Week 4

This week's assignments:

1. Go to the link below and watch sections 5.2, 5.5, 6.1, 6.2, and 9.3.
   `https://learning.oreilly.com/videos/python-for-data/9780135687253/9780135687253-pfds_01_05_02_00`

5.4 Pandas

Suppose you have three arrays: one that contains a list of country names, one that contains the correlating populations, and one that contains the correlating GDP's. You want to be able to look at the arrays and compare the countries to each other, but doing so is difficult with the array format. Pandas is a package that uses dictionaries and DataFrames to organize your data into a readable format. You are also able to retrieve data easily from DataFrames.

5.4.1 Dictionaries

Dictionaries is one method to organize your data. They are groups of pairs, or tuples, organized together. Tuples are like a list, except once they are created, the content cannot be changed. In this case, you will create tuples that consist of a title and a list. First type the title you want to give your array in quotes, then a colon, followed by the corresponding array. Do this again with the next array, separated by a comma. Wrap the whole phrase in curly brackets, and you are done.

```
names = ['United States','Canada','France',
         'United Kingdom','Italy','Germany']
GDP = [51958, 43376, 37360, 38083, 34876, 45320]
Pop = [327, 37, 67, 66, 60, 83]

my_dictionary = { #name of dictionary and open braces
    'Country': names, #tab before typing
    'GDP': GDP,       #don't put array in quotes
    'Population': Pop,
    }
```

A dictionary is like a table in an Excel file. Each title is like a column name, and each list is like the values in that column. However, if you were to print the dictionary you just created, it wouldn't look very good.

Try this: Print the dictionary that was created above. What does it look like?

Dictionaries are just a way to organize your data. They don't give your data any shape. DataFrames are the next step in creating an organized data set. They will organize your data into rows and columns so that you can read them easily.

5.4.2 DataFrames

DataFrames will give your data organizational structure. To do this, wrap your newly created dictionary in the DataFrame command.

```
my_df = pd.DataFrame(gdp_dictionary)
#Capitalize Data and Frame
print(my_df)
```

You can also combine the creation of a dictionary and a DataFrame into one command.

```
my_df = pd.DataFrame({ #name of dictionary and open braces
    'Country': names, #tab before typing
    'GDP per Capita': GDP,        #don't put array in quotes
    'Population (in Millions)': Pop,
    } )
print(my_df)
```

Both codes will get you an output like this:

	Country	GDP per Capita	Population (in Millions)
0	United States	51958	327
1	Canada	43376	37
2	France	37360	67
3	United Kingdom	38083	66
4	Italy	34876	60
5	Germany	45320	83

You can also perform operations on a DataFrame. If you name the operation, then it will turn into a new DataFrame. Suppose you want to multiply GDP per capita by population to get total GDP. See the code below for how to do this.

```
GDPtot = my_df['GDP per Capita'] * my_df['Population']
print(GDPtot)
```

```
0    16990266
1     1604912
2     2503120
3     2513478
4     2092560
5     3761560
```

You can add your new GDPtot to your DataFrame by using the code below.

```
my_df['GDP Total'] = GDPtot
print(my_df)
```

```
          Country  GDP per Capita  Population (in Millions)  GDP (in Millions)
0   United States           51958                       327           16990266
1          Canada           43376                        37            1604912
2          France           37360                        67            2503120
3  United Kingdom           38083                        66            2513478
4           Italy           34876                        60            2092560
5         Germany           45320                        83            3761560
```

You may be asked to add data to an empty section of your DataFrame. For example, instead of being asked to add a new column that contains the GDP total, you may see something like this:

```
names = ['United States','Canada','France',
         'United Kingdom','Italy','Germany']
GDP = [51958, 43376, 37360, 38083, 34876, 45320]
Pop = [327, 37, 67, 66, 60, 83]
GDPtot = ["","","","","",""]

my_df = pd.DataFrame({
    'Country': names,
    'GDP per Capita': GDP,
    'Population (in Millions)': Pop,
    'GDP (in Millions)': GDPtot
    } )
print(my_df)
```

	Country	GDP per Capita	Population (in Millions)	GDP (in Millions)
0	United States	51958	327	
1	Canada	43376	37	
2	France	37360	67	
3	United Kingdom	38083	66	
4	Italy	34876	60	
5	Germany	45320	83	

Filling in the DataFrame is very similar to creating a new column in a DataFrame. Follow the code below to do so.

```
GDPtot = my_df['GDP per Capita'] * my_df['Population (
    in Millions)']
my_df['GDP (in Millions)'] = GDPtot
print(my_df)
```

	Country	GDP per Capita	Population (in Millions)	GDP (in Millions)
0	United States	51958	327	16990266
1	Canada	43376	37	1604912
2	France	37360	67	2503120
3	United Kingdom	38083	66	2513478
4	Italy	34876	60	2092560
5	Germany	45320	83	3761560