实验 4-2: 求通讯网的最小代价生成树

1. 问题描述

输入一个无向铁通讯网图，用Prim和Kruskal算法计算最小生成树并输出。

输入要求: 第一行是两个数n，m(1 < n < 10000，1 < m < 100000)，分别表示顶点数量和边的数量。 接下来的m行每行输入三个数a、b、w；表示顶点a与顶点b之间有代价为w的边相连，顶点编号从1到n。
输出要求: 输出包含一个数，即最小生成树的各边的长度之和

2. 算法的描述

2.1 数据结构的描述

A. Kruskal 使用了边构成的表 `pENode* EList`, 每个 e 标记了它连接的两边和它的权。处理后，使用 Kruscal 进行排序。步骤是，先根据边权对 `Elist` 进行排序后，逐步选取每条边，并确保没有圈形成。

B. Prim 使用了领接表如下所示：

```
struct eNode {
    int weight;
    struct eNode* next;
    struct vNode* connect_to;
};

struct vNode {
    struct eNode* first;
    int no;
};
```

2.2 程序结构的描述

`main.c` 中主要的函数有:

`main()`: 读取边权和连接的点，做出图，并调用算法进行计算 `Kruscal()`: 使用 Kruskal 算法进行最小生成树的计算 `Find()`: 寻找两个点共同的 parent，若两个点有共同的 parent，则不能取此边（并查集）

`prim.c` 读取后直接对其进行 Prim 算法的处理。

3. 调试分析

在写 Kruskal 算法时，我发现时间总是🅺 超限，后发现在 Find() 算法中，寻找共同 parent 时，可以将路上所有点的 parent 都递归地改为真 parent，从而降低搜索时间。

4. 算法的时间分析

Kruskal 的时间复杂度为 O(NLogN) Prim 的时间复杂度为 O(ELog(V))