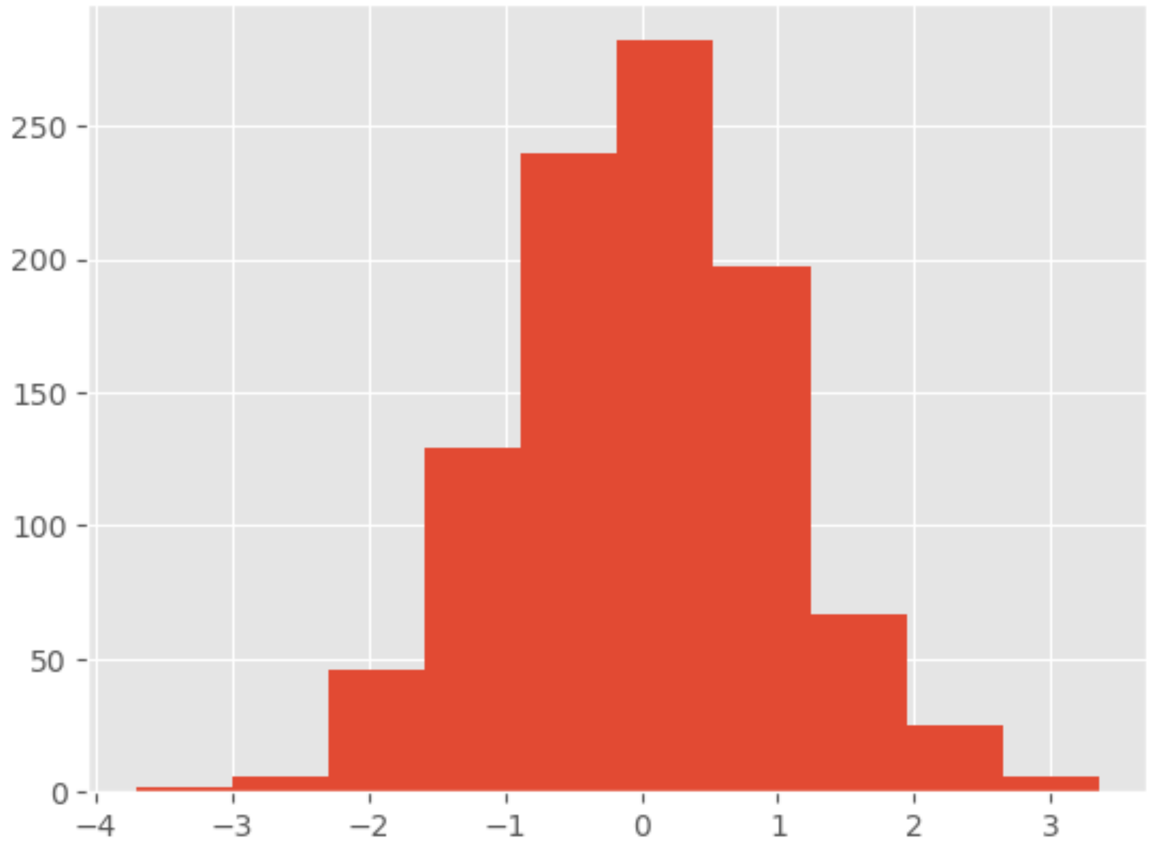
Out[70]:
```
array([0.62902004, 0.88484477, 0.27570957, 0.43154038, 0.58479144,
       0.48939863, 0.74224344, 0.68915272, 0.38106724, 0.57732845,
       0.03372957, 0.83528319, 0.87147308, 0.88579039, 0.18502511,
       0.73489683, 0.03996998, 0.59798767, 0.52759378, 0.17061822])
```

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

```
In [74]: array = np.random.normal(size = 1000)

         plt.hist(array)
```

```
Out[74]: (array([  2.,   6.,  46., 129., 240., 282., 197.,  67.,  25.,   6.]),
          array([-3.70187147, -2.99658452, -2.29129757, -1.58601062, -0.880723
         67,
                 -0.17543672,  0.52985023,  1.23513719,  1.94042414,  2.645711
         09,
                  3.35099804]),
          <BarContainer object of 10 artists>)
```



Pandas

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

In [78]:
```
import pandas as pd
data = pd.read_csv('hw_data.csv')
print(data)
data.dtypes
```

```
   id sex  weight  height
0   1   M     190      77
1   2   F     120      70
2   3   F     110      68
3   4   M     150      72
4   5   O     120      66
5   6   M     120      60
6   7   F     140      70
```

Out[78]:
```
id           int64
sex         object
weight       int64
height       int64
dtype: object
```

2. Find the average weight

In [81]:
```
data.mean()
```

```
/var/folders/g3/jswb050d2gbg2jfqy91z6nq40000gn/T/ipykernel_74342/5319
03386.py:1: FutureWarning: The default value of numeric_only in DataF
rame.mean is deprecated. In a future version, it will default to Fals
e. In addition, specifying 'numeric_only=None' is deprecated. Select
only valid columns or specify the value of numeric_only to silence th
is warning.
  data.mean()
```

Out[81]:
```
id          4.000000
weight    135.714286
height     69.000000
dtype: float64
```

