题目

G 单源最小路

时间限制：2000ms 内存限制：65536kb

通过率：17/75 (22.67%) 正确率：17/244 (6.97%)

题目描述

定义一条路的代价为该条路所经过的边权值的**最大值**。现在给出一个包含有 n 个点 m 条边的无向带权图，结点编号为 $1 \sim n$。请你对于除结点 1 以外的点，分别求出从结点 1 到它的路，使得该条路的代价**最小**。

输入

第一行，两个正整数 n, m（$2 \leq n \leq 2 \times 10^5, 2 \leq m \leq 5 \times 10^5$）。

接下来 m 行，每行三个正整数 u, v, w，表示节点 u 和 v 之间有一条权值为 w 的边（$1 \leq u, v \leq n, 1 \leq w \leq 10^9$）。

输出

输出共 $n-1$ 行。第 i 行包含一个整数，表示从结点 1 到结点 $i+1$ 的路的最小代价。如果无法从结点 1 到达结点 $i+1$，则输出 -1。

思路

迪杰斯特拉变形，把判断条件从max（dis[to],min(dis[from],weight)）修正为min（dis[to],max(dis[from],weight)）;时间复杂度为O（nlogn）

代码实现

```
#include<queue>
#include<iostream>
using namespace std;
#define maxn (int)1e6+10
#define inf 2e9
#define ll long long
#define pr pair<int,int>
priority_queue<pr, vector<pr>, greater<pr>>q;
class edgex {
public:
    int to;
    int next;
    int weight;
};
int head[maxn];
edgex edge[maxn * 2];
int cnt = 0;
void add(int u, int v, int w) {
    edge[++cnt].next = head[u];
    edge[cnt].to = v;
    edge[cnt].weight = w;
```

```
    head[u] = cnt;
}
int dis[maxn];
bool visit[maxn];
int main() {
    int n, m;
    cin >> n >> m;
    int u, v, w;
    for (int i = 1; i <= m; ++i) {
        cin >> u >> v >> w;
        add(u, v, w);
        add(v, u, w);
    }
    for (int i = 2; i <= n; ++i) {
        dis[i] = inf;
    }
    q.push(make_pair(0, 1));
    while (!q.empty()) {
        auto it = q.top();
        q.pop();
        int num = it.second;
        if (visit[num]) {
            continue;
        }
        visit[num] = true;
        for (int i = head[num]; i; i = edge[i].next) {
            int to = edge[i].to;
            int weight = edge[i].weight;
            dis[to] = min(dis[to], max(weight, dis[num]));
            q.push(make_pair(dis[to], to));
        }
    }
    for (int i = 2; i <= n; ++i) {
        if (dis[i] == inf) {
            cout << "-1\n";
        }
        else {
            cout << dis[i] << "\n";
        }
    }
}
```

要注意的

用链式前向星操作存图 前向星数组对应的其实是边的信息

```
int to;//边的起点
int next;//边的终点
int weight;//边长
```