In [37]:
```
import numpy as np
import pandas as pd
import scipy as sp
```

In [38]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('ggplot')
```

In [39]:
```
%%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
```

```
Overwriting 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 [40]:
```
def average(my_list):
    total = 0
    for item in my_list:
        #do something with item!
        total = total + item
    list_avg = total / len(my_list)
    return list_avg

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

Out[40]:
```
3.375
```

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

In [41]:
```
def counts(my_list):
    counts_dict = dict()
    for item in my_list:
        #do something with item!
        counts_dict[item] = counts_dict.get(item, 0) + 1
    return counts_dict

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

Out[41]:
```
{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 [99]:
```
paragraph_text = '''
For a minute or two she stood looking at the house, and wondering what to do next, when suddenly a footman in livery ca
The Fish-Footman began by producing from under his arm a great letter, nearly as large as himself, and this he handed o
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 their hearing her; and when she next
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. First, because I'm on the same side
'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 her, 'if we had the door betwee
'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, straight at the Footman's head: it ju

counts()
```

```
In [103]: punc = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''

          for ele in paragraph_text:
              if ele in punc:
                  paragraph_text = paragraph_text.replace(ele, "")

          split_text = paragraph_text.split(" ")


          counts(split_text)
```

```
Out[103]: {'\nFor': 1,
           'a': 15,
           'minute': 1,
           'or': 2,
           'two': 2,
           'she': 6,
           'stood': 1,
           'looking': 2,
           'at': 6,
           'the': 32,
           'house': 2,
           'and': 16,
           'wondering': 1,
           'what': 2,
           'to': 15,
           'do': 1,
           'next': 2,
           'when': 2,
           'suddenly': 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 [69]: file = open("zen_of_python.txt", 'r')
         #print(file.read())

         # Using for loop
         file_2 = open("titled_text.txt", 'w')
         for line in file:
             file_2.write(line.title())

         file_2.close()
```

Numpy

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

```
In [53]: simple_list = [1,2,1,4,3,2,5,9]

         np.average(simple_list)

Out[53]: 3.375
```

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

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

```
In [92]: np_height_m = np.array(heights) / 100
         np_weight_kg = np.array(weights)
         bmi = np_weight_kg / np_height_m ** 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 [77]: np.linspace(0, 1, 20)
```

```
Out[77]: array([0.        , 0.05263158, 0.10526316, 0.15789474, 0.21052632,
                0.26315789, 0.31578947, 0.36842105, 0.42105263, 0.47368421,
                0.52631579, 0.57894737, 0.63157895, 0.68421053, 0.73684211,
                0.78947368, 0.84210526, 0.89473684, 0.94736842, 1.        ])
```

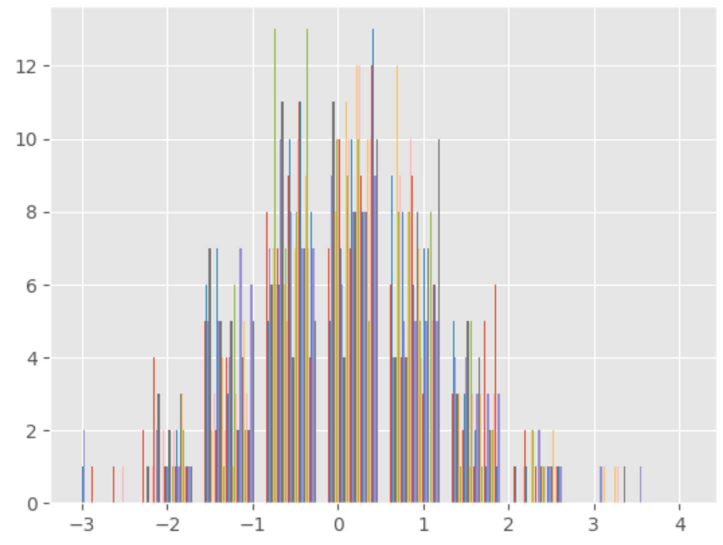
4. Create an array with at least 1000 random numbers from normal distributions (normal). Then, plot a histogram of these values (`plt.hist`).

```
In [68]: norm_array = np.random.randn(32, 32)
         norm_array
```

```
Out[68]: array([[-1.36040446,  1.86376877,  1.64465316, ...,  0.67101757,
                 -1.37056536,  1.15649238],
                [ 1.18159173,  0.55019201, -0.88437891, ...,  0.26587681,
                 -1.98405254,  0.23355589],
                [ 0.19754413,  1.38671359, -1.1811326 , ..., -0.38498621,
                 -1.07386798, -1.03619566],
                ...,
                [ 2.05111649, -2.62738531, -1.62673723, ...,  0.22563039,
                  0.08906509, -0.14519216],
                [-1.33229387, -0.23755742, -0.26656738, ...,  0.44926177,
                  0.24321748, -0.01335638],
                [-0.52726786,  1.75792539,  1.56085627, ..., -0.29900856,
                 -0.10242594,  0.68509459]])
```