3. Find the Value Counts on column `sex`

In [83]:
```
  data['sex'].value_counts()
```

Out[83]:
```
M    3
F    3
O    1
Name: sex, dtype: int64
```

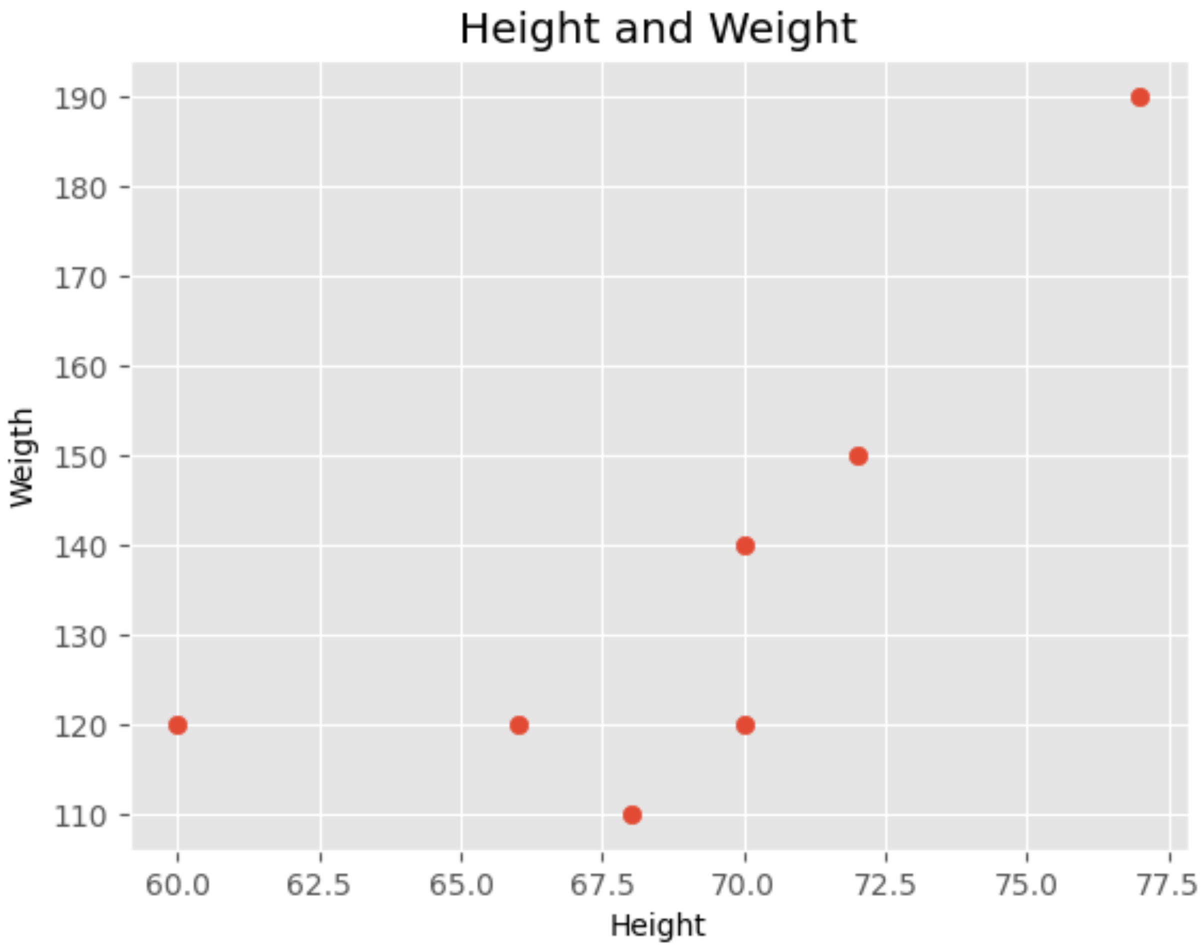
4. Plot Height vs. Weight

In [89]:

```
import matplotlib.pyplot as plt
plt.scatter(data['height'], data['weight'])
plt.title("Height and Weight")
plt.xlabel("Height")
plt.ylabel("Weigth")
```

Out[89]:

```
Text(0, 0.5, 'Weigth')
```



5. Calculate BMI and save as a new column

In [93]:

```
#weight and height in this data are assumed to be inches/pounds based
#formula: https://home.ubalt.edu/ntsbarsh/business-stat/otherapplets/B
# (height / weight squared) x 703 to convert to metric values

#Add BMI Column

data['BMI'] = (data['weight'] / (data['height']**2)) * 703

print(data)
```

```
   id sex  weight  height  Meters        BMI
0   1   M     190      77    0.77  22.528251
1   2   F     120      70    0.70  17.216327
2   3   F     110      68    0.68  16.723616
3   4   M     150      72    0.72  20.341435
4   5   O     120      66    0.66  19.366391
5   6   M     120      60    0.60  23.433333
6   7   F     140      70    0.70  20.085714
```

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

In [94]:

```
data.to_csv('hw_dataB.csv')
```

Run the following (Mac)

In [95]:

```
!cat hw_dataB.csv
```

```
,id,sex,weight,height,Meters,BMI
0,1,M,190,77,0.77,22.52825096980941
1,2,F,120,70,0.7,17.216326530612243
2,3,F,110,68,0.68,16.72361591695502
3,4,M,150,72,0.72,20.341435185185187
4,5,O,120,66,0.66,19.366391184573004
5,6,M,120,60,0.6,23.433333333333334
6,7,F,140,70,0.7,20.085714285714285
```

Run the following (Windows)

In [ ]:

```
!type hw_dataB.csv
```