In [ ]:

```
#Simple example of processing data with Python.
#Using pandas to create a dataframe, read from csv and json
#clean, analyze and use dataset to select specific columns or select row by value
```

In [46]:

```
#Creating a Pandas DataFrame

import pandas as pd
#creating canned data
data = {'Week':pd.Series(['Sunday','Monday','Tuesday','Wednesday','Thursday','Friday','Saturday'])
        ,'Snowfall':pd.Series(['3.5','0.1','1.00','0','4.6','1.0','0.2'])}

#Reading from a dataframe
dfcanned = pd.DataFrame(data)
print("Amount of Snowfall (in) each day of the week: \n",dfcanned)
```

```
Amount of Snowfall (in) each day of the week: 
         Week Snowfall
0     Sunday      3.5
1     Monday      0.1
2    Tuesday     1.00
3  Wednesday        0
4   Thursday      4.6
5     Friday      1.0
6   Saturday      0.2
```

In [12]:

```
#Reading from a csv

dfc = pd.read_csv(r'C:\Users\prati\Desktop\data.csv')
print("Reading from a csv File, The Monthly Rainfall and Temperature data:\n",dfc)
```

```
Reading from a csv File, The Monthly Rainfall and Temperature data:
         Month  Rainfall  Temperature
0     January     1.650         20.0
1    February     1.250         32.0
2       March     1.940         50.0
3       April     2.750         64.0
4         May     2.750         74.0
5        June     3.645         80.0
6        July     5.500         88.0
7      August     1.000         70.0
8   September     1.300         60.0
9     October       NaN          NaN
10   November     0.500         40.0
11   December     2.300         28.0
```

In [24]:

```
#Reading from a json file

df = pd.read_json(r'C:\Users\prati\data.json')
print("Reading from a json file:\n",df)
```

```
Reading from a json file:
         Month  Rainfall  Temperature
0     January     1.650         20.0
1    February     1.250         32.0
2       March     1.940         50.0
3       April     2.750         64.0
4         May     2.750         74.0
5        June     3.645         80.0
6        July     5.500         88.0
7      August     1.000         70.0
8   September     1.300         60.0
9     October       NaN          NaN
10   November     0.500         40.0
11   December     2.300         28.0
```

In [ ]:

```
#Next Cleaning the data:
```

In [29]:

```
#Filling '0' in the missing values

dfzeros = df.fillna(0)
print("The data with zeroed values: \n")
print(dfzeros)
```

```
The data with zeroed values: 

        Month  Rainfall  Temperature
0     January     1.650         20.0
1    February     1.250         32.0
2       March     1.940         50.0
3       April     2.750         64.0
4         May     2.750         74.0
5        June     3.645         80.0
6        July     5.500         88.0
7      August     1.000         70.0
8   September     1.300         60.0
9     October     0.000          0.0
10   November     0.500         40.0
11   December     2.300         28.0
```

In [28]:

```
#Removing rows that have invalid data

dfclean = df.dropna()
print("The data with dropped values: \n")
print(dfclean)
```

```
The data with dropped values: 

        Month  Rainfall  Temperature
0     January     1.650         20.0
1    February     1.250         32.0
2       March     1.940         50.0
3       April     2.750         64.0
4         May     2.750         74.0
5        June     3.645         80.0
6        July     5.500         88.0
7      August     1.000         70.0
8   September     1.300         60.0
10   November     0.500         40.0
11   December     2.300         28.0
```

In [31]:

```
#Counting number of rows with NaNs

count = 0
for index, row in df.iterrows():
    if any(row.isnull()):
        count = count + 1

print("Total Number of rows with Nans: "+str(count))
```

```
Total Number of rows with Nans: 1
```

In [34]:

```
#Basic Data Analysis

print("Mean: ",dfclean.mean())
print("\nMedian: ",dfclean.median())
print("\nStandard Deviation: ",dfclean.std())
```

```
Mean:  Rainfall        2.235000
Temperature    55.090909
dtype: float64

Median:  Rainfall        1.94
Temperature    60.00
dtype: float64

Standard Deviation:  Rainfall        1.413936
Temperature    22.669162
dtype: float64
```

In [ ]:

```
#Data Subset
```

In [41]:

```
#Indexing to print the rainfall and mean for first three months

rainfall = dfclean['Rainfall'][0:3]
print("Rainfall\n",rainfall)
print("Mean Rainfall for first 3 months is: ",rainfall.mean())
```

```
Rainfall
 0    1.65
1    1.25
2    1.94
Name: Rainfall, dtype: float64
Mean Rainfall for first 3 months is:  1.6133333333333333
```

In [42]:

```
#Using Indexing to select multiple columns from the dataset
#Printing just temperature and rainfall

dftr = (dfclean[['Temperature','Rainfall']])
print(dftr)
```

```
    Temperature  Rainfall
0          20.0     1.650
1          32.0     1.250
2          50.0     1.940
3          64.0     2.750
4          74.0     2.750
5          80.0     3.645
6          88.0     5.500
7          70.0     1.000
8          60.0     1.300
10         40.0     0.500
11         28.0     2.300
```

In [45]:

```
#loc function to selecting a specific row using a certain value

#Need to create a index as to use a loc function, we need to have a properly indexed framework
index = dfclean['Month']
dfIndexed = dfclean.set_index(index)
print("Selects a row by value \n", dfIndexed.loc['March'])
```

```
Selects a row by value 
 Month          March
Rainfall        1.94
Temperature       50
Name: March, dtype: object
```

In [ ]: