Assignment 2

Before working on this assignment please read these instructions fully. In the submission area, you will notice that you can click the link to **Preview the Grading** for each step of the assignment. This is the criteria that will be used for peer grading. Please familiarize yourself with the criteria before beginning the assignment.

An NOAA dataset has been stored in the file `data/C2A2_data/BinnedCsvs_d400/fb441e62df2d58994928907a91895ec62c2c42e6cd075c2700843b89.c` The data for this assignment comes from a subset of The National Centers for Environmental Information (NCEI) Daily Global Historical Climatology Network (https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt) (GHCN-Daily). The GHCN-Daily is comprised of daily climate records from thousands of land surface stations across the globe.

Each row in the assignment datafile corresponds to a single observation.

The following variables are provided to you:

- **id** : station identification code
- **date** : date in YYYY-MM-DD format (e.g. 2012-01-24 = January 24, 2012)
- **element** : indicator of element type
    - TMAX : Maximum temperature (tenths of degrees C)
    - TMIN : Minimum temperature (tenths of degrees C)
- **value** : data value for element (tenths of degrees C)

For this assignment, you must:

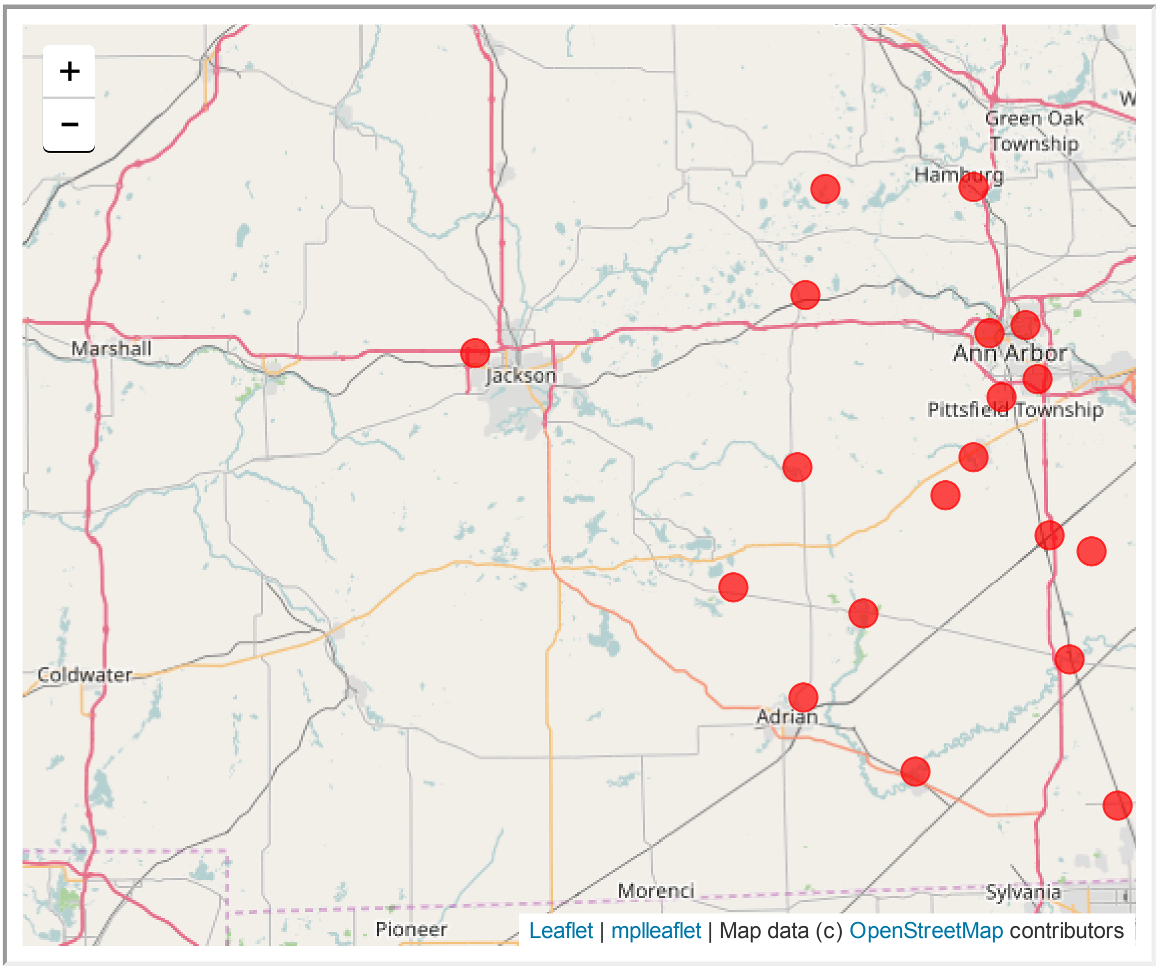
1. Read the documentation and familiarize yourself with the dataset, then write some python code which returns a line graph of the record high and record low temperatures by day of the year over the period 2005-2014. The area between the record high and record low temperatures for each day should be shaded.
2. Overlay a scatter of the 2015 data for any points (highs and lows) for which the ten year record (2005-2014) record high or record low was broken in 2015.
3. Watch out for leap days (i.e. February 29th), it is reasonable to remove these points from the dataset for the purpose of this visualization.
4. Make the visual nice! Leverage principles from the first module in this course when developing your solution. Consider issues such as legends, labels, and chart junk.