```
In [69]: plt.hist(norm_array)
```

```
Out[69]: (array([[ 0.,  2.,  5.,  8.,  7.,  6.,  3.,  1.,  0.,  0.],
                [ 1.,  0.,  6.,  5.,  5.,  9.,  5.,  1.,  0.,  0.],
                [ 2.,  0.,  5.,  7.,  9.,  4.,  4.,  0.,  0.,  1.],
                [ 0.,  1.,  7.,  6., 11.,  4.,  3.,  0.,  0.,  0.],
                [ 0.,  0.,  2.,  7.,  8., 12.,  3.,  0.,  0.,  0.],
                [ 0.,  0.,  0., 13., 10.,  8.,  1.,  0.,  0.,  0.],
                [ 0.,  1.,  3.,  6., 10.,  9.,  2.,  1.,  0.,  0.],
                [ 1.,  4.,  2.,  7., 10.,  4.,  2.,  2.,  0.,  0.],
                [ 0.,  0.,  7.,  6.,  7.,  8.,  3.,  1.,  0.,  0.],
                [ 0.,  2.,  5., 10.,  6.,  5.,  4.,  0.,  0.,  0.],
                [ 0.,  3.,  5., 11.,  4.,  4.,  5.,  0.,  0.,  0.],
                [ 0.,  0.,  4.,  6., 11.,  8.,  1.,  2.,  0.,  0.],
                [ 0.,  0.,  1.,  7.,  9.,  8.,  5.,  2.,  0.,  0.],
                [ 0.,  2.,  2.,  5., 10., 10.,  3.,  0.,  0.,  0.],
                [ 0.,  1.,  4.,  9.,  7.,  9.,  1.,  1.,  0.,  0.],
                [ 0.,  1.,  3., 10., 10.,  6.,  2.,  0.,  0.,  0.],
                [ 0.,  1.,  4.,  8.,  8.,  5.,  3.,  2.,  1.,  0.],
                [ 0.,  2.,  5.,  4.,  8.,  8.,  4.,  1.,  0.,  0.],
                [ 0.,  0.,  1.,  7., 12.,  7.,  3.,  1.,  1.,  0.],
                [ 0.,  1.,  6.,  8., 10.,  5.,  1.,  1.,  0.,  0.],
                [ 0.,  2.,  3., 10., 12.,  4.,  0.,  1.,  0.,  0.],
                [ 1.,  1.,  2., 11.,  9.,  3.,  5.,  0.,  0.,  0.],
                [ 0.,  2.,  2., 11.,  8.,  7.,  1.,  1.,  0.,  0.],
                [ 0.,  1.,  7.,  7.,  8.,  5.,  3.,  1.,  0.,  0.],
                [ 0.,  3.,  4.,  7.,  8.,  7.,  2.,  1.,  0.,  0.],
                [ 0.,  3.,  5.,  9., 10.,  0.,  2.,  2.,  1.,  0.],
                [ 0.,  2.,  2., 13.,  5.,  8.,  2.,  0.,  0.,  0.],
                [ 1.,  1.,  3.,  7., 10.,  5.,  3.,  1.,  1.,  0.],
                [ 0.,  1.,  2.,  4., 12.,  6.,  6.,  1.,  0.,  0.],
                [ 0.,  1.,  2.,  8., 13.,  6.,  1.,  1.,  0.,  0.],
                [ 0.,  1.,  6.,  7.,  9.,  5.,  3.,  1.,  0.,  0.],
                [ 0.,  1.,  5.,  5., 10., 10.,  0.,  0.,  1.,  0.]]),
          array([-3.08788286, -2.36310539, -1.63832792, -0.91355045, -0.18877297,
                  0.5360045 ,  1.26078197,  1.98555944,  2.71033691,  3.43511439,
                  4.15989186]),
          <a list of 32 BarContainer objects>)
```



Pandas

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

```
In [27]: df = pd.read_csv("hw_data.csv")
         print(df)
         df.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[27]: id          int64
         sex        object
         weight      int64
         height      int64
         dtype: object
```

2. Find the average weight

```
In [57]: avg_weight = df["weight"].mean()
         print (avg_weight)
```

```
135.71428571428572
```

3. Find the Value Counts on column `sex`

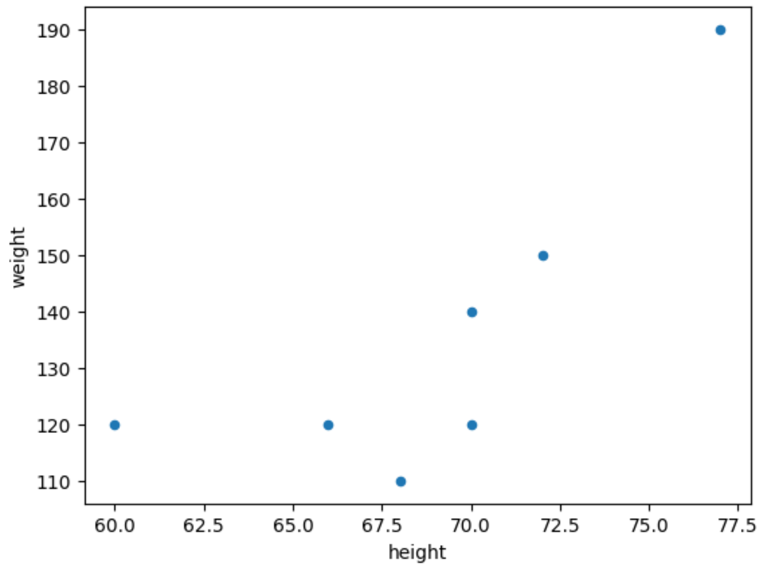
```
In [61]: sex_count = counts(df["sex"])
         print(sex_count)

         sex_count_2 = df["sex"].value_counts()
         print(sex_count_2)
```

```
{'M': 3, 'F': 3, 'O': 1}
M    3
F    3
O    1
Name: sex, dtype: int64
```

4. Plot Height vs. Weight

```
In [65]: df.plot.scatter("height", "weight")
```

```
Out[65]: <AxesSubplot: xlabel='height', ylabel='weight'>
```



5. Calculate BMI and save as a new column

```
In [84]: df ["BMI"] = (df['weight'] / df['height']**2) * 703
         print(df)
```

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

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

```
In [93]: df.to_csv("hw_dataB.csv")
```

Run the following (Mac)

```
In [43]: !cat hw_dataB.csv
```

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

Run the following (Windows)

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