//最短路

struct edge {

int from, to, cost;

edge() {}

edge(int ffrom, int tto, int ccost)

{

from = ffrom;

to = tto;

cost = ccost;

} };

vector<int> g[maxn];

vector<edge> edges; //整个有向图

int rank1[maxn]; //最短路的改进次数

int dis[maxn]; // dist[i]是源到i的目前最短路长度

bool inque[maxn];//是否走过

void add(int u, int v, int w){

edges.push\_back(edge(u, v, w));

int m = edges.size();

g[u].push\_back(m-1);}

bool spfa(int s, int n){

for(int i = 0; i <= n; i++)

{

dis[i] = inf;

rank1[i] = 0;

inque[i] = false;

}

dis[s] = 0;

rank1[s] = 1;

inque[s] = true;

queue<int> q;

q.push(s);

while( !q.empty())

{

int u = q.front();

inque[u] = false;

q.pop();

for(int i = 0; i < (int)g[u].size(); i++)

{

edge e = edges[ g[u][i] ];

if(dis[e.to] > dis[u] + e.cost)

{

dis[e.to] = dis[u] + e.cost;

if(!inque[e.to])

{

q.push(e.to);

inque[e.to] = true;

rank1[e.to]++;

if( rank1[e.to] >= n) return false;

}

}

}

}

return true;

}

int u, v, w; int s, n, t;

while( scanf("%d %d", &N, &M) !=EOF)

{

for (int i = 0; i <= N; i++) g[i].clear();

edges.clear();

for(int i = 1; i <= M; i++) {

scanf("%d %d %d", &u, &v, &w);

add(u, v, w); add(v, u, w)}

scanf("%d %d", &s, &t);

//init(N) spfa(s, N) ;

if(dis[t] != inf)