The data you have been given is near **Ann Arbor, Michigan, United States**, and the stations the data comes from are shown on the map below.

In [1]:

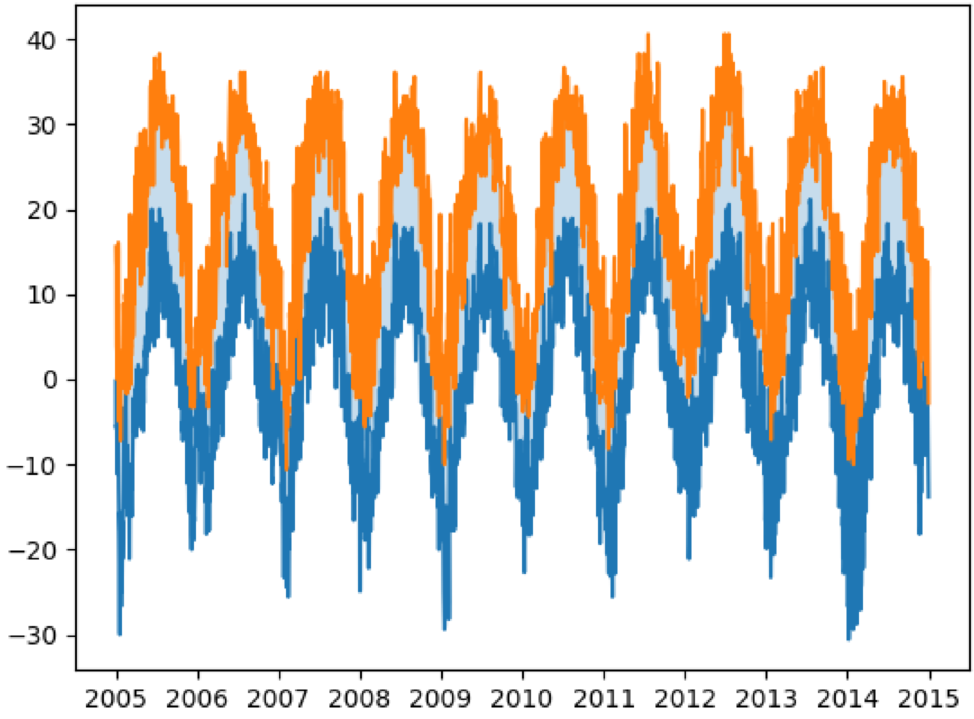
```
import matplotlib.pyplot as plt
import mplleaflet
import pandas as pd

def leaflet_plot_stations(binsize, hashid):

    df = pd.read_csv('data/C2A2_data/BinSize_d{}.csv'.format(binsize))
    station_locations_by_hash = df[df['hash'] == hashid]

    lons = station_locations_by_hash['LONGITUDE'].tolist()
    lats = station_locations_by_hash['LATITUDE'].tolist()

    plt.figure(figsize=(8,8))

    plt.scatter(lons, lats, c='r', alpha=0.7, s=200)

    return mplleaflet.display()

leaflet_plot_stations(400,'fb441e62df2d58994928907a91895ec62c2c42e6cd075c2700843b89')
```

Out[1]:



In [2]:

```
%matplotlib notebook
import pandas as pd
import numpy as np
#fig, ax = plt.subplots()
#fig.canvas.draw()

df = pd.read_csv('data/C2A2_data/BinnedCsvs_d400/fb441e62df2d58994928907a91895ec62c2c42e6cd075c2700843b89.csv')
MIN = df[df['Element'] == 'TMIN'].groupby('Date').agg({'Data_Value': np.min}).reset_index()
MIN = MIN[pd.to_datetime(MIN['Date']) < pd.Timestamp('2015-01-01')]
MAX = df[df['Element'] == 'TMAX'].groupby('Date').agg({'Data_Value': np.max}).reset_index()
MAX = MAX[pd.to_datetime(MAX['Date']) < pd.Timestamp('2015-01-01')]
Date_array = np.array(MIN['Date'].tolist())
Date_array = list(map(pd.to_datetime, Date_array))
Min_array = np.array(MIN['Data_Value'].tolist())/10
Max_array = np.array(MAX['Data_Value'].tolist())/10
plt.plot(Date_array,Min_array,'-',Date_array,Max_array,'-')
plt.gca().fill_between(Date_array,Min_array, Max_array,alpha=0.25)
```

