*CS205 C/ C++ Programming - Lab Assignment2

Name: 郑鑫颖

SID: 11912039

Part 1- Analysis

The problem include two parts. First load a .csv data file which contains the information of cities. Then promotes user to input two city names and searches in the table then based on the location information in the table, calculate the distance between the two cities.

Macros I used:

#include < iostream >:cin

#include < fstream >:fstream

#include < cstring >:strtok_r(), strlen(), strncpy()

#include< algorithm >:transform()

#include < math.h >:sin(),cos(),acos()

#include < string >:getline()

I solve the problem by two steps.

step I :

Construct a structure to store the information of a city.

Use ifstream to load the data into a city array.

I: use strtok to do the String segmentation.

II: use two boolean variables to denote wether country or name is null.

III: implement a function called convert to transform the string to double for latitude and longitude.

step II :

use cin to get user's input, traverse the city array to match the input city name.

I: implement a funtion called Matching to match the input name with the city name, if several possible names matched, give a list of this cities.

1)to be case insensitive, use transform function to transfrom to uppercase and then do the maching.

2)use erase to delete the whitespace in both end.

II： use the function in Assignment I to calculate the distance between two cities.

Error check:

I use some techniques to solve the following error, which all display the information without crashing the program.

During the loading of data:

No such file error / The length of the name exceed the max length / The array is full /

During the operation:

input format mismatch / Input is too short / No such city

Part 2- Code

My develop environment is:

```
C:\Users\联想>gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-pc-cygwin/7.4.0/lto-wrapper.exe
Target: x86_64-pc-cygwin
Configured with: /cygdrive/i/szsz/tmpp/gcc/gcc-7.4.0-1.x86_64/src/gcc-7.4.0/configure --srcdir=/cygdrive/i/szsz/tmpp/gcc
/gcc-7.4.0-1.x86_64/src/gcc-7.4.0 --prefix=/usr --exec-prefix=/usr --localstatedir=/var --sysconfdir=/etc --docdir=/usr/
share/doc/gcc --htmldir=/usr/share/doc/gcc/html -C --build=x86_64-pc-cygwin --host=x86_64-pc-cygwin --target=x86_64-pc-c
ygwin --without-libiconv-prefix --without-libintl-prefix --libexecdir=/usr/lib --enable-shared --enable-shared-libgcc --
enable-static --enable-version-specific-runtime-libs --enable-bootstrap --enable-__cxa_atexit --with-dwarf2 --with-tune=
generic --enable-languages=ada,c,c++,fortran,lto,objc,obj-c++ --enable-graphite --enable-threads=posix --enable-libatomi
c --enable-libcilkrts --enable-libgomp --enable-libitm --enable-libquadmath --enable-libquadmath-support --disable-libss
p --enable-libada --disable-symvers --with-gnu-ld --with-gnu-as --with-cloog-include=/usr/include/cloog-isl --without-li
biconv-prefix --without-libintl-prefix --with-system-zlib --enable-linker-build-id --with-default-libstdcxx-abi=gcc4-com
patible --enable-libstdcxx-filesystem-ts
Thread model: posix
gcc version 7.4.0 (GCC)
```

```cpp
#include <iostream>
#include <fstream>
#include <cstring>
#include <string>
#include <algorithm>
#include <math.h>
using namespace std;
#define RAD_TO_DEGREE(x) ((x)*3.14159 / 180.0)
#define LENGTH_NAME 35
#define LENGTH_ARRAY 1000
using namespace std;

struct City
{
    char* name = new char[LENGTH_NAME + 1];
    char* country = new char[LENGTH_NAME + 1];
    double latitude = 0;
    double longitude = 0;
};
int Matching(string name, City* cities, int index);
double Convert(char*);
double Distance(City city1, City city2);

int main()
{
    //load data
    char contents[100];
    char gap[] = ",";
    char* token = NULL;
    char* temp = NULL;
    City* cities = new City[LENGTH_ARRAY];
```

```
    ifstream infile;
    infile.open("world_cities.csv");
    if (infile.is_open())
    {
        int index = 0;
        while (!infile.eof())
        {

            bool flag_2 = false;
            bool flag_3 = false;

            infile.getline(contents, 200);

            token = strtok_r(contents, gap, &temp);
            if (strlen(token) > LENGTH_NAME)
            {
                cout << "The length of the city name is too long! The procedure
will exit. Currently at " << index + 1 << " row";
                return 0;
            }

            strncpy(cities[index].name,token, strlen(token));

            if (*temp == ',')
            {
                flag_2 = true;
                if (*(temp + 1) == ',')
                {
                    flag_3 = true;
                }
            }
            if (!flag_2)
            {
                token = strtok_r(NULL, gap, &temp);
            }
            else if (!flag_3)
            {
                token = strtok_r(NULL, gap, &temp);
                strncpy(cities[index].country,token, strlen(token) );
                if (strlen(token) > LENGTH_NAME)
                {
                    cout << "The length of the country name is too long! The
procedure will exit. Currently at " << index + 1 << " row.\n";
                    return 0;
                }
            }
            else
                cities[index].country = NULL;

            token = strtok_r(NULL, gap, &temp);
            cities[index].latitude = Convert(token);
            token = strtok_r(NULL, gap, &temp);
            cities[index].longitude = Convert(token);
            index++;
            if (index >= LENGTH_ARRAY)
            {
                cout << "The space is already full! The procedure will exit!
Currently at " << index << " row \n";
```

```
            return 0;
        }
    }
    infile.close();

    //operation
    string name;
    int* input = new int[2];
    int number = 0;
    do
    {
        bool jump = false;
        cout << "Please enter the city's name:";
        getline(cin, name);
        transform(name.begin(), name.end(), name.begin(), ::toupper);
        if (name == "BYE")
        {
            cout << "Exit the procedure! \n";
            return 0;
        }
        else if (name.length() < 3)
        {
            cout << "The input is too short!\n";
            continue;
        }
        else
        {
            for (int i = 0; i < name.length(); i++)
            {
                if (name[i] != ' ' && name[i] != ',' && !isalpha(name[i]))
                {
                    cout << "There are some incorrect format.\n";
                    jump = true;
                    break;
                }
            }
            if (jump)
            {
                continue;
            }
        }

        int matched = Matching(name, cities, index);

        if (matched == -1)
        {
            cout << "Which one is the right city? ";
            continue;
        }
        else if (matched == -2)
        {
            cout << "There's no such city.\n";
            continue;
        }

        input[number] = matched;
        if (number == 1)
        {
```

```
                if(input[0]==input[1])
                    cout<<"The distance between " << cities[input[0]].name << "
and " << cities[input[1]].name << " is " << 0 << " km" << endl;
                else cout << "The distance between " << cities[input[0]].name <<
" and " << cities[input[1]].name << " is " << Distance(cities[input[0]],
cities[input[1]]) << " km" << endl;
            }
            number = (number + 1) % 2;
        } while (true);
    }
    else
    {
        cout << "The file doesn't exist or cannot open the file.";
    }

    return 0;
}

int Matching(string name, City* cities, int index)
{
    //trim right
    name.erase(name.find_last_not_of(" ") + 1);
    // trim left
    name.erase(0, name.find_first_not_of(" "));
    string matched[LENGTH_ARRAY];
    int number = 0;
    int last_index = 0;

    for (int i = 0; i < index; i++)
    {
        string city_name = cities[i].name;
        transform(city_name.begin(), city_name.end(), city_name.begin(),
::toupper);
        if (city_name.find(name) < city_name.length())
        {
            if (name.length() == city_name.length())
            {
                return i;
            }
            else
            {
                last_index = i;
                matched[number++] = cities[i].name;
            }
        }
    }
    if (number == 1)
    {
        return last_index;
    }
    else if (number == 0)
    {
        return -2;
    }

    for (int i = 0; i < number; i++)
    {
        cout << matched[i] << endl;
```

```
    }
    return -1;
}

double Convert(char* s)
{
    int index = 0;
    int integer = 0;
    double decimal = 0;
    bool negative = false;

    if (*(s + index) == '-')
    {
        negative = true;
        index++;
    }

    while (*(s + index) != '\0' && *(s + index) != '.')
    {
        //interger
        if (negative)
        {
            integer = integer * 10 - (*(s + index) - '0');
        }
        else
            integer = integer * 10 + (*(s + index) - '0');
        index++;
    }
    //decimal
    if (*(s + index) == '.')
    {
        index++;
        double y = 10;
        while (*(s + index) != '\0')
        {
            if (negative)
            {
                decimal = decimal - ((double)*(s + index) - '0') / y;
            }
            else
                decimal = decimal + ((double)*(s + index) - '0') / y;
            y *= (double)10;
            index++;
        }
    }
    return integer + decimal;
}
double Distance(City city1, City city2)
{
    double phi1 = RAD_TO_DEGREE(90 - city1.latitude);
    double phi2 = RAD_TO_DEGREE(90 - city2.latitude);
    double theta1 = RAD_TO_DEGREE(city1.longitude);
    double theta2 = RAD_TO_DEGREE(city2.longitude);
    double c = sin(phi1) * sin(phi2) * cos(theta1 - theta2) + cos(phi1) *
cos(phi2);
    return 6371 * acos(c);
}
```

Part 3- Result & Verification

Test case #1:

```
nancy@LAPTOP-6UPALDO7:~/c++file/assignment2$ ./as2
Please enter the city's name:shenzhen
Please enter the city's name:beijing
The distance between Shenzhen and Beijing is 9707.47 km
Please enter the city's name:
```

Test case #2:

```
nancy@LAPTOP-6UPALDO7:~/c++file/assignment2$ ./as2
Please enter the city's name:new york
Please enter the city's name:shanghai
The distance between New York City and Shanghai is 7923.69 km
Please enter the city's name:
```

Test case #3: if we type she, there will be a matching list and asks user to choose the right one, if we type new york then new york city is retrieved.

```
Please enter the city's name:she
Sheffield
Shenyang
Shenzhen
Which one is the right city? Please enter the city's name:shenyang
Please enter the city's name:new york
The distance between Shenyang and New York City is 13300.2 km
Please enter the city's name:
```

Test case4: Exit the program.

```
Please enter the city's name:bye
Exit the procedure!
```

Error check:

1:if the length of name is only 25.

```
nancy@LAPTOP-6UPALDO7:~/c++file/assignment2$ ./as2
The length of the country name is too long! The procedure will exit. Currently at 418 row.
```

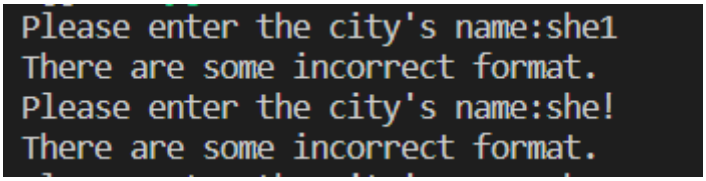
2:if the size of array is only 800.

```
nancy@LAPTOP-6UPALDO7:~/c++file/assignment2$ ./as2
The space is already full! The procedure will exit!  Currently at 800 row
```

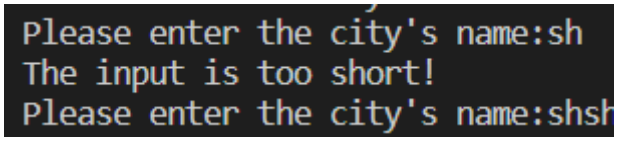
3:No such file.

```
nancy@LAPTOP-6UPALDO7:~/c++file/assignment2$ ./as2
The file doesn't exist or cannot open the file.nancy@L
```

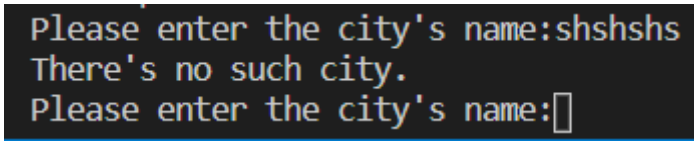
4: Input format mismatch



5: Input is too short



6: No such city



Part 4 - Difficulties & Solutions

There are totally five difficulies I met.

1. I spend lots of time to do the string segenmentation, firstly I used strtok(), I seached its usage in the Internet. And then I shifted to a safer version: strtok_r(). Next time I'll implement a string segementation function by myself.
2. strncpy() is also a safer version of strcpy().
3. Firstly, I don't know how to to covert string to double. Then I implemented a function to transform string to double.
4. While loading the data, my programm crashed because I don't take into account that the second and third term may be null, then I use two flags to check and fix the bug.
5. I don't know how to break outer loop. I asked my friends and set a flag to solve this problem.