

# 题目八 电网建设造价模拟系统

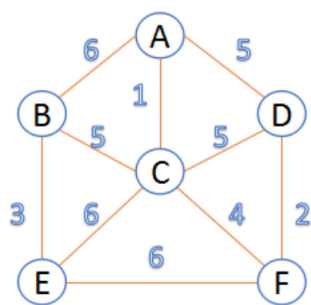
## 一.设计思路

---

电网造价系统要实现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。

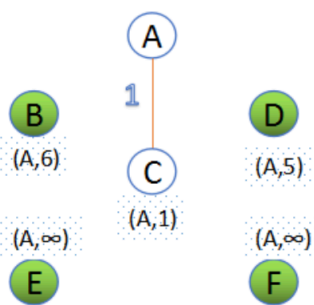
连通所有城市的线路不止一种，但是我们要求的是造价最小的。这本质上是一个图论的问题。各个城市可以认为是图中的各个节点。任意两个城市间的电网连接可看作是图中的边，线路的造价就相当于边的权重。因此，求电网最小造价的问题就相当于在有权图中求最小生成树。最小生成树问题常用的算法主要有 `prim算法` 以及 `Kruskal 算法`。本项目中采用prim算法进行构造。

其原理可表示如下：

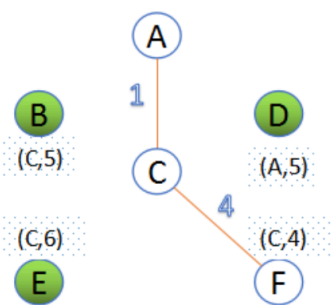


连通网G

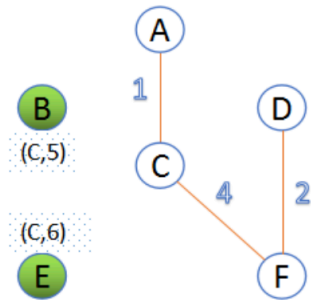
1. 初始 $u=\{A\}$ ,  $v=\{B,C,D,E,F\}$ ; 顶点B下方(A,6), 表示与集合u中A的代价为6作为最小代价边。选择最小的代价边(A,C), 把C并入到集合u中。



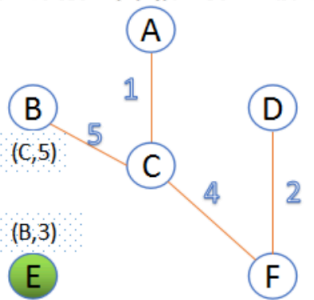
2.  $u=\{A,C\}$ ,  $v=\{B,D,E,F\}$ ; 更新v中顶点与集合u的最小的代价边; 例如: 顶点E之前为(A,∞), 更新为(C,6); 选择最小代价边(C,F), F并入u。



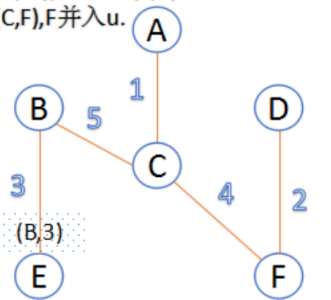
3.  $u=\{A,C,F\}$ ,  $v=\{B,D,E\}$ ; 更新v中顶点与集合u的最小的代价边; 选择最小代价边(F,D), D并入u。



4.  $u=\{A,C,F,D\}$ ,  $v=\{B,E\}$ ; 更新v中顶点与集合u的最小的代价边; 选择最小代价边(C,B), B并入u。



5.  $u=\{A,C,F,D,B\}$ ,  $v=\{E\}$ ; 更新v中顶点与集合u的最小的代价边; 选择最小代价边(B,E), E并入u。



建立一个图结构，存储所有节点以及边的权重。从一个选中节点开始，不断将距离最近的节点拉入集合，同时将对应的边加入最小生成树集合。重复下去，直到收入N个节点。如果最终收入的节点不满N个，则说明图不连通。

## 二. 数据结构实现

### 1. 边类型 (edge)

定义了结构体 `struct edge` 作为图中的边。其成员如下：

```
string x,y;
int weight;
```

`x` , `y` 为 C++ STL 容器的 `string` 类, 代表这条边两端节点的名称。  
`weight` 为 `int` 为类, 代表这条边的权重。