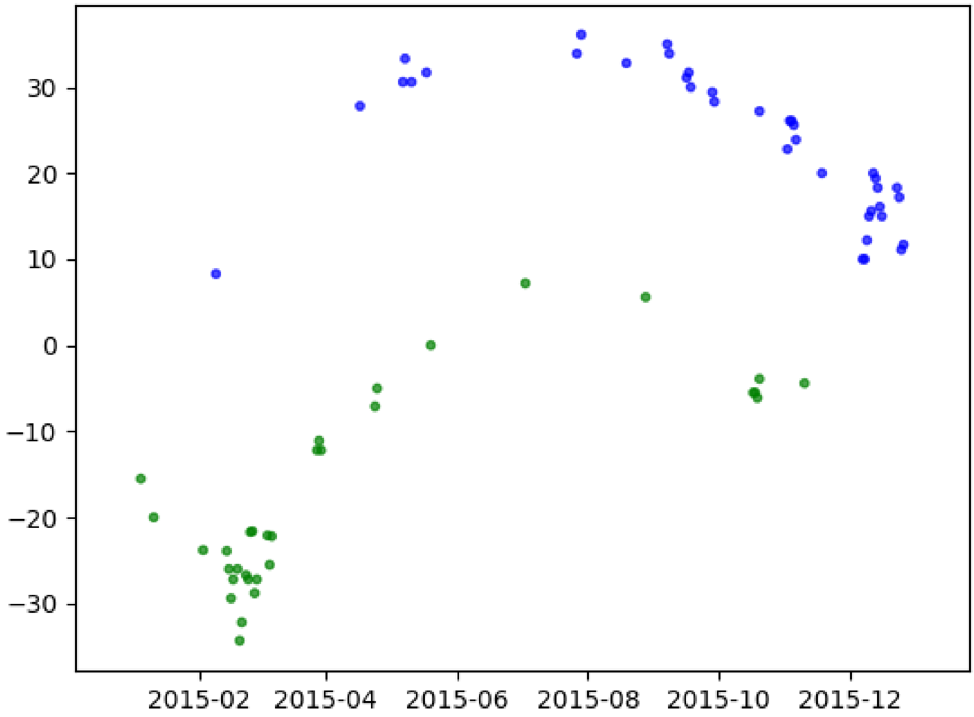


Out[2]:

```
<matplotlib.collections.PolyCollection at 0x7efe68d75668>
```

In [44]:

```
%matplotlib notebook
import pandas as pd
import numpy as np

df = pd.read_csv('data/C2A2_data/BinnedCsvs_d400/fb441e62df2d58994928907a91895ec62c2c42
e6cd075c2700843b89.csv')
MIN = df[df['Element'] == 'TMIN'].groupby('Date').agg({'Data_Value': np.min}).reset_ind
ex()
test = MIN[pd.to_datetime(MIN['Date']) >= pd.Timestamp('2015-01-01')]
MIN = MIN[pd.to_datetime(MIN['Date']) < pd.Timestamp('2015-01-01')]
MAX = df[df['Element'] == 'TMAX'].groupby('Date').agg({'Data_Value': np.max}).reset_ind
ex()
test1 = MAX[pd.to_datetime(MAX['Date']) >= pd.Timestamp('2015-01-01')]
MAX = MAX[pd.to_datetime(MAX['Date']) < pd.Timestamp('2015-01-01')]
MIN = MIN[MIN.Date != '2008-02-29']
MIN = MIN[MIN.Date != '2012-02-29']
MIN.reset_index(drop = True, inplace = True)
MAX = MAX[MAX.Date != '2008-02-29']
MAX = MAX[MAX.Date != '2012-02-29']
MAX.reset_index(drop = True, inplace = True)
test['max'] = test1['Data_Value']
test.rename(columns = {'Data_Value':'min'},inplace = True)
test.reset_index(drop = True, inplace = True)
test['min_all'] = 0
test['max_all'] = 0
for i in range(0,365):
    min = 100
    max = 0
    for j in range(0,10):
        if (min > MIN.loc[i+365*j]['Data_Value']):
            min = MIN.loc[i+365*j]['Data_Value']
        if (max < MAX.loc[i+365*j]['Data_Value']):
            max = MAX.loc[i+365*j]['Data_Value']
    test.set_value(i, 'min_all', min)
    test.set_value(i, 'max_all', max)
max_extreme = test[test['max']>test['max_all']]
min_extreme = test[test['min']<test['min_all']]
Date_array_max = np.array(max_extreme['Date'].tolist())
Date_array_max = list(map(pd.to_datetime, Date_array_max))
Date_array_min = np.array(min_extreme['Date'].tolist())
Date_array_min = list(map(pd.to_datetime, Date_array_min))
Min_array = (min_extreme['min']/10).tolist()
Max_array = (max_extreme['max']/10).tolist()
colors = ['green']*len(Date_array_min)+['blue']*len(Date_array_max)
Date = Date_array_min+Date_array_max
Value = Min_array+Max_array
plt.scatter(Date,Value,s = 10,c = colors,alpha=.7)
```



```
Out[44]:

<matplotlib.collections.PathCollection at 0x7efe6136b9b0>
```