Read about matplotlib for plotting (http://matplotlib.org).

Read about NumPy and SciPy third party libraries. Pay special attention to Multivariate (2D) data interpolation.

Part 1:

Repeat lab 05 using matplot lib to illustrate the specified stock prices.

Part 2:

AirMonitor

Air monitor stations are placed throughout the metroplex in order to measure the concentration of air borne pollutants every hour. Each monitor is identified by a unique id (long integer). For convenience, we name each station (for example, ‘tollway’, ‘H1’, 'H2', 'H3', ‘LBJ’, etc). We track the location of each station in degrees longitude and latitude (for example, -96.7 and 103.45). Each monitor collects a certain kind of data (for example "ozone"). And each monitor maintains a collection of timestamped data for the daily data observations (collected hourly). Data values are integers and represent parts per million. –1 represents a missing data observation. Data lines start with the number of hourly observations available.

We read this data from a file: Each station is characterized by one lines in a text file: id, name, longitude latitude, metric_name, val0, val1, val2, val3…..val23

where val0 is the value sampled at hour 0:00 (the beginning of the day). The value may be missing (-1).

```
13 Ft._Worth_Northwest/A302 32.805556 -97.356389 ozone 62 55 60 50 60 64 67 51 49 59 58 69 65 76 58 55 42 35 43 56 54 61 47 55
17 Keller 32.922500 -97.281944 ozone 67 55 58 48 57 69 69 55 47 58 54 69 72 83 60 55 44 36 44 53 51 62 49 52
31 Frisco 33.1325 -96.786111 ozone 66 54 50 44 58 71 71 61 49 59 60 61 69 81 71 58 50 40 42 52 49 59 51 55
….
```

We can ask questions of the air monitors:

- what is your short name?
- where are you located (what x? what y?)
- how far are you away from this location (x,y)?
- what data do you keep?
- what is the value at time t? Use linear interpolation to derive the result. For example, if we know that ozone was 46 at 2am and 48 at 3am. Then if we ask for the value at 2:15am, we interpolate the answer to be 46.5

AirDataCollector

We need to define another class of object to help us manage a collection of monitors. Our AirDataCollector class allows us to access a specified data monitor from the collection of monitors. An AirDataCollector functions like a back-end database managing the monitors. Use a python dictionary.

We can ask questions of the collectors:

- what monitors do you serve?
- give me monitor by id

- what is the value of *data* at time *t* for monitor *id*? (returns a single double)
- what are the values of *data* at time *t* for all monitors (returns list of doubles).

Air Data Application

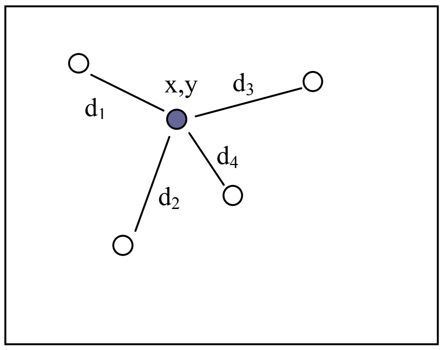
Our client application should build up the collection from the data file. Then the application should calculate the minimum containing rectangle based on the position of the air monitors.

Then prompt the user for an arbitrary position within the minimum containing rectangle (lon,lat).

Prompt the user for a specific time of the day (hours and minutes) and answer the question "at this time, what is the value of data right here (at this location – lon,lat)?"

Since we only know the actual values at a few critical places, our application will need to interpolate the value of the data at an arbitrary location (x,y). But how?

Foundation:
One strategy is known as the inverse distance (invd) approach. In the figure below, we want to estimate the value of the data (say, ozone) at (x,y) even though we only have readings at a few other points. We assume that monitor values close to x,y influence our estimate more than monitor values far away.



First we find the distances from (x,y) to each other monitor location. We sum the inverse distances into one total distance factor: D = $1/d_1 + 1/d_2 + 1/d_3 + ...$

Then we iterate (again) over each monitor in turn and get the value of the data monitor (v_i), the distance away from the point of interest, and calculate the total contribution of that monitor on the final estimated value: $c_i = v_i / di / D$.

Thus, the value at an arbitrary location x,y is the sum of the individual contribution.
$V = (c_1 + c_2 + c_3 + ... + c_n)$

As an example, let us assume that we know the 10 o'clock data for 4 monitors serving ozone data (10, 20, 20, and 30 parts per million -- ppm). Because of the relative locations, we calculate that the arbitrary point of interest is 5 units from the first monitor, 10 units from the second, 10 from the third and 4 units from the fourth monitor. The total distance factor is D = 1/5+1/10+1/10+1/4 = 0.65. According to our formulae above, each monitor influences our estimated value at the point of interest: the closer the monitor, the more the influence. For our example, the first monitor's value at 10 o'clock is 10 ppm. The contribution of that monitor to our estimate is 10 / 5 / 0.65. If we sum the individual contributions, we can formulate our guess for the point x,y:

probe value := ((10/5/0.65)+ (20/10/0.65) + (20/10/0.65) + (30/4/0.65)) = 20.7692

You must implement this inverse distance method to compute the requested value.

Use the scipy library to provide at least one alternative computation.

You must use python unittest module to be test driven.

Now plot the ozone in a 2D colormapped contour plot of the ozone at the specified time of day from the data source.