## 构造函数

```
edge(string a,string b,int c):x(a),y(b),weight(c){}
```

构造函数, 传入边两端节点名称以及权重, 建立一个新的边。

## 2.图结构 (graph)

定义了二维数组 `int graph[MAXSIZE][MAXSIZE];` 作为图类型。

## 3.名称-位置映射 (node)

定义了字典类型 `map<string,int> node` 作为图中节点名称到节点存储位置的映射。

实现已知节点名称时快速定位其信息。

## 4.位置-名称映射 (denode)

定义了字典类型 `map<int,string> denode;` 作为图中节点存储位置到名称的映射。实现已知节点位置时快速得到其名称。

## 5.最小生成树 (MST)

定义了结构体 `edge` 的向量 `vector<edge> MST` 作为图最小生成树。存储最小生成树所有的边。

## 6.主要功能函数

```
void getNodes();    // 向图中插入顶点
```

向图中插入节点, 构建节点存储位置和节点名称的两个映射, 同时对节点中的边进行初始化。

```
void addEdge();      // 向图中插入边
```

向图书插入边，无边连接的节点间边为 `MAXWEIGHT`

```
void prim( int start);      // prim 算法求最小生成树
```

prim算法构建最小生成树。

```
void printTree();    // 打印最小生成树
```

输出最小生成树。

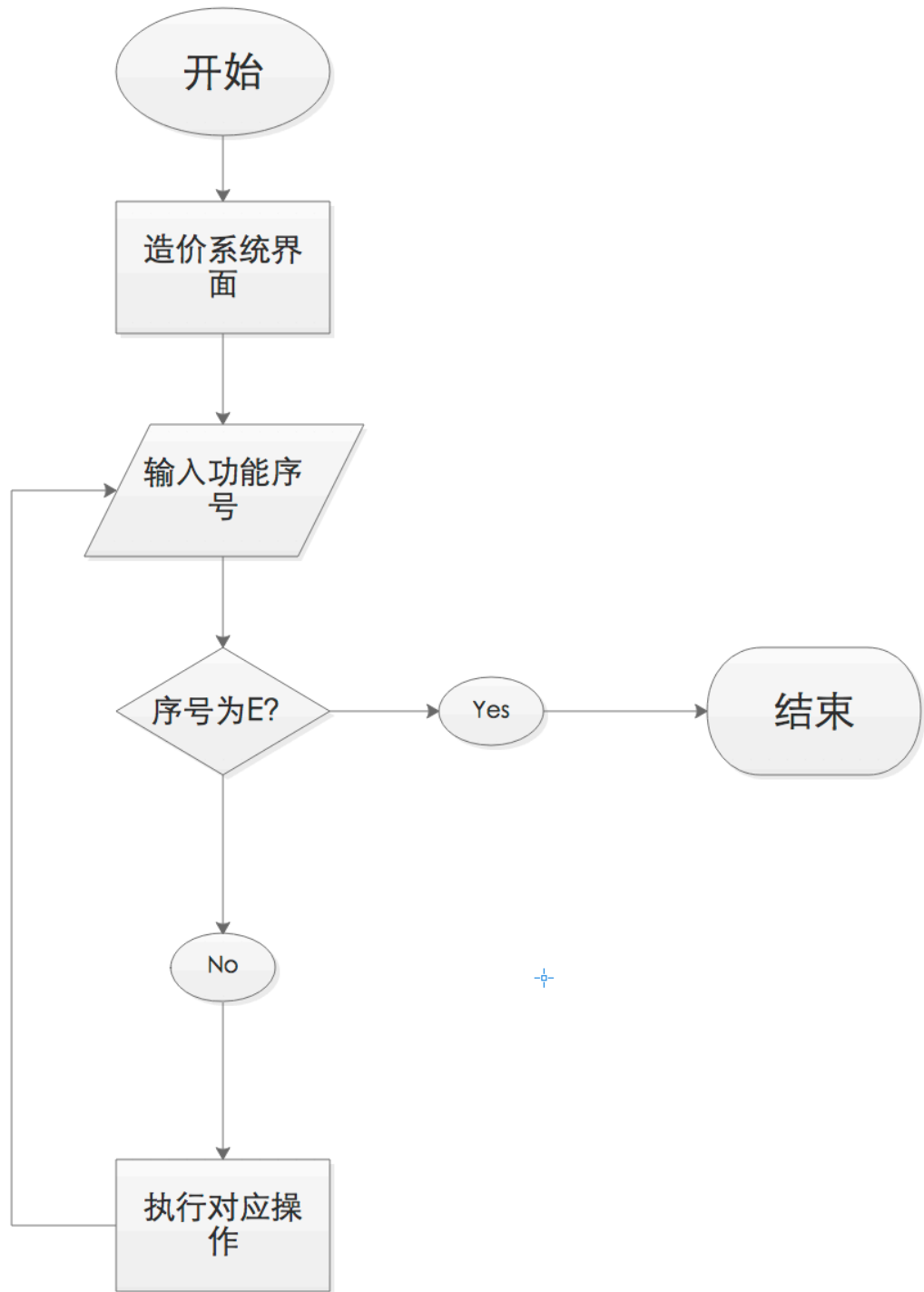
## 7.辅助函数

```
void printMenu(){
    cout<<"**      电网造价模拟系统      **\n";
    cout<<"=====\n";
    cout<<"**      A --- 创建电网顶点      **\n";
    cout<<"**      B --- 添加电网的边      **\n";
    cout<<"**      C --- 构造最小生成树      **\n";
    cout<<"**      D --- 显示最小生成树      **\n";
    cout<<"**      E --- 退出程序      **\n";
    cout<<"=====\n";
}
```

