CS 5412 Assignment 1: Restaurant Geo-location

Objective

Your objective is to create a program which takes in some location information and then outputs a list of nearby restaurants. The list of existing restaurants will be provided in a CSV file discussed later in this document.

Implementation Details

Your implementation will take as user input either an address string such as “115 Rich Rd, Ithaca, NY 14850” or lattitude and longitude coordinates such as “42.415949, -76.481877”, and a radius in miles such as “0.5”. The program will then output a list of nearby restaurants that are within the given radius. Given the input string, you will need a way to find which addresses from the database are closest to that location. To aid in this task, it is useful to know that the distance between two points with coordinates {lat1,lon1} and {lat2,lon2} can be computed in the following way[1]:

$$distance = acos(sin(lat1)*sin(lat2)+cos(lat1)*cos(lat2)*cos(lon1-lon2))$$

The above formula would be enough to solve the entire problem easily if only the database of restaurants contained their GPS coordinates. Unfortunately, only the street addresses are in the database, so you will need some way to convert the addresses into coordinates. For this purpose, you may use cloud-based resources such as the ones discussed later in this document. However, all such services we know of have relatively low quotas[2], so it is not possible to simply translate the entire database into coordinates at once. Furthermore, you will be graded partially on speed and such an approach would result in very long run times[3]. In order to narrow down the possible results, you may choose to limit the search to only locations with the same zip code as the query[4].

We realize that limiting to a single zip code might exclude some results that are within the specified radius, and solutions with this behavior are perfectly acceptable. On the other hand, a single zip-code might contain a large number of results which could quickly exhaust the quota. To avoid this problem, you should evaluate the available API options and devise a plan to use the quota efficiently. For example, some services may allow batch queries which allow multiple addresses to be geocoded in one call.

---

1 http://williams.best.vwh.net/avform.htm#Dist

2 For example, Google Geocoding API limits to 2,500 requests per day, 5 requests per second.

3 You are not allowed to cache the results of the queries. You need to start with our database and gather the rest of the information on the fly as queries arrive.

4 Note that this means you'll need a way to translate coordinates into addresses to get the zip code.

Restaurant Database

We will be providing you with a restaurant database[5] that has names, a bit of classification information, and address information for about 900,000 restaurants, formatted as a table in "comma-separated" format. The database is for use by CS5412 students only and should not be copied from our web site or shared outside of the class. Your program should access it directly from the course web site when running, and you will need to be on a Cornell network (or VPN). You have our permission to copy a few dozen rows to your own machine for debugging if that would be helpful.

Cloud-based resources

There are many different cloud services which you might want to use in order to implement your solution. For example, Google provides a free Geocoding API[6], which can translate a string address such as “115 Rich Rd, Ithaca, NY 14850” into latitude and longitude pairs. There are other APIs provided by Microsoft and Apple that provide similar functionality, and you are free to use whichever you like provided you only use free resources.

We realize that there are some cloud platforms that would solve this entire project for you (given the address, they would give you that list of nearby restaurants). **Your solution is not allowed to use such an API**. You need to solve this as a query against the database we provided, because we want you to think about how to solve that problem--combining cloud resources with local data.

Extra Credit (GUI)

For extra credit, your GUI should also display the restaurants on a Google map centered at the address given in the query, using the push-pin mashups that Google offers and popping up the data from the database when the cursor clicks or hovers over a pushpin.

Submission

Your submission should include the following items, in a .zip file:

1. Write-up of your project in pdf format, explaining how the program works, the design decisions you faced and your reasoning behind implementing your solution in the way that you did. Be sure to include any web services and APIs you used. Name this file **writeup.pdf**.
2. Output for the queries which we will announce 48 hours before the deadline in a

5 http://www.cs.cornell.edu/Courses/CS5412/2015sp/_cuonly/restaurants_all.csv
6 https://developers.google.com/maps/documentation/geocoding/

file called **queries.txt**. The output should be of the format[7]:

<query1 address or coordinates>|<distance>|<run time>

<address of restaurant>|<distance from query1 address>

<address of restaurant>|<distance from query1 address>

.

.

.

<address of restaurant>|<distance from query1 address>

<blank line>

<query2 address or coordinates>|<distance>|<run time>

<address of restaurant>|<distance from query2 address>

<address of restaurant>|<distance from query2 address>

.

.

.

<address of restaurant>|<distance from query2 address>

<blank line>

<query3 address or coordinates>|<distance>|<run time>

<address of restaurant>|<distance from query3 address>

<address of restaurant>|<distance from query3 address>

.

.

.

<address of restaurant>|<distance from query3 address>

3. For those doing the extra credit, an image of the map for each query with push-pins should be included. Each of these three images should display the pop-up metadata for one of the locations. Concatenate the images into a single .jpg file called **maps.jpg**.
4. The source code for your project. This can be in either C, C#, C++, Python, Java or Javascript. Put all code in a subdirectory called **src**.
5. Documentation for how to compile and run the program in plaintext format. If

7 Note that <> are not included in the output.

your project requires installation of dependencies in order to run, you should provide instructions on how to do so in this document. Call this file **README.txt**.

In the end, you should have a directory structure that unzips like this:

\submission

\submission\writeup.pdf

\submission\README.txt

\submission\queries.txt

\sumbission\maps.jpg

\src*[8]

8 * represents your source files, which can be named however you like and organized in an directory structure you like as long as it's under src.