In [156]:
```
import numpy as np
import pandas as pd
import scipy as sp
```

In [157]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('ggplot')
```

In [158]:
```
%%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 [159]:
```
#I will create a function that takes the provided list and returns the average
def average(my_list):
    total = 0
    for item in my_list:
        total += item

    return total / len(my_list)

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

Out[159]:
```
3.375
```

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

In [160]:
```
#I will create a function that will count the numbers/items from a list (using
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[160]:
```
{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 [161]:
```
import string

#I will create counts() function that will analyze the instances of each item
def counts(my_list):
    counts = dict()
    for item in my_list:
        counts[item] = counts.get(item, 0) + 1
    return counts

#As requsted for the bonus segement of this question, I will remove punctuatio
def no_punctuation(text):
    translator = str.maketrans('', '', string.punctuation)
    return text.translate(translator)

paragraph_text = '''
For a minute or two she stood looking at the house, and wondering what to do n
The Fish-Footman began by producing from under his arm a great letter, nearly
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
Alice went timidly up to the door, and knocked.
‘There’s no sort of use in knocking,’ said the Footman, ‘and that for two reas
‘Please, then,’ said Alice, ‘how am I to get in?’
‘There might be some sense in your knocking,’ the Footman went on without atte
‘I shall sit here,’ the Footman remarked, ‘till tomorrow—’
At this moment the door of the house opened, and a large plate came skimming o

#Post-processing of the text
normalized_text = no_punctuation(paragraph_text.lower()) #normalization to ens
words = normalized_text.split() #returns a list of words from the text
word_counts = counts(words) #utilizes the counts function that was created int

#Print the results
for word, count in word_counts.items():
    print(f"'{word}': {count}")
```

```
'for': 5
'a': 15
'minute': 1
'or': 2
'two': 2
'she': 7
'stood': 1
'looking': 2
'at': 7
'the': 34
'house': 2
'and': 17
'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 [162]:
```
%%file jhmikesample.txt
my name is michael reiderman
i like to read and cook
i also like listening to music
i wish i had a dog
```

```
Overwriting jhmikesample.txt
```

In [163]:
```
#Define the file path - input file that contains the original text is titled j
reading_file = "jhmikesample.txt"

#Define the file path - output file that contains the transformed text is titl
writing_file = "jhmikenewsample.txt"

#Open both files at the same time. "reading_file" is opened in reading mode wh
with open(reading_file, "r") as input_file, open(writing_file, "w") as output_
    for line in input_file: #Go through each line in the reading file
        transformed_line = line.title() #Utilize the title() function in order
        output_file.write(transformed_line) #Transformed letters (lower case t
        print(transformed_line, end='') #View the transformed letters
```

```
My Name Is Michael Reiderman
I Like To Read And Cook
I Also Like Listening To Music
I Wish I Had A Dog
```

Numpy

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

In [164]:
```
#Given list
simple_list = [1,2,1,4,3,2,5,9]

#Calculate the mean of g
average_value = np.mean(simple_list)

#Print the average value of the given list
average_value
```

Out[164]:
```
3.375
```

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

In [165]:
```
#Original list given (assumption here is that heights is in centimeters and we
#as BMI = Weight (in kilograms) / Height ^ 2 (in meters)
heights = [174, 173, 173, 175, 171]
weights = [88, 83, 92, 74, 77]

#Create numpy arrays for heights and weights
heights = np.array([174, 173, 173, 175, 171])
weights = np.array([88, 83, 92, 74, 77])

#Before we can calculate the BMI, we miust first obtain the variables that are
#then square the heights that we just finished coverting
cm_meters = heights / 100
meters_squared = cm_meters ** 2

#We will now calculate the BMI by dividing their weight by their respective sq
bmi = weights / meters_squared

#Print the BMI array
bmi
```

Out[165]:
```
array([29.06592681, 27.73229978, 30.73941662, 24.16326531, 26.33288875])
```

