Homework 36

Due: Nov. 22, 2011, 2:25 pm

Task 1:

Download `HW3-6Starter.py` (or use your own program from the class if you had all working) and make sure it runs and does everything as expected.

Task 2:

In class, we finished with a function that can create a KML file that puts a push pin on Earth with coordinates and color specified by the user. Eventually we want to put multiple push pins on Earth. Let us see if we could do that.

As usual, we start simple. The specification of the function looks like below:

```
1 def writeMultiplePinsSimple(longitude, latitude, color):
2     ''' Create pin.kml (at a convenient location) that puts 10
        identical push pins on Earth, with the attributes
        specified by the three arguments '''
3     # input: number, number, string
4     # output: none
```

Remeber that a push pin is represented by the things between `<Placemark>` and `</Placemark>`.

Hint: Again, think about what parts of your string will not change and separate them out first. Then think about what part will be repeated 10 times. You will want to write a for-loop that iterates 10 times, each time appending something to the final string you are building. Among the things you append, there will be the changing parts which come from the arguments. After you run the program and call the function, look at the KML file and see if it looks like what you expect. (You probably will not be able to verify it in Google Earth since all 10 pins are stacked.)

Task 3:

Now let's write a function that produces a file which actually creates different pins according to user's input. The specification is as follows:

```
1  def writeMultiplePins(coordList, colorList):
2    ''' Create pin.kml (at a convenient location) that puts
       multiple push pins on Earth, with coordinates and
       colors specified by arguments. '''
3
4    # input:
5    # coordList is a list of lists, each sublist consisting of
       two numbers (longitude, latitude). For example, if I
       want to put 3 push pins at (28,36,0), (32,78,0),
       (12,33,0), coordList should be [[28,36],
       [32,78],[12,33]].
6    # colorList is a list of strings, each string specifying
       the color code for each push pin, in the same order its
       coordinates appear in coordList. e.g. ['ffff0000',
       ''ff00ee66'', ''ff724ad9']
7    # coordList has to have the same number of subLists as
       colorList has strings.
8
9    # output: none
```

Hint: This function is quite similar to the last one, except that: (1) The number of iterations is no longer fixed at 10, but depends on the inputs. (2) At each iteration, the coordinates and color you append to your final string will not be identical each time, but rather come from the inputs. (3) Try to iterate using indices, or you will hate yourself and me:

```
for index in range(0, len(coordsList)):
```

Task 4:

The function we just wrote can put multiple pins on Earth, given that someone specify the two input lists for it. Where do the lists come from? The twitter data we downloaded! Now let's consider the whole assembly line. It should look like this:

```
1
2  def tweetsToKML(filename, phrase1, phrase2):
3    ''' Creats pins.kml (at a convenient location) that puts a
       pin on Earth for each tweet present in the file
       specified by filename. A pin is red if its text
```

```
        contains phrase1, blue if its text contains phrase2, or
        green if contains both. '''
4
5     # input: string, string, string. phrase1 and phrase2 are
        the two phrases tracked when streaming the data. e.g.
        'coke' and 'pepsi'
6     # output : none
7
8     f = open(filename)
9     data = f.read()
10    f.close()
11
12    # tweets is a list of dictionaries, each dictionary
        representing a single tweet
13    tweets = digestData(data)
14
15    # compute coordinate list from tweets
16    coordList = getCoordList(tweets)
17
18    # compute color list from tweets
19    colorList = getColorList(tweets)
20
21    # write to kml file
22    writeMultiplePins(coordList, colorList)
23
24    return
```

What are the functions that we have not defined yet? For each one of them, write its definition lines (the `def` line, tooltip in triple quotes, input and output specification as comments). You do not have to write the bodies of the functions.

Handin

Email your program to `cs0931tas@cs.brown.edu` and title the file `'YOURNAME'HW3-6.py` — for example, `DylanFieldHW3-6.py`.