**CS205 C/C++ Programming Lab Assignment 1**

**Name:** 钟兆玮 (Zhaowei Zhong)

**SID:** 11611722

**Part 1 - Analysis**

The problem is to calculate the distance between two cities by using the data which user entered.

Assume the Earth is a perfect sphere. Let all angles be measured in signed degrees (negative latitude means South; negative longitude means West).

Let phi = 90 - latitude. The North Pole has phi = 0, the South Pole has phi = 180, and 0 <= phi <= 180.

Let theta = longitude. Greenwich, England, has theta = 0, and -180 <= theta <= 180.

Let the angles for the two points be (phi1, theta1) and (phi2, theta2). Then compute c = sin(phi1)\*sin(phi2)\*cos(theta1-theta2) + cos(phi1)\*cos(phi2).

Then the shortest great circle distance between the two points is d = R\*arccos(c) where R is the radius of the earth in kilometers, and the arccosine is taken between 0 and 180 degrees, inclusive. (Earth radius: 6,371 km)

**Part 2 - Code**

#include <iostream>

#include <math.h>

#include <string>

#include <regex>

using namespace std;

struct City {

    string name;

    float latitude;

    float longitude;

};

float calc(City city1, City city2);

int main() {

    regex pattern("^([A-Za-z]+[,]\*[\\s]\*[,]\*[\\s]\*)+$");

    City city1;

    cout << "The first city: ";

    getline(cin, city1.name);

    if (!regex\_match(city1.name, pattern)) {

        cout << "The name of first city is illegal.";

        return 0;

    }

    cout << "The latitude and longitude of first city: ";

    cin >> city1.latitude;

    if (cin.fail() || city1.latitude >= 90.01 || city1.latitude <= -90.01) {

        cout << "The latitude of first city is illegal.";

        return 0;

    }

    cin >> city1.longitude;

    if (cin.fail() || city1.longitude >= 180.01 || city1.longitude <= -180.01) {

        cout << "The longitude of first city is illegal.";

        return 0;

    }

    if (char buf = getchar() != '\n' && buf != EOF) {

        cout << "The latitude or longitude of first city is illegal.";

        return 0;

    }

    City city2;

    cout << "The second city: ";

    getline(cin, city2.name);

    if (!regex\_match(city2.name, pattern)) {

        cout << "The name of second city is illegal.";

        return 0;

    }

    cout << "The latitude and longitude of second city: ";

    cin >> city2.latitude;

    if (cin.fail() || city1.latitude >= 90.01 || city1.latitude <= -90.01) {

        cout << "The latitude of second city is illegal.";

        return 0;

    }

    cin >> city2.longitude;

    if (cin.fail() || city1.longitude >= 180.01 || city1.longitude <= -180.01) {

        cout << "The longitude of second city is illegal.";

        return 0;

    }

if (char buf = getchar() != '\n' && buf != EOF) {

        cout << "The latitude or longitude of second city is illegal.";

        return 0;

    }

    cout << "The distance between " << city1.name << " and " << city2.name << " is " << calc(city1, city2) << "km";

    return 0;

}

float calc(City city1, City city2) {

    float phi1 = 90.0 - city1.latitude;

    float phi2 = 90.0 - city2.latitude;

    float theta1 = city1.longitude;

    float theta2 = city2.longitude;

    float c = sin(phi1) \* sin(phi2) \* cos(theta1 - theta2) + cos(phi1) \* cos(phi2);

    return 6371 \* acos(c);

}

**Part 3 - Result & Verification**

**Test Case #1:** Normal Case

Input:

Shenzhen: 22.55 114.1

图片包含 文字

描述已自动生成Tokyo: 35.42 139.46

**Test Case #2:** City name with blank or comma

Input:

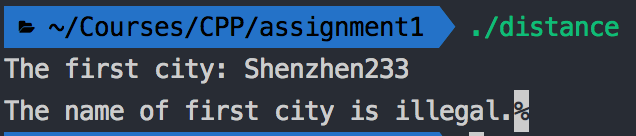
Shen Zhen: 22.55 114.1

图片包含 文字, 户外

描述已自动生成Tokyo, JP: 35.42 139.46

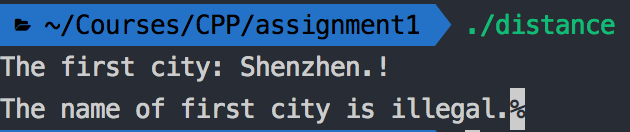
**Test Case #3:** City name with number

Input:

Shenzhen233: 22.55 114.1

**Test Case #4:** City name with other characters

Input:

Shenzhen.!: 22.55 114.1

**Test Case #5:** Negative latitude and longitude

Input:

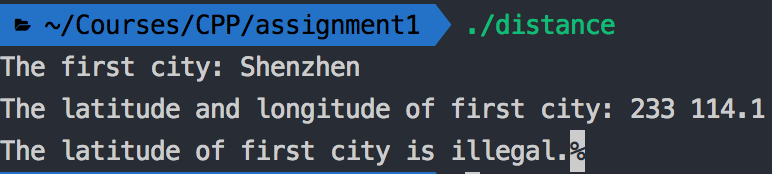
Rio de Janeiro: -22.9083 -43.1964

图片包含 文字

描述已自动生成Sydney: -33.865 151.209444

**Test Case #6:** Latitude/Longitude too large

Input:

Shenzhen: 233 114.1

**Test Case #6:** Latitude/Longitude too small

Input:

图片包含 道路

描述已自动生成Shenzhen: -233 114.1

**Part 4 - Difficulties & Solutions**

Determine whether the input city name string meets the rules is the main difficulty of this assignment. I use regex to test the input string, so it is much easier to determine the rules.