项目说明文档

数据结构课程设计

——电网建设造价模拟系统

作 者 姓 名： 陈君

学 号： 2250420

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

# 1 分析

## 1.1 背景分析

## 我们面临着一个城市规划的重要任务：连接城市内的 n 个小区，确保它们之间的电网能够相互接通。我们的目标是设计一种造价最低的电网方案，以实现高效而经济的城市基础设施。

## 1.2 功能分析

1、创建电网顶点：用户可以通过选择菜单中的选项 A 来输入电网的顶点个数以及各个顶点的名称。这一步创建了电网的基本结构，为后续的边添加和最小生成树构建奠定了基础。

2、添加电网的边：通过选择菜单中的选项 B，用户可以输入两个顶点及它们之间的边权重，将这些信息存储到邻接矩阵中。用户可以反复添加边，直到输入 '?' 表示结束。该步骤用于构建完整的电网图。

3、构造最小生成树：

选择菜单中的选项 C，用户可以执行 Prim 算法来构建电网的最小生成树。通过选择起始顶点，算法逐步添加顶点，形成最小生成树。最小生成树的信息存储在 mintree 数组中。

4、显示最小生成树：

选择菜单中的选项 D，用户可以查看构建出的最小生成树的顶点和边信息。该信息以直观的方式展示了构建最小生成树的路径和造价。

5、退出程序：

选择菜单中的选项 E，用户可以退出程序，结束电网造价模拟系统的运行。

## 2.1 数据结构设计

如上功能分析所述，该系统需要存储一个图，为了方便表示顶点以及边的权值，可以采用邻接矩阵存储，二维数组 net[100][100]。同时需要边的结构体：start：起始顶点的名称。end：结束顶点的名称。cost：边的权重。用于存储最后算出的最小生成树。

## 2.2 类结构设计

## 主要成员变量：

## 邻接矩阵： 二维数组，表示电网的连接关系，记录各个顶点之间的边权重。

## 最小距离数组： 数组，用于 Prim 算法，记录当前已加入的顶点到其他顶点的最短距离。成员函数方面：主要是邻接矩阵的创建，以及prim算法算最小生成树以及输出最小生成树

## 2.3 成员与操作设计

class Net

{

public:

void prim();

void update\_lowcost(int p);

void setpointnum(int n) { pointnum = n; }

int getpointnum() { return pointnum; }

void createpoint();

void add\_sides();

int findbyname(char name);

void putout();

private:

int net[100][100] = { INT\_MAX };//使用邻接矩阵来存放电网图

int lowcost[100] ;//记录当前以加入的结点到达其他结点的最短距离

int pointnum = 0;

char names[100] = {0};

int namelength=0;

side mintree[100] = {};

int mintreenum = 0;

## };

## 2.4 系统设计

先cout这个系统可以使用的功能：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。然后等待用户输入选项，然后用switch（）语句跳转实现到各个功能的函数。

# 3 实现

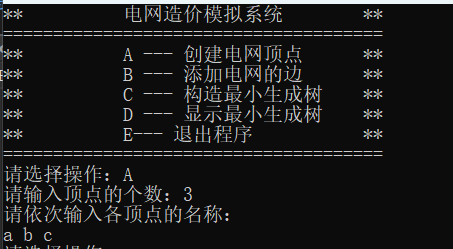
### 3.1 void Net::createpoint()//创造电网顶点

具体实现：提示用户输入顶点个数：输出提示信息，要求用户输入电网中顶点的个数。获取用户输入的顶点个数：使用 cin 从用户输入中获取电网中顶点的个数，并将其存储在类的成员变量 pointnum 中。

初始化邻接矩阵：

使用两层嵌套的循环，遍历电网的邻接矩阵。对于每一对顶点（i 和 j），将 net[i][j] 初始化为 INT\_MAX，表示它们之间没有直接的边连接。

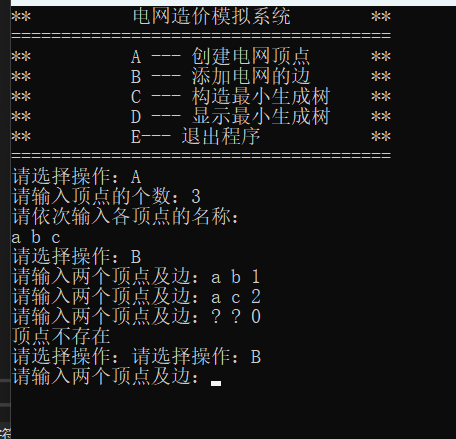
## 3.2.1 void Net::createpoint()功能截屏示例



## 3.2 void Net::add\_sides()//添加电网的边

具体实现：1、循环添加边：使用 do-while 循环，允许用户多次输入边的信息，直到输入 '?' 或者输入的边权重为 0 时结束。2、检查顶点是否存在：如果 row 或 column 的值为负，说明输入的顶点不存在，输出错误信息并结束函数。获取用户输入的边权重：

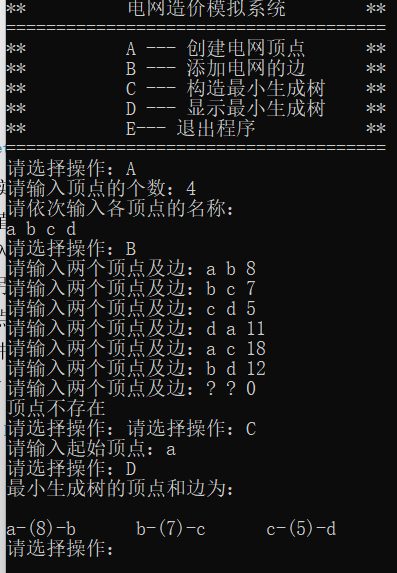
使用 cin 从用户输入中获取边的权重，并将其存储在邻接矩阵中对应的位置。为了表示无向图，同时更新对称位置的邻接矩阵。结束条件：如果用户输入 '?' 或者输入的边权重为 0，则结束循环。

3.2.1 void Net::add\_sides()功能截屏示例

## 3.3 void Net::prim()//Prim算法

具体实现：从连通网络 N = { V, E }中的某一顶点 u 0 出 发，选择与它关联的具有最小权值的边 发，选择与它关联的具有最小权值的边 ( u 0, v ) ，将其顶点加入到 将其顶点加入到生成树的顶点集合 生成树的顶点集合 U中。 以后每一步从一个顶点在 U 中，而另一个顶 点不在 U 中的各条边中选择 的各条边中选择权值最小的边 ( u, v ) , 把它的顶点加入到 把它的顶点加入到集合 U中。如此继续下去， 中。如此继续下去， 直到网络中的所有顶点都加入到生成树顶点集 直到网络中的所有顶点都加入到生成树顶点集 合 U中为止。

## 3.2.1 void Net::prim()功能截屏示例



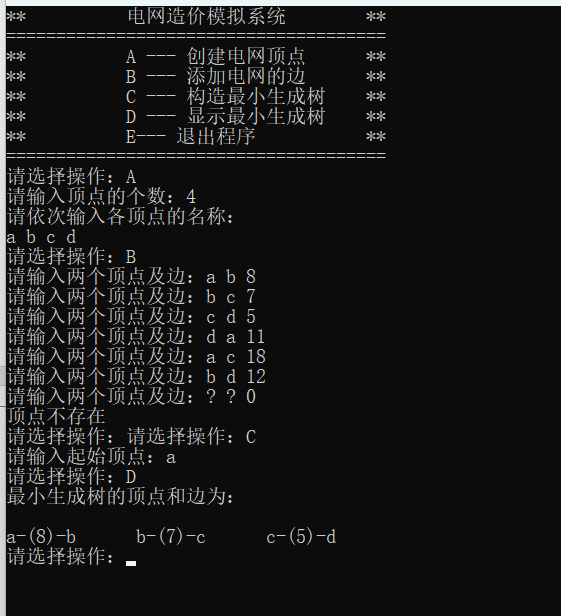
## 3.4 void Net::update\_lowcost(int p)//更新最小距离数组

具体实现： 将新加入顶点标记为已访问：将 lowcost[p] 设置为 0，表示该顶点已经加入最小生成树。遍历新加入顶点到其他点的距离：使用 for 循环，遍历所有顶点 i，检查从新加入的顶点 p 到其他顶点的距离。如果 net[p][i] 小于当前记录的 lowcost[i]，则更新 lowcost[i] 为更小的距离值。

## 3.6 总体系统的实现

先cout这个系统可以使用的功能：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。然后等待用户输入选项，然后用switch（）语句跳转实现到各个功能的函数。

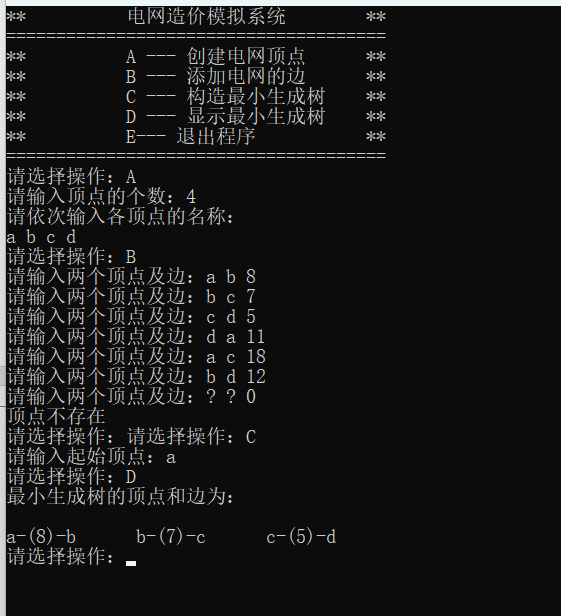
### }3.6.1 总体系统截屏示例



# 4 测试

## 4.1 系统总体功能测试功能测试

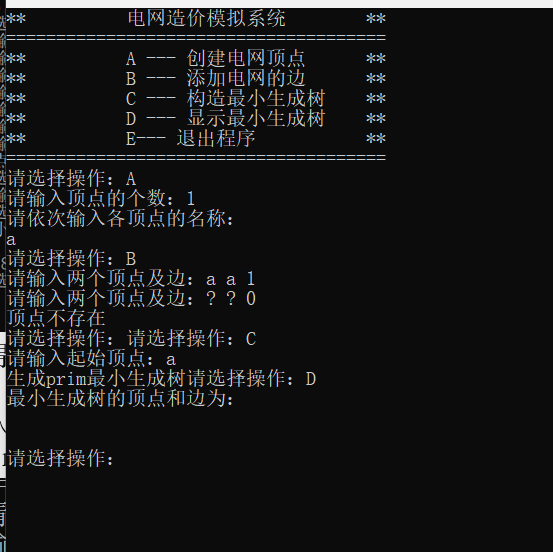
用例一：使用题目中所示的数据输入数据



**得到正确的结果**

4.2 边界条件测试

输入一个顶点

4.2.1截屏展示

4.3 错误输入测试

具有错误输入的提示。

# 

4.4有多余的边输入

