國立清華大學

電機工程學系

資料結構 Homework1

學號：103061223

姓名：李俊穎(Lee Junying)

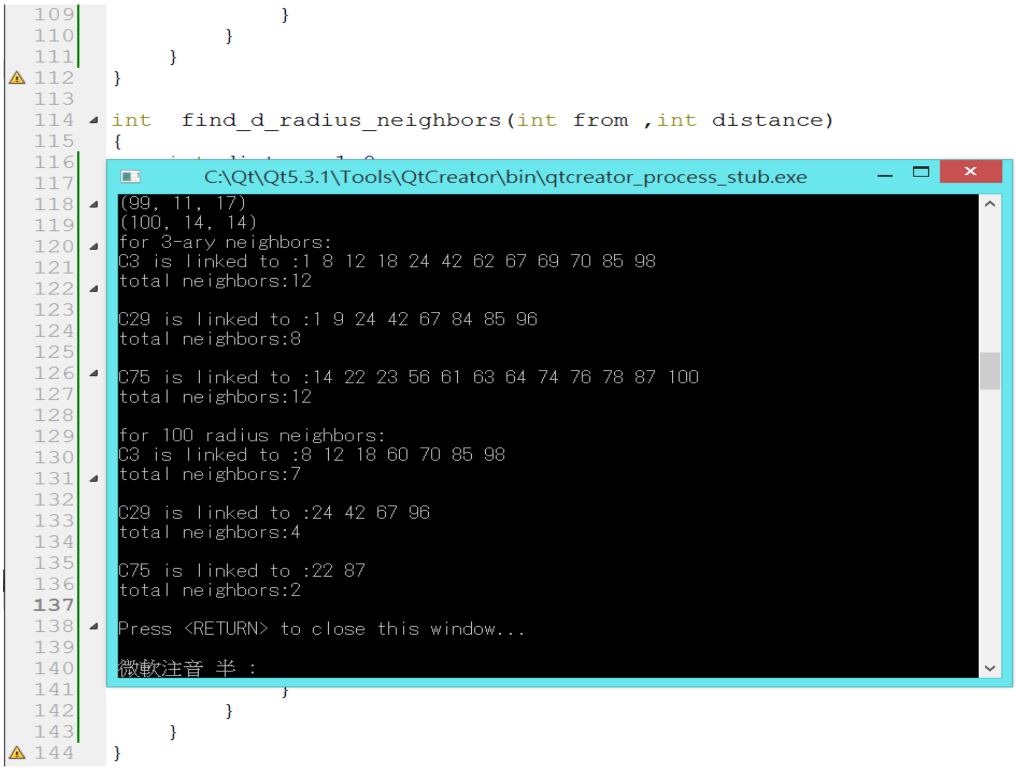
All of my source codes：

```
#include <stdio.h>
#include <iostream>
#include <string.h>
#include <fstream>
#include <algorithm>
#include <climits>

using namespace std;
int  D[1000][1000];
int find_k_ary_neighbors(int from , int hopes);
int find_d_radius_neighbors(int from , int distance);

int total_neighbors;
int visited[1000];
int NUM_CITY = 100;
int MAX_HOPS = 3;
int MAX_DISTANCE = 100;

int main()
{
    ifstream in_file ("highway_map");
    string line;
    int first, second, third,total=0;
    char       c_line[1000];
    if (!in_file.is_open()) exit(-1);

    for(int m=1; m<=NUM_CITY; m++){
        for(int n=1; n<=NUM_CITY; n++){
            D[m][n] = INT_MAX;
        }
    }
    int total_link = 0;
    while ( getline (in_file,line) ){
        // cout << line << '\n';
        strcpy(c_line, line.c_str());
        sscanf(c_line, "(%d,%d,%d)", &first, &second, &third);
    printf("(%d, %d, %d)\n", first, second, third);
    D[first][second] = third;
    D[second][first] = third;
    total_link++;
}

// Copy D[m][n] to D[n][m]
for(int m=1; m<=NUM_CITY; m++){
    for(int n=1; n<=NUM_CITY; n++){
        if(m!=n && m < n && D[m][n] != INT_MAX)
            D[n][m] = D[m][n];
    }
}

cout<<"for 3-ary neighbors:"<<endl;

// find_out_k_ary_neighbors
for(int from=1; from<=NUM_CITY; from++){
    find_k_ary_neighbors(from, MAX_HOPS);
    if(from==3||from==29||from==75)
    {
        cout<<"C"<<from<<" is linked to :";
        for(int a=1;a<=NUM_CITY;a++)
            if(visited[a]==1&&a!=3&&a!=29&&a!=75){
                cout<<a<<" ";
                total++;
            }
        cout<<endl<<"total neighbors:"<<total<<endl<<endl;
        total=0;
    }
    for(int a=1;a<=1000;a++)
    {
        visited[a]=0;
    }
}
cout<<"for 100 radius neighbors:"<<endl;

// find_out_d_radius_neighbors
```

```
    for(int from=1; from<=NUM_CITY; from++){
        find_d_radius_neighbors(from, MAX_DISTANCE);
        if(from==3||from==29||from==75)
        {
            cout<<"C"<<from<<" is linked to :";
            for(int a=1;a<=NUM_CITY;a++)
                if(visited[a]==1&&a!=3&&a!=29&&a!=75){
                    cout<<a<<" ";
                    total++;
                }
            cout<<endl<<"total neighbors:"<<total<<endl<<endl;
            total=0;
        }
        for(int a=1;a<1000;a++)
        {
            visited[a]=0;
        }
    }

    return 0;
}


int  find_k_ary_neighbors(int from,int hopes)
{
    visited[from]=1;
    for(int m=1; m<=NUM_CITY; m++){
        if(D[from][m]!=INT_MAX)
        {
            visited[m]=1;
            hopes--;
            if(hopes>0){
                find_k_ary_neighbors(m,hopes);
                hopes++;
            }
        }
    }
}
```

```
int  find_d_radius_neighbors(int from ,int distance)
{
    int distance1=0;
    visited[from]=1;
    for(int m=1;m<=NUM_CITY;m++)
    {
        if(D[from][m]!=INT_MAX)
        {
            if(visited[m]==1)
            {
                continue;
            }
            else
            {
                distance1=distance-D[from][m];
            }

            if(distance1>0)
            {
                visited[m]=1;
                //cout<<distance<<endl;
                find_d_radius_neighbors(m,distance1);
                distance1=distance1+D[from][m];
            }
            else
            {
                distance1=distance1+D[from][m];
            }
        }
    }
}
```

