Project 3

due at midnight on Wed Sep 30 (60 points)

Like project 2, your program for this project will do a distance calculation. However, the calculation is a good deal more complex, and uses mathematical functions. Also, in evaluating this program, I will use a rubric (PDF) and consider the way you have formatted, commented, and tested your program.

Your assignment will calculate an approximate distance in kilometers and miles between two points specified in decimal degrees of longitude and latitude. The first point will always be LIU Brooklyn, which is at 40.690° North and 73.980° West. Latitude is negative in the southern hemisphere, and longitude is negative in the western hemisphere, so we will specify these as `40.690` and `-73.980`. (You can find the longitude and latitude of any point using Google maps by right-clicking and selecting "What's here?")

The second point is to be provided as input by the user running your program. Here is a sample transcript of one run of the program. As the user, I have provided the coordinates of the Space Telescope Science Institute in Baltimore.

```
Enter your latitude: 39.3325
Enter your longitude: -76.6231
The distance to LIU Brooklyn is 271.093 km
which is 168.449 miles.
```

The Haversine formula

This description was adapted from an interactive calculator by Andrew Hedges at http://andrew.hedges.name/experiments/haversine/

The Haversine formula is a way to calculate the *great circle* distance (*dist*) between two points on the globe (point A specified as *latA*, *lonA*, and point B specified as *latB*, *lonB*). It's an approximation because it does not take into account the non-spheroidal shape of the Earth. (It will tend to overestimate trans-polar distances and underestimate trans-equatorial distances.)

$$a = \left[\sin\left(\frac{lat_A - lat_B}{2}\right)\right]^2 + \cos(lat_A) \cdot \cos(lat_B) \cdot \left[\sin\left(\frac{lon_A - lon_B}{2}\right)\right]^2$$

$$d = 2 \cdot R \cdot \arctan2\left(\sqrt{a},\ \sqrt{1-a}\right)$$

Where *R* is the radius of the Earth. For this value, let's use 6373 (kilometers), which optimizes the calculations for locations around 39 degrees from the equator (roughly the latitude of Washington DC).

Note: in C++, the arctan2 function is spelled `atan2`.

Radians

One problem you may have with the above formula is that C++ computes sine and cosine using Radians, but we have specified latitudes and longitudes in degrees. You will need to convert degrees to radians before applying the formula.

The conversion is that 360° = 2π radians, so any value in degrees can be multiplied by (π/180) to convert to radians. (Use the constant `M_PI` from the `cmath` library.)

Submit

Please name your program `p3geo.cpp` (all lower-case and no spaces) and submit to this dropbox for project 3.