电网建设造价模拟系统

1.项目简介

假设一个城市有n个小区，要实现n个小区之间的电网都能够相互接通，构造这个城市n个小区之间的电网，使总工程造价最低。请设计一个能够满足要求的造价方案。

2.项目功能要求

在每个小区之间都可以设置一条电网线路，都要付出相应的经济代价。n个小区之间最多可以有n（n-1）/2条线路，选择其中的n-1条使总的耗费最少。

3.算法设计

本题就是一道求最小生成树的裸题，对于求最小生成树，可以采用kruskal（克鲁斯卡尔）算法或prim（普里姆）算法求得。这里我采用prim算法解决该题。首先我采用邻接矩阵来保存整个图结构，由于这里给出的顶点名称为字母，所以我将每个字母映射成数字并将这种映射关系保存在char2Index数组中，每次获得name的时候使用getIndex函数从char2Index函数中取出该顶点对应的数字。对于Prim算法的思想就是用一个栈存放放入的顶点，每次取最末尾的顶点，找到与它相连，权值最小并且之前没有加入到栈中的顶点push到栈中，直到所有顶点都已经被push到栈中，最后依次输入内部的节点和对应的树。

4.核心代码及其功能

- **声明定义的变量，常量**

```
#define INF 0x7fffffff              //定义无穷大数为SIGNED INT的MAX
```

- **创建顶点功能**
  依次读入顶点并且将读入的顶点映射到数字的关系保存在char2Index数组中

```
void Graph::CreateV() {
    cout<<"请输入顶点的个数: ";
    cin>>vNum;
    cout<<"请依次输入各顶点的名称: ";
    getchar();
    for(int i=0;i<vNum;i++){            //依次获得顶点的名称并将其对应成数字
        char2Index[i]=getchar();        //映射关系保存在char2Index数组中
        getchar();
    }
    cout<<endl;
}
```

- **根据顶点名称获得其对应的数字**
  由于这里我采用了一种映射，所以我写了一个专门根据顶点名称获得其对应的数字的函数，返回对应的数字，如果没有找到返回-1。对于如何根据数字获得名称直接用char2Index数组索引就可以了。

```
int Graph::getIndex(char name) {
    for(int i=0;i<vNum;i++){            //从保存的节点中找到name的节点并返回对应的数字
        if(char2Index[i]==name){
            return i;
        }
    }
```

```
    return -1;                          //如果没有找到名字为name的顶点就返回-1
}
```

- **创建边功能**
  首先对整个邻接矩阵进行初始化，这里用将所有数组内的数值设置为INF来表示这两个顶点不联通，然后接受用户输入，并根据getIndex函数获得顶点名称对应的数字然后将读入的权值保存入map中，注意这里map为邻接矩阵，所以为对称矩阵，所以每次读入一次权值就需要将其对称的位置也设置为该权值，这里有专门检查用户是否存在输入非法的权值的功能。

```
void Graph::CreateE() {
    for (int i = 0; i < 1000; ++i) {
        for (int j = 0; j < 1000; ++j) {
            map[i][j]=INF;              //这里将两个顶点没有通路时其权值为INF
        }
    }
    for(int i=0;i<vNum*(vNum-1)/2;i++){   //有vNum个顶点所以最多有vNum*(vNum-1)/2条边
        cout<<"请输入两个顶点及边：";
        char vertex1,vertex2;
        getchar();
        vertex1=getchar();
        getchar();
        vertex2=getchar();
        cout<<vertex1<<" "<<vertex2<<endl;
        int index1=getIndex(vertex1),index2=getIndex(vertex2);  //获得顶点对应的数字
        if(index1<0||index2<0){         //检查用户输入是否合法
            cout<<"the vertex doesn't exist! Please reinput!"<<endl;
            i--;
            continue;
        }
        else{
            cin>>map[index1][index2];
            map[index2][index1]=map[index1][index2];       //邻接矩阵为对称
        }
    }
}
```

- **Prim算法**
  一开始保证之前使用后保存的权值和顶点内容都被清空。然后依次将找到的顶点推入栈中，下一步取得栈顶的顶点找到与其连接权值最小且没有已经加入到栈的顶点并push到栈中，直至所有顶点都加入完毕。这里我声明了一个visited数组用来保存该顶点是否已经被加入到栈中。

```
void Graph::Prim() {
    while(!primTree.empty())primTree.pop_back();    //清空原先prim树
    bool visited[vNum];                             //用于保存该顶点是否已经被连接入树
    memset(visited,false,vNum);
    cout<<"请输入起始顶点:";
    char start;                                     //最小树开始的起点
    cin>>start;
    primTree.push_back(start);                      //将起点放入primTree中
    for(int i=0;i<vNum-1;i++){                      //从所有边中找出权值最小的边的另一个顶点
        int vertex=getIndex(primTree.back());
        int vMin=INF,temp;
        for(int i=0;i<vNum;i++){
            if(map[vertex][i]<vMin&&!visited[i]){
                vMin=map[vertex][i];
                temp=i;
            }
        }
        visited[temp]=true;                         //将该顶点设置为已经被并入树中
        primTree.push_back(char2Index[temp]);       //将该顶点放入primTree中作为最后答案
    }
}
```

5.项目实例

```
/Users/kirito/CLionProjects/untitled/cmake-build-debug/untitled
**                    电网造价模拟系统                    **
=======================================================
**                A.---创建电网顶点                    **
**                B.---添加电网的边                    **
**                C.---构造最小生成树                  **
**                D.---显示最小生成树                  **
**                E.---退出程序                        **
=======================================================
请选择操作: A
请输入顶点的个数: 4
请依次输入各顶点的名称: a b c d

请选择操作: B
请输入两个顶点及边: a b 8
a b
请输入两个顶点及边: b c 7
b c
请输入两个顶点及边: c d 5
c d
请输入两个顶点及边: d a 11
d a
请输入两个顶点及边: a c 18
a c
请输入两个顶点及边: b d 12
b d
请选择操作: C
请输入起始顶点:a
请选择操作: D
a-<8>-b  b-<7>-c  c-<5>-d
```