打印主菜单。

## 三.系统实现

### 1.系统执行框架

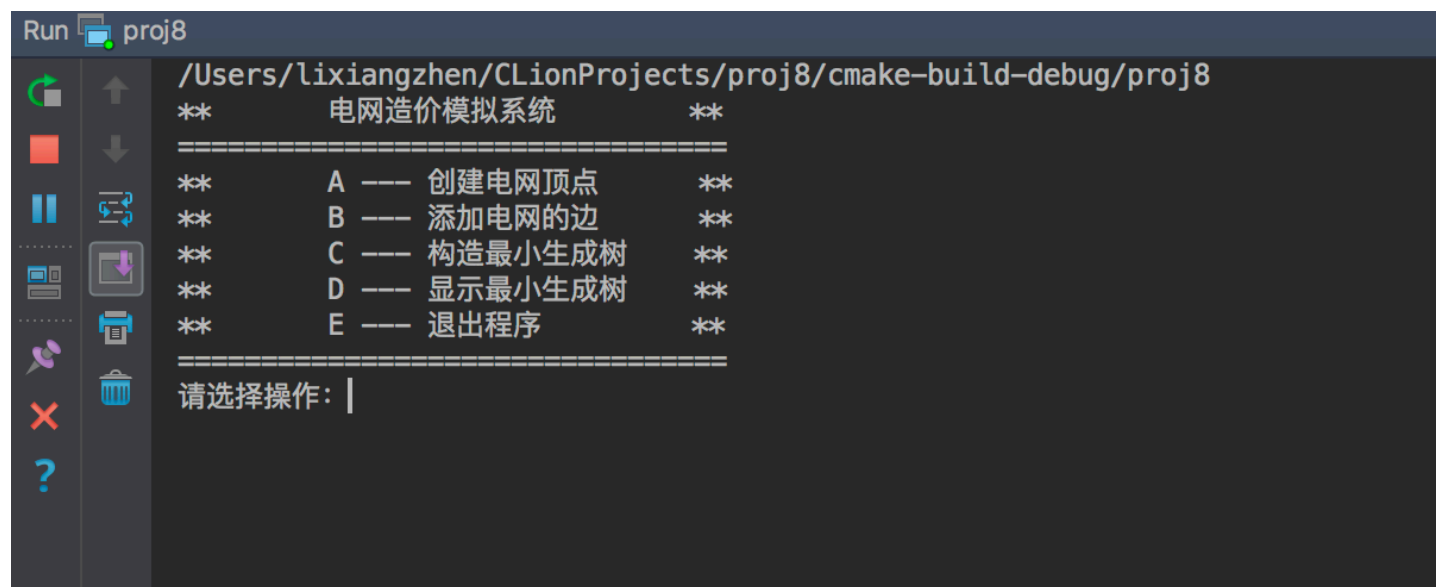


首先调用 `void printMenu()` 函数初始化菜单界面。  
提示用户输入要执行的操作。

核心代码如下：

```
printMenu();    // 输出主菜单
char c;
cout<<"请选择操作："; // 用户输入操作
cin>>c;
```

程序执行情况如下：



```
Run proj8
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8
**      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：|
```