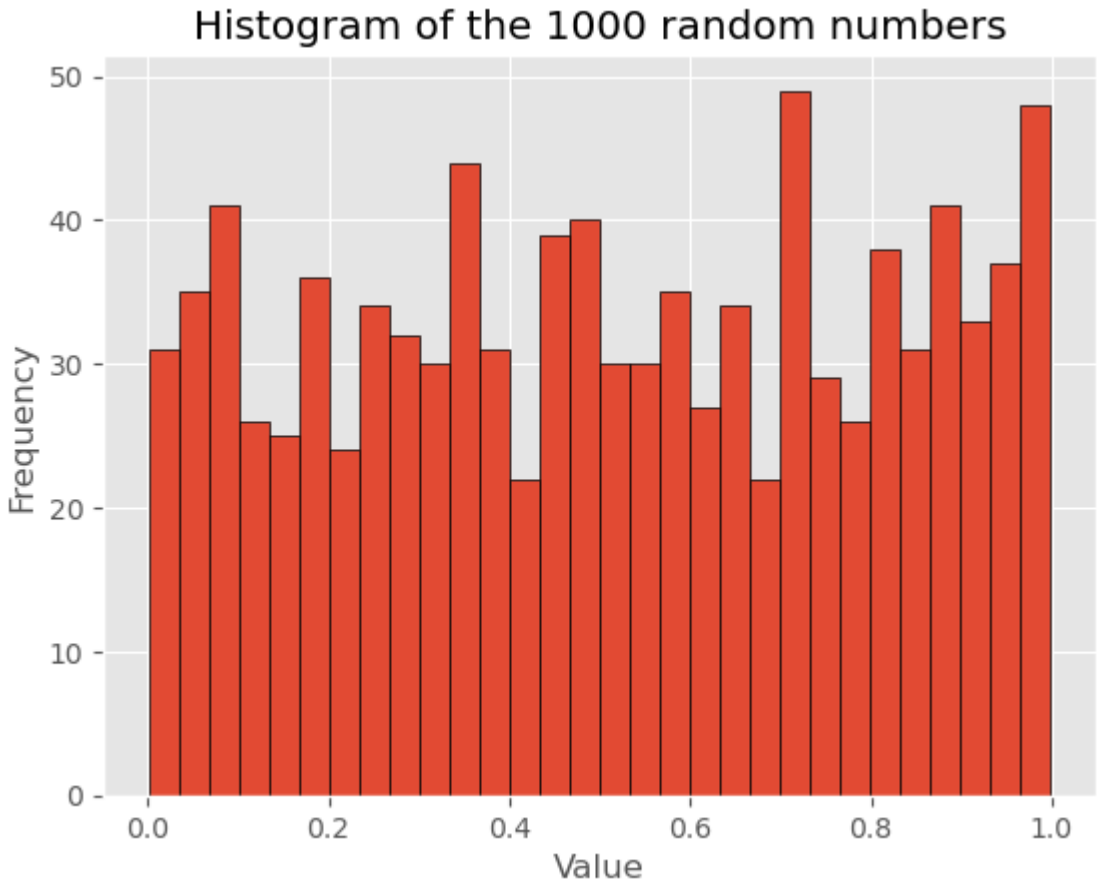
3. Create an array of length 20 filled with random values (between 0 to 1)

In [166]:
```
#Generate a random array that contains 20 random values
random_array = np.random.rand(20)

#Print the random array that contains 20 random values
random_array
```

Out[166]:
```
array([0.78972191, 0.75083956, 0.94949538, 0.95477119, 0.26296688,
       0.49053968, 0.13422387, 0.60381616, 0.79197629, 0.85374614,
       0.87880855, 0.47383821, 0.28716788, 0.00647413, 0.10610397,
       0.15285787, 0.50764088, 0.43006694, 0.3670808 , 0.47752507])
```

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

In [167]:
```
#Generate a random array that contains 1000 random values
random_array_2 = np.random.rand(1000)

#Plot the random array
plt.hist(random_array_2, bins=30, edgecolor='black')
plt.title('Histogram of the 1000 random numbers')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.show()
```



Pandas

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

In [168]:
```
#Create the variable "hw_data_df" that reads the hw_data.csv
hw_data_df = 'hw_data.csv'

#Read the variable "hw_data_df" and convert it to a pandas dataframe
hw_data_df_2 = pd.read_csv(hw_data_df)

#Print the data contained in the hw_data_df_2 pandas dataframe
print(hw_data_df_2.dtypes)
```

```
id         int64
sex       object
weight     int64
height     int64
dtype: object
```

2. Find the average weight

In [169]:
```
#Select the weight column in the "hw_data_df_2" dataframe and calculate the me
average_weight = hw_data_df_2['weight'].mean()

#Print the average weight
print("The average weight if weight column is equal to (in pounds i am assumin
```

```
The average weight if weight column is equal to (in pounds i am assuming): 13
5.71428571428572
```