The execution results of my program：



```
                }
            }
        }
}

int  find_d_radius_neighbors(int from ,int distance)
{
```

C:\Qt\Qt5.3.1\Tools\QtCreator\bin\qtcreator_process_stub.exe

```
(99, 11, 17)
(100, 14, 14)
for 3-ary neighbors:
C3 is linked to :1 8 12 18 24 42 62 67 69 70 85 98
total neighbors:12

C29 is linked to :1 9 24 42 67 84 85 96
total neighbors:8

C75 is linked to :14 22 23 56 61 63 64 74 76 78 87 100
total neighbors:12

for 100 radius neighbors:
C3 is linked to :8 12 18 60 70 85 98
total neighbors:7

C29 is linked to :24 42 67 96
total neighbors:4

C75 is linked to :22 87
total neighbors:2

Press <RETURN> to close this window...

微軟注音 半 :
```

```
            }
        }
}
```

想法：

第一小題是要找出與 C3，C29，C75 相連三步以內的城市。而一開始用 for 迴圈先呼叫 function，並且傳入一開始要找的城市以及他的階層數 MAX_HOPES，並且將一開始的起始城市 visited 設為 1，以防下次進入重複的尋找路徑。接下來，function 內再用 for 迴圈將全部的城市都掃過一次，如果他們之間的距離 D 不是無限大(INT_MAX)，則代表這兩個城市是有相連的，於是將其設置為拜訪過的鄰居(將其 visited 存入 1)，並將階層數 hopes 減 1。如果階層數沒有被減到 0，代表尚未找完三步以內的鄰居，所以再設一個判斷條件並再次呼叫自己一次(recursive)，並在底下將 hopes 加回 1，如此才能在搜索到

底部時，回到上一層繼續往另外一條路找。之後再將全部 visited 陣列裡數字是 1 的全部印出來，即為解答。

第 2 題的做法與第一題就大同小異了，幾乎完全一樣，唯一不同的是為了防止重複無限迴圈，我設置如果找到原本 visited 陣列裡已經是 1 的城市，就讓其跳過那次的 for 迴圈，且以最大距離 100 往下減去做判斷，如果大於 0，代表就是在 100 公里以內的鄰居。這次的想法大概如上，謝謝。