随后输入所选操作，利用 `switch--case` 语句判定不同操作，并跳转到相应位置。

核心代码如下：

```
while(c != 'E'){
    switch (c){        // 根据操作执行相关函数
        case 'A':
            getNodes();
            break;
        case 'B':
            addEdge();
            break;
        case 'C':
            cout<<"请输入起始顶点: ";
            cin>>t;
            if(node.find(t) == node.end())
                cout<<"顶点不存在! \n";
            else
                prim(node[t]);
            break;
        case 'D':
            printTree();
            break;
    }

    cout<<"请选择操作: ";
    cin>>c;
}
```

程序执行情况如下：

```
Run proj8
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8
**      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
请输入顶点个数:
```

## 2.节点插入功能

核心代码如下：



```

/*
 * 向图中插入节点
 * 构建节点存储位置和节点名称的两个映射
 * 同时对节点中的边进行初始化
 * */
void getNodes(){
    cout<<"请输入顶点个数:  ";
    int n;
    cin>>n;
    cout<<"请依次输入各顶点名称: \n";
    string name;
    for(int i = 0;i < n;++i){
        cin>>name;
        node.insert(make_pair(name,i));    // 建立节点名称到节点存储位置的映射
        denode.insert(make_pair(i,name));    // 建立节点存储位置到名称的映射
    }
    for(int i = 0;i < n;++i)
        for(int j = 0;j < n;++j){
            graph[i][j] = MAXWEIGHT;    //初始化所有边
        }
    cout<<endl;
}

```

- 要求用户依次输入图中节点名称
- 同时建立映射关系
- 初始化个边为一个极大值

程序执行情况如下：

```
Run proj8
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8
**      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
请输入顶点个数: 4
请依次输入各顶点名称:
a b c d
请选择操作:
```

### 3.边插入功能

核心代码如下:

```

/*
 * 向图书插入边
 * 无边连接的节点间边为MAXWEIGHT
 * */
void addEdge(){
    int w;
    string n1,n2;
    cout<<"请输入两个顶点及边： ";
    cin>>n1>>n2>>w;
    while(n1 != "?"&&n2 != "?"&&w != 0){          //循环插入所有边
        if(node.find(n1) == node.end()||node.find(n2) == node.end()){          //处理
节点不存在的情况
            cout<<"顶点不存在! \n";
        }
        else{
            graph[node[n1]][node[n2]] = graph[node[n2]][node[n1]] = w;
        }
        cout<<"请输入两个顶点及边： ";
        cin>>n1>>n2>>w;
    }
    cout<<endl;
}
}

```

- 要求用户依次输入个边
- 输入过程判断节点是否存在

程序执行情况如下：

```
Run proj8
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8
**      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
请输入顶点个数: 4
请依次输入各顶点名称:
a b c d

请选择操作: B
请输入两个顶点及边: a b 8
请输入两个顶点及边: b c 7
请输入两个顶点及边: c d 5
请输入两个顶点及边: d a 11
请输入两个顶点及边: a c 18
请输入两个顶点及边: b d 12
请输入两个顶点及边: ? ? 0

请选择操作: |
```

## 4.prim 算法执行功能

核心代码如下:

```
/*
 * prim算法
 * 构建最小生成树
 * */
void prim(int start){
    const int N = node.size(); //节点数量
    int distan[N];             //未选入节点到已选入及节点集的最短距离
    int parent[N];             //节点连向选入节点的父节点
    int count = 1;
    for(int i = 0;i < N;++i){ //初始化辅助数组
```

```

        distan[i] = graph[start][i];
        parent[i] = start;
    }
    parent[start] = -1;        //初始化 0 节点
    distan[start] = 0;

    while(1){
        int v = -1;           //记录距离最近节点
        int mindis = MAXWEIGHT; //记录最近距离

        for(int i = 0;i < N;++i){    //循环遍历，找出最近节点
            if(distan[i] != 0&&distan[i] < mindis){
                v = i;
                mindis = distan[i];
            }
        }
        if(v == -1)            //不存在最近节点，则算法结束
            break;
        ++count;               //记录已选入节点数量
        distan[v] = 0;
        MST.emplace_back(edge(denode[parent[v]],denode[v],graph[parent[v]][v]));
        //将新生成的边插入最小生成树

        for(int i = 0;i < N;++i){    //跟新未选入节点到已选入及节点集的最短距离
            if(distan[i] != 0&&distan[i] > graph[i][v]){
                distan[i] = graph[i][v];
                parent[i] = v;
            }
        }

    }
    if(count < N - 1)           //处理图不连通的情况
        cout<<"不存在最小生成树! ";
    else
        cout<<"生成Prim最小生成树! \n";
}