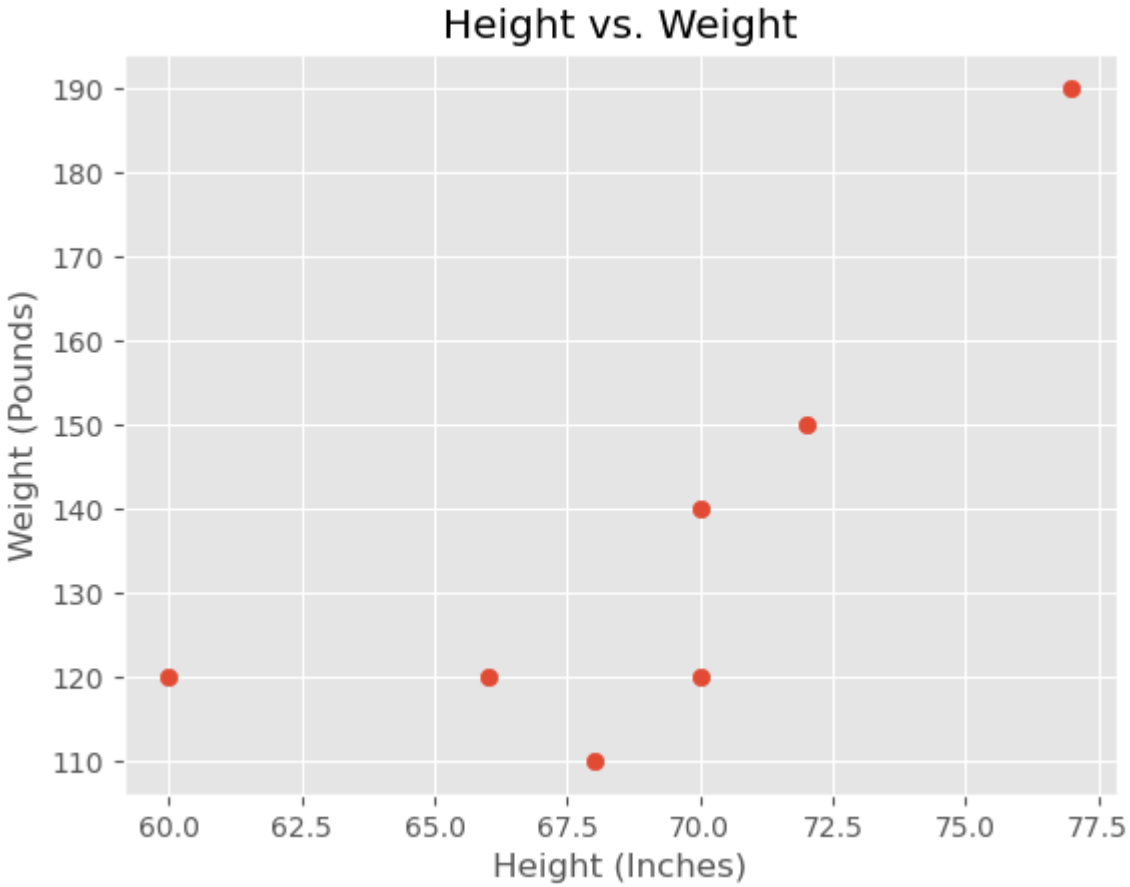
3. Find the Value Counts on column sex

In [170]:
```
#This selects the "sex" column located in the "hw_data_df_2" dataframe and com
#Following the computation, assign the values to the "sex_value_counts" variab
sex_value_counts = hw_data_df_2['sex'].value_counts()

#Print the unique value counts for the sex column
print(sex_value_counts)
```

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

4. Plot Height vs. Weight

In [171]:
```
plt.scatter(hw_data_df_2['height'], hw_data_df_2['weight'])
plt.title('Height vs. Weight')
plt.xlabel('Height (Inches)')
plt.ylabel('Weight (Pounds)')
plt.show()
```



5. Calculate BMI and save as a new column

In [172]:
```
#First and foremost, we must convert weight (pounds) and height (inches) to we
#order to calculate the BMI values

#For height, convert inches to meters
hw_data_df_2['mHeight'] = hw_data_df_2['height'] * .0254

#For weight, convert pounds to kilograms
hw_data_df_2['kgWeight'] = hw_data_df_2['weight'] / 2.20462

#Now that we've obtained the proper variables, calculate the BMI values
hw_data_df_2['BMI'] = hw_data_df_2['kgWeight'] / (hw_data_df_2['mHeight'] ** 2

#View the BMI values that were just calculated
print(hw_data_df_2.head())
```

```
   id sex  weight  height  mHeight   kgWeight        BMI
0   1   M     190      77   1.9558  86.182653  22.530508
1   2   F     120      70   1.7780  54.431149  17.218051
2   3   F     110      68   1.7272  49.895220  16.725291
3   4   M     150      72   1.8288  68.038936  20.343473
4   5   O     120      66   1.6764  54.431149  19.368331
```

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

In [173]:
```
hw_data_df_2.to_csv('hw_dataB.csv', index=False)
```

Run the following (Mac)

In [119]:
```
#Not applicable since I am on a Windows computer
!cat hw_dataB.csv
```

```
'cat' is not recognized as an internal or external command,
operable program or batch file.
```

Run the following (Windows)

In [174]:
```
!type hw_dataB.csv

#The following command returns a rows of observations along with variable colu

#Column 1 represents the ID (observation)
#Column 2 represents the sex (male or female, with one instance of O)
#Column 3 represents teh weight (in pounds i am assuming)
#Column 4 represents the height (in inches i am assuming)
#Column 5 represents the height (in meters)
#Column 6 represents the weight (in kilograms)
```

```
id,sex,weight,height,mHeight,kgWeight,BMI
1,M,190,77,1.9558,86.18265279277155,22.53050750473164
2,F,120,70,1.778,54.43114913227677,17.218050998352812
3,F,110,68,1.7271999999999998,49.895220037920375,16.72529103249742
4,M,150,72,1.8288,68.03893641534596,20.343472678416468
5,O,120,66,1.6764,54.43114913227677,19.368331012839484
6,M,120,60,1.524,54.43114913227677,23.43568052553577
7,F,140,70,1.778,63.50300732098956,20.087726164744943
```