最小传输时延

题目描述：某通信网络中有 N 个网络结点，用 1 到 N 进行标识。网络通过一个有向无环图表示，其中图的边的值表示结点之间的消息传递时延。 现给定相连节点之间的时延列表 times[i]={u,v,w}，其中 u 表示源结点，v 表示目的结点，w 表示 u 和 v 之间的消息传递时延。请计算给定源结点到目的结点的最小传 输时延，如果目的结点不可达，返回-1。 注： 1、N 的取值范围为[1,100]； 2、时延列表 times 的长度不超过 6000，且 1 <= u,v <= N, 0 <= w <= 100； 输入描述：输入的第一行为两个正整数，分别表示网络结点的个数 N，以及时延列表的长度 M，用空格分隔； 接下来的 M 行为两个结点间的时延列表[u v w]； 输入的最后一行为两个正整数 u 和 v，分别表示源结点和目的结点；
输出描述：输出一个整数，表示源结点到目的结点的最小时延。

示例1
输入：3 3
1 2 11
2 3 13
1 3 50
1 3
输出：24
说明：1->3的时延是50，1->2->3时延是11+13=24，所以1到3的最小时延是24；

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

#define mypair pair<int, int>
#define MAXN 1005

vector<mypair> edges[MAXN];
int dis[MAXN];
bool vis[MAXN];

void dijkstra(int u) {
    for (int i = 0; i < MAXN; i++) {
        // vis[i] = false;
        dis[i] = INT32_MAX;
    }
    dis[u] = 0;

    priority_queue<mypair> pq;

    pq.push(make_pair(dis[u], u));

    while (!pq.empty()) {
        mypair tmp = pq.top();
        pq.pop();
        int cur_node = tmp.second, cur_dis = tmp.first;
        // if (vis[cur_node]) continue;
        if (cur_dis != dis[cur_node])  continue;
        // vis[cur_node] = true;
        for (mypair edge : edges[cur_node]) {
            int new_dis = cur_dis + edge.second;
            if (dis[edge.first] > new_dis) {
                pq.push(make_pair(new_dis, edge.first));
                dis[edge.first] = new_dis;
            }
        }
    }


}

int main() {
    int n, m, u, v, w;
    cin >> n >> m;
    for (int i = 0; i < m; i++) {
        cin >> u >> v >> w;
        edges[u].emplace_back(make_pair(v, w));
    }
    cin >> u >> v;

    dijkstra(u);

    if (dis[v] == INT32_MAX)
        cout << -1 << endl;
    else
        cout << dis[v] << endl;

    return 0;
}
// 64 位输出请用 printf("%lld")
```