堆优化Dijkstra

```cpp
#include <bits/stdc++.h>
using namespace std;

const int N = 1e5 + 5, M = 5e5 * 2 + 10;
int n, m, s;
int tot, h[N], dist[N];
bool vis[N];
typedef pair<int, int> PII;
struct edge{
    int to, nxt, w;
}e[M];

void add(int u, int v, int w){
    e[++tot].to = v;
    e[tot].w = w;
    e[tot].nxt = h[u];
    h[u] = tot;
}

void dijkstra(){
    memset(dist, 0x3f, sizeof(dist));
    dist[s] = 0;
    priority_queue<PII, vector<PII>, greater<PII>> heap;
    heap.push({0, s});

    while(!heap.empty()){
        auto t = heap.top();
        heap.pop();
        int dis = t.first, v = t.second;
        if(vis[v]) continue;
        vis[v] = true;
        for(int i = h[v]; i; i = e[i].nxt){
            int j = e[i].to;
            if(dis + e[i].w < dist[j]){
                dist[j] = dis + e[i].w;
                heap.push({dist[j], j});
            }
        }
    }
}

int main(){
    cin >> n >> m >> s;
    for(int i = 0; i < m; i++){
        int u, v, w;
        scanf("%d%d%d", &u, &v, &w);
        add(u, v, w);
    }

    dijkstra();
```

```
    for(int i = 1; i <= n; i++){
        printf("%d ", dist[i]);
    }
    return 0;
}
```