CS205 C/ C++ Programming - Lab Assignment 2

Name: Jiachen Zhang

SID: 11713020

Part 1 - Analysis

> This Lab Assignment is an upgraded version of lab Assignment 1.
> We need to write a program with similiar function but more easier to use.

> I finish this Lab Assignment with the following small steps.
>
> - First, we should load the `.csv` file into the array, whose element type is struct.
> - Second, we should get the data of latitude and longitude by searching the city name in the struct array.
> - Third, we should deal with this case:
>   If people type `"New York"` , then `"New York City"` must be retrieved.
>   However, if users only type `"New"` (minimum acceptable length), it can match several cities.
>   So, the list of the matched cities must be displayed, prompting the user for typing a more precise one.
> - Finally, the names of the cities and their distance must be displayed as we did in lab Assignment 1.

Part 2 - Code

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <malloc.h>

#define F_PATH "world_cities.csv"
#define MAX_LINE 1
#define MAX_NAME_LENGTH 25
#define MAX_ARRAY_SIZE 800

typedef struct location
{
        char City_Name[MAX_NAME_LENGTH + 1];
        char Province_Name[MAX_NAME_LENGTH + 1];
        char Country_Name[MAX_NAME_LENGTH + 1];
        double Latitude;
        double Longitude;
} Location, *pLocation;
void data_Initial(void);
int data_Search(char *);
int readFile(void);
char *strtok_new(char *, char const *);
void printLocation(pLocation);
int getData(void);
void process(void);
int read_state; //The number of the data we get
pLocation data[MAX_ARRAY_SIZE];
double compute(double, double, double, double);

int main()
{
        printf("-------------------------------------------------------------------\n");
        data_Initial();
        printf("-------------------------------------------------------------------\n");
        while (1)
        {
                process();
                system("pause");
        }
        // system("pause");
        return 0;
}

void process(void)
{
        int index_City = 0;

        index_City = getData();
        printLocation(data[index_City]);
        char *city_1 = data[index_City]->City_Name;
        double phi_1 = 90 - data[index_City]->Latitude;
        double theta_1 = data[index_City]->Longitude;
        printf("\n\n");
        index_City = getData();
        printLocation(data[index_City]);
        char *city_2 = data[index_City]->City_Name;
        double phi_2 = 90 - data[index_City]->Latitude;
        double theta_2 = data[index_City]->Longitude;

        double distance = compute(phi_1, theta_1, phi_2, theta_2);
        printf("\n\n-------------------------------------------------------------------\n");
        printf(">>> The distance between %s and %s is %f km.\n", city_1, city_2, distance);
        printf("-------------------------------------------------------------------\n\n");
        return;
}

int getData(void)
{
        int i = -1;
        char city_Name[MAX_NAME_LENGTH + 1];
        int length = 0;
        printf("-------------------------------------------------------------------\n");
        printf("Please input the name of the city you want, or <bye> to quit.......\n> ");
        while (i < 0)
        {
                fflush(stdin);
                fgets(city_Name, MAX_NAME_LENGTH, stdin);
                /* 解决fgets()会把'\n'也读取的问题 */
                city_Name[strlen(city_Name) - 1] = '\0';
                printf("-------------------------------------------------------------------\n");

                if (!strcasecmp("bye", city_Name))
                {
                        exit(0);
                }
                length = strlen(city_Name);
                if (length < 3 || length > MAX_NAME_LENGTH)
                {
                        printf("Please input another name which is more specific...\n");
                        continue;
                }
                else
                {
                        i = data_Search(city_Name);
                }
        }

        return i;
}

/* return the index of the city, if more than one, print the following and return -1, */
int data_Search(char *city_Name)
{

        int length = strlen(city_Name);
        // printf("city_Name is %s\t%d\n", city_Name, length);
        int i = 0;
        int j = 0;
        int num = 0;
        while (i++ < read_state)
        {
                j = 0;
                /* 需要控制用户输入长度不超过MAX_NAME_LENGTH */
                if (!strnicmp(data[i]->City_Name, city_Name, length))
                { /* 前length个字母不区分大小写匹配成功 */
                        if (!strnicmp(data[i + 1]->City_Name, city_Name, length))
                        {
                                printf("\n-------------------------------------------------------------------\n");
                                printf("There are several cities whose name include <%s>, such as\n", city_Name);
                                while (i < read_state && !strnicmp(data[i]->City_Name, city_Name, length))
                                {
                                        printf("> %s\n", data[i]->City_Name);
                                        i++;
                                }
                                printf("-------------------------------------------------------------------\n\n");
                                printf("-------------------------------------------------------------------\n");
                                printf("Please input a more specific one again...\n> ");
                                return -1;
                        }
                        printf("\n-------------------------------------------------------------------\n");
                        printf("We have found the city <%s> succeedly.\n", data[i]->City_Name);
                        printf("-------------------------------------------------------------------\n");
                        // printf("i = %d\n", i);
                        return (i);
                }
        }
        printf("The name of the city you input <%s> is not found.\n");
        printf("Please input another again...\n");
        return -2;
}

void data_Initial(void)
{
        read_state = readFile(); // store the number of the data we readFile may less than what we want
        // printf("read_state = %d\n", read_state);
        switch (read_state)
        {
        case -1:
                printf("File is not found!\n");
                system("pause");
                exit(0);
                break;
        case MAX_ARRAY_SIZE:
```

