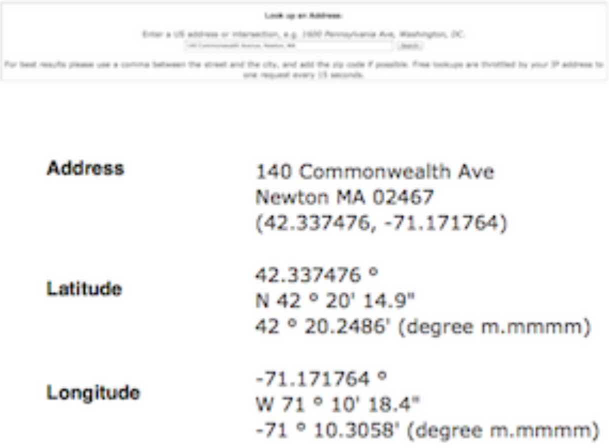
The site http://geocoder.us/demo.cgiDC allows you to enter an address, and it finds the latitude and longitude of that location. Some screen shots are shown below:

Portion of:
http://geocoder.us/demo.cgi?address=140+Commonwealth+Avenue%2C+Newton%2C+MA



Look up an Address:

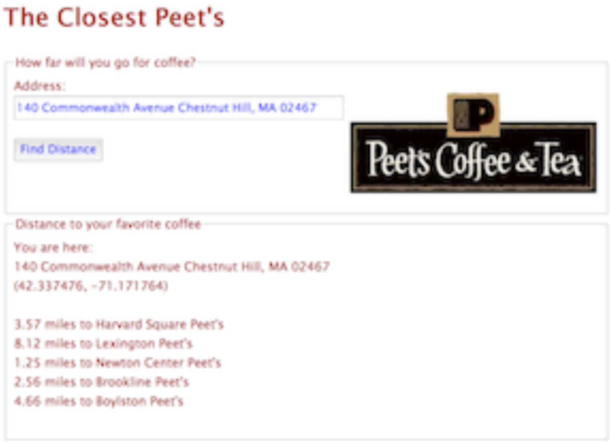
Enter a US address or intersection, e.g. *1600 Pennsylvania Ave, Washington, DC.*

140 Commonwealth Avenue, Newton, MA

For best results please use a comma between the street and the city, and add the zip code if possible. Free lookups are throttled by your IP address to one request every 15 seconds.

Address	140 Commonwealth Ave Newton MA 02467 (42.337476, -71.171764)
Latitude	42.337476 ° N 42 ° 20' 14.9" 42 ° 20.2486' (degree m.mmmm)
Longitude	-71.171764 ° W 71 ° 10' 18.4" -71 ° 10.3058' (degree m.mmmm)

Now, there are a lot of opinions about coffee. Some people like Starbucks, some like Dunkin Donuts. I like Peet's the best. Below is a screen shot of a webpage that takes an address, and computes the distance to several nearby Peet's locations.



The Closest Peet's

How far will you go for coffee?

Address:

140 Commonwealth Avenue Chestnut Hill, MA 02467

Distance to your favorite coffee

You are here:
140 Commonwealth Avenue Chestnut Hill, MA 02467
(42.337476, -71.171764)

3.57 miles to Harvard Square Peet's
8.12 miles to Lexington Peet's
1.25 miles to Newton Center Peet's
2.56 miles to Brookline Peet's
4.66 miles to Boylston Peet's

Write a page of your own that allows a user to enter an address, and return the locations and distances to several coffee shops (or other store of your liking) . You can use Peet's, Starbucks, Dunkin Donuts, or what ever coffee/tea or whatever places you like. They can be chains or independent businesses.   Just make sure you have at least 5 locations. When the user clicks the submit button the page should use one call to `file_get_contents()`, show the address entered with latitude and longitude, and also show the distance from the address entered to all the coffee places you entered.

To help you develop your pattern, go to *http://geocoder.us/* and enter an address. When that page finds the latitude and longitude, do a “view source” in your browser, copy the entire source, and paste it into the “Text” area of http://cslab.bc.edu/~cs254/Demos/L15/matches.php. Then you can enter your pattern and see if you’re finding the latitude and longitude correctly.

A location is a latitude and longitude pair, so it makes sense to store it in a PHP array with two elements, like this:

```
$location = array("latitude" => $latitude, "longitude" => $longitude);
```

I stored my Peet's stores and locations in a multidimensional array as shown below.

```
$peets = array(
   "Harvard Square" => array( "address"=>"100 Mount Auburn Street
Cambridge, MA 02138",
                              "location"=> array("latitude"=>42.372663,
"longitude"=>-71.120697)),
…
);
```

I looked up the addresses and found the latitude and longitude manually (i.e. using geocoded.us) to complete the array. Do not lookup all the latitude, longitude pairs every time you load the page. Each page load should only lookup the current address the user entered.

Notice that $peets['Harvard Square']['address'] is a string, but $peets['Harvard Square']['location'] is an array with two elements, latitude and longitude.

To encode the address portion of the URL, use the urlencode() php function. This function encodes white spaces and special characters so they can be used in a URL.

Structure your page with displayForm() and handleForm(). Split your handleForm() into three functions as shown below. As a hint, my complete page (without CSS) was around 100 lines of php.

```
    function handleForm($address){
          // your code here.
          // This function calls getLocation() and
displayDistances($loc)
    }
    function getLocation($page, $pattern)
    {
            // your code here
            // this function returns an array with two elements,
latitude and longitude
            return $location;
    }
    function displayDistances($location){
          // your code here
          // this function calls the funciton dist()
     }
```

To find the distance between two latitude/longitude pairs, you can use the following php code[*]:

```
    function haversin($z)
    {
        return 0.5*(1-cos($z));
    }

    function dist($loc1, $loc2)
    {
        $R = 3963; // earth radius, miles
        $dp = deg2rad($loc2['latitude']-$loc1['latitude']);
        $dt = deg2rad($loc2['longitude']-$loc1['longitude']);
        $h  = haversin($dp) +
                   cos(deg2rad($loc1['latitude']))*
                   cos(deg2rad($loc2['latitude']))*haversin($dt);
        $d  = 2*$R*asin(sqrt($h));
        return $d;
    }
```

Useful demos:
http://cslab.bc.edu/~cs254/demos/lweek9/googlestock.php.txt
http://cslab.bc.edu/~cs254/demos/lweek9/matches.php

[*] Formula obtained from http://en.wikipedia.org/wiki/Haversine_formula