dijkstra+heap

```
#include<bits/stdc++.h>
using namespace std;
const int INF = 0x3f3f3f3f;
const int MAXN = 1000010;
struct qNode
{
    int v;
    int c;
    qNode(int _v = 0, int _c = 0) : v(_v), c(_c) {}
    bool operator < (const qNode &r) const
    {
        return c > r.c;
    }
};
struct Edge
{
    int v;
    int cost;
    Edge(int _v = 0, int _cost = 0) : v(_v), cost(_cost) {}
};
vector<Edge> E[MAXN];
bool vis[MAXN];
int dist[MAXN];     //  最短路距离

void Dijkstra(int n, int start)     //  点的编号从1开始
{
    memset(vis, false, sizeof(vis));
    memset(dist, 0x3f, sizeof(dist));
    priority_queue<qNode> que;

    while (!que.empty())
    {
        que.pop();
    }
    dist[start] = 0;
    que.push(qNode(start, 0));
    qNode tmp;

    while (!que.empty())
    {
        tmp = que.top();
        que.pop();
        int u = tmp.v;
        if (vis[u])
        {
            continue;
        }
        vis[u] = true;
        for (int i = 0; i < E[u].size(); i++)
        {
            int v = E[u][i].v;
```

```
            int cost = E[u][i].cost;
            if (!vis[v] && dist[v] > dist[u] + cost)
            {
                dist[v] = dist[u] + cost;
                que.push(qNode(v, dist[v]));
            }
        }
    }
}

void addEdge(int u, int v, int w)
{
    E[u].push_back(Edge(v, w));
}
```