```c
		printf("Finished reading the file\n");
		break;
	default:
		printf("Finished reading the file\n");
		printf("Data is too much, some of which havn't been loaded!\n");
		break;
	}
	printf("We get %d data\n", read_state);
}

int readFile(void)
{
	FILE *fp = NULL;
	char line[1000];
	/* 加载数据文件，若加载失败报告并结束 */
	int i = -1;
	if ((fp = fopen(F_PATH, "r")) != NULL)
	{
		printf("File is opened\n");

		/* 逐行读取并存入数据 */
		while (fgets(line, 100000, fp) && i < MAX_ARRAY_SIZE)
		{
			pLocation pCity = (pLocation)malloc(sizeof(Location));
			strncpy(pCity->City_Name, strtok_new(line, ","), MAX_NAME_LENGTH);
			strncpy(pCity->Province_Name, strtok_new(NULL, ","), MAX_NAME_LENGTH);
			strncpy(pCity->Country_Name, strtok_new(NULL, ","), MAX_NAME_LENGTH);
			pCity->Latitude = atof(strtok_new(NULL, ","));
			pCity->Longitude = atof(strtok_new(NULL, "\n"));
			data[i] = pCity;
			i++;
		}
		fclose(fp);
		fp = NULL;
	}
	return i;
}
/* 处理连续分隔符问题 Reference: http://www.it1352.com/482667.html */
char *strtok_new(char *string, char const *delimiter)
{
	static char *source = NULL;
	char *p, *riturn = 0;
	if (string != NULL)
		source = string;
	if (source == NULL)
		return NULL;
	if ((p = strpbrk(source, delimiter)) != NULL)
	{
		*p = 0;
		riturn = source;
		source = ++p;
	}

	return riturn;
}

double compute(double phi_1, double theta_1, double phi_2, double theta_2)
{
	double Pi = acos(-1.0);
	double c = 0.0;
	phi_1 = phi_1 / 180 * Pi;
	theta_1 = theta_1 / 180 * Pi;
	phi_2 = phi_2 / 180 * Pi;
	theta_2 = theta_2 / 180 * Pi;
	c = sin(phi_1) * sin(phi_2) * cos(theta_1 - theta_2) + cos(phi_1) * cos(phi_2);
	int R = 6371;
	double d = 0.0;
	d = R * acos(c);
	return d;
}

/* print the node of the struct with format */
void printLocation(pLocation l)
{
	printf("[");
	printf(" %s,", l->City_Name);
	printf(" %s,", l->Province_Name);
```

```
        printf(" %s,", l->Country_Name);
        printf(" %f,", l->Latitude);
        printf(" %f,", l->Longitude);
        printf("]\n");
        return;
}
```

Part 3 - Result & Verification

Test case #1:

```
If the file name is modified or file is disappeared.
Output:
File is not found!
```

Test case #2:

```
Input:
New
Output:
-------------------------------------------------------------------
There are several cities whose name include <new>, such as
> New Delhi
> New Orleans
> New York City
> Newcastle upon Tyne
> Newcastle
-------------------------------------------------------------------
```

Test case #3:

```
Input:
New York
Output:
-------------------------------------------------------------------
We have found the city <New York City> succeedly.
-------------------------------------------------------------------
[ New York City, New York, United States, 40.667000, -73.933000,]
-------------------------------------------------------------------
Please input the name of the city you want, or <bye> to quit.......
>
```

Test case #4:

```
Input:
beij
Output:
-------------------------------------------------------------------
We have found the city <Beijing> succeedly.
-------------------------------------------------------------------
[ Beijing, , China, 39.900000, 116.400000,]
-------------------------------------------------------------------
>>> The distance between New York City and Beijing is 10995.543029 km.
-------------------------------------------------------------------
```

Part 4 - Difficulties & Solutions

> `strtok` can't deal with continuous separators while `ProvinceOrState` may not `null` .
> So I use the rewrited version `strtok_new` .
> The last separator should be `\n` .

> I need to make the program to be easy used.
> So I choose to use the function `strnicmp` to compare the first n characters of the two strings with ignoring the cases of characters.

> For storing the data efficiently, I make the struct as the element of the array.
> In this way, I can search the data easily because the location is bundled with its name in a same struct.

> I also make some progress to make the program more beautiful.
> Welcome to use it.