```

- 利用Prim算法生成最小生成树
- 辅助数组记录每个节点前驱节点
- 辅助数组记录每个节点到入选集合的距离
- 每次将距离最小节点移入

- 找不到这样的节点是程序退出。 \*

程序执行情况如下：

```
Run: proj8 proj8 proj8 proj8
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8
**      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
请输入顶点个数: 4
请依次输入各顶点名称:
a b c d

请选择操作: B
请输入两个顶点及边: a b 8
请输入两个顶点及边: b c 7
请输入两个顶点及边: c d 5
请输入两个顶点及边: d a 11
请输入两个顶点及边: a c 18
请输入两个顶点及边: b d 12
请输入两个顶点及边: ? ? 0

请选择操作: C
请输入起始顶点: A
生成Prim最小生成树!
请选择操作:
```

## 5.显示最小生成树功能

核心代码如下：

```
/*
 * 输出最小生成树
 * */
void printTree(){
    cout<<"最小生成树的顶底及边为： \n\n";
    for(int i = 0;i < MST.size();++i){
        cout<<MST[i].x<<"-<"<<MST[i].weight<<">->"<<MST[i].y<<"    ";    //
    }
    cout<<"\n\n";
}
```

- 格式化输入最小生成树
- 输出节点及其边

程序执行情况如下：

```
请输入两个顶点及边： d a 11
请输入两个顶点及边： a c 18
请输入两个顶点及边： b d 12
请输入两个顶点及边： ? ? 0

请选择操作： C
请输入起始顶点： A
生成Prim最小生成树！
请选择操作： D
最小生成树的顶底及边为：

a-<8>->b    b-<7>->c    c-<5>->d

请选择操作：
```

# 四.测试

## 1.基本功能测试

### 测试用例

节点个数：

节点名称：

边：

```
3 4 70
1 2 1
5 4 50
2 6 50
5 6 60
1 3 70
4 6 60
3 6 80
5 1 100
2 4 60
5 2 80
```

### 创建电网顶点

程序执行情况如下：



```
Run: proj8 proj8 proj8 proj8
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug
**      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
请输入顶点个数: 6
请依次输入各顶点名称:
1 2 3 4 5 6
```

添加电网的边

程序执行情况如下:

```
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8
**      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
请输入顶点个数: 6
请依次输入各顶点名称:
1 2 3 4 5 6

请选择操作: B
请输入两个顶点及边: 3 4 70
请输入两个顶点及边: 1 2 1
请输入两个顶点及边: 5 4 50
请输入两个顶点及边: 2 6 50
请输入两个顶点及边: 5 6 60
请输入两个顶点及边: 1 3 70
请输入两个顶点及边: 4 6 60
请输入两个顶点及边: 3 6 80
请输入两个顶点及边: 5 1 100
请输入两个顶点及边: 2 4 60
请输入两个顶点及边: 5 2 80
请输入两个顶点及边: ? ? 0
```

## 构造最小生成树

程序执行情况如下:

```
请输入两个顶点及边: 5 1 100
请输入两个顶点及边: 2 4 60
请输入两个顶点及边: 5 2 80
请输入两个顶点及边: ? ? 0

请选择操作: C
请输入起始顶点: 3
生成Prim最小生成树!
```

## 显示最小生成树

程序执行情况如下：

```
请选择操作: C
请输入起始顶点: 3
生成Prim最小生成树!
请选择操作: D
最小生成树的顶底及边为:

3-<70>->1    1-<1>->2    2-<50>->6    2-<60>->4    4-<50>->5

请选择操作:
```

## 程序退出

程序执行情况如下：

```
3-<70>->1    1-<1>->2    2-<50>->6    2-<60>->4    4-<50>->5

请选择操作: E

Process finished with exit code 0
```

## 2.边界测试

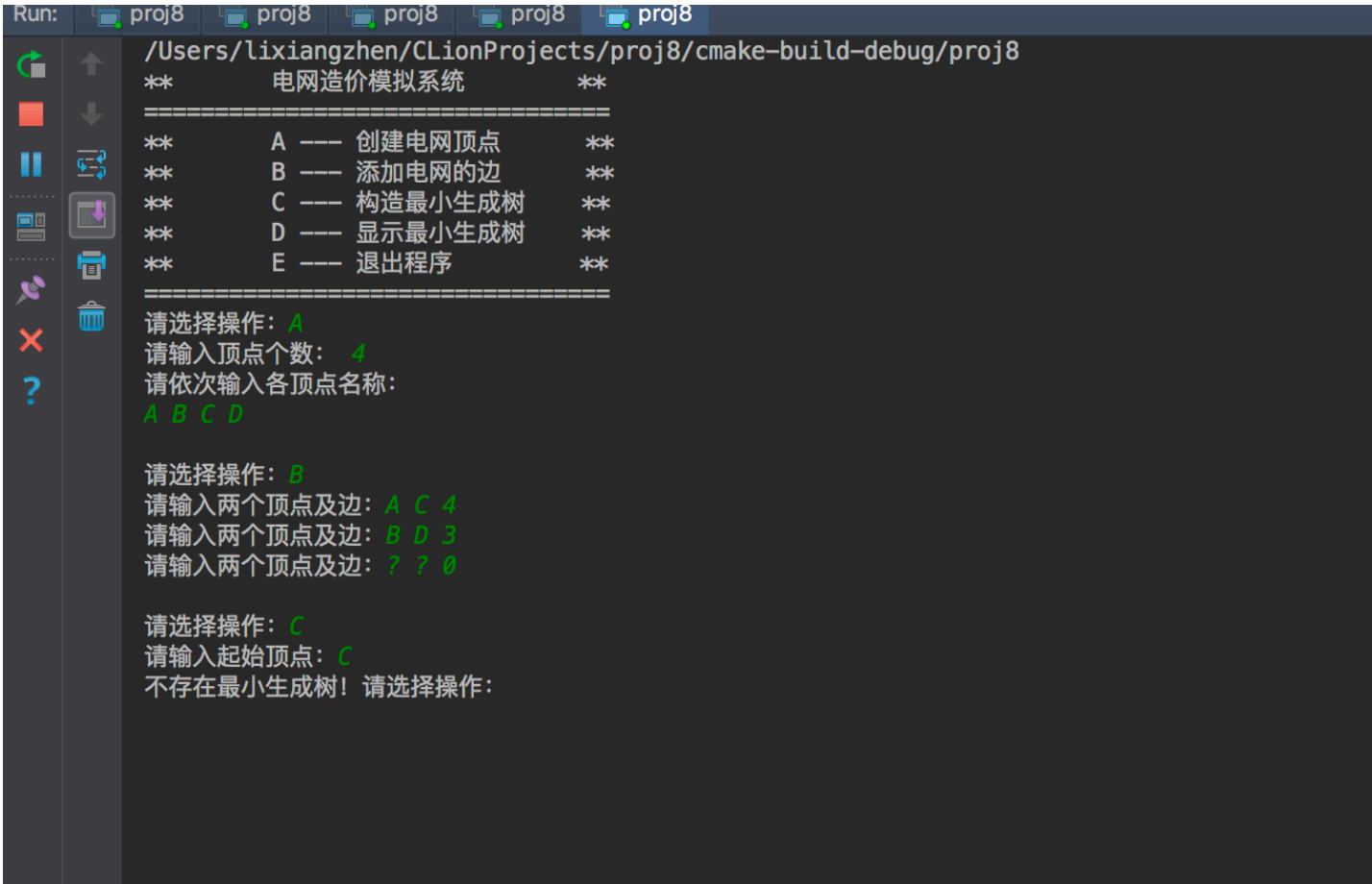
图不连通

测试用例

节点个数: 4  
节点名称: A B C D .  
边:

A C 4  
B D 3

程序执行情况如下:



插入边时顶点不存在

节点个数：

节点名称：

边：

Q W 2

Q S 4

程序执行情况如下：

节点个数：

节点名称：

边：

A B 2

B C 6

A C 11

```
Run: proj8 proj8 proj8 proj8
/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8
**      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
请输入顶点个数: 3
请依次输入各顶点名称:
Q W E

请选择操作: B
请输入两个顶点及边: Q W 2
请输入两个顶点及边: Q S 4
顶点不存在!
请输入两个顶点及边:
```

构建连通图时顶点不存在

程序执行情况如下：

/Users/lixiangzhen/CLionProjects/proj8/cmake-build-debug/proj8

\*\* 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**

请输入顶点个数: **3**

请依次输入各顶点名称:

**A B C**

请选择操作: **B**

请输入两个顶点及边: **A B 2**

请输入两个顶点及边: **B C 6**

请输入两个顶点及边: **A C 11**

请输入两个顶点及边: **? ? 0**

请选择操作: **C**

请输入起始顶点: **E**

顶点不存在!

请选择操作: