#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#include<stack>

#define MAX 505

using namespace std;

struct store {

int dist=0;

int value=0;

};

struct found\_path

{

vector<int>citys;

int dis = 0;

int total\_cost;

};

vector<found\_path>all\_path;

vector<int>passed;

store map[MAX][MAX];

int vst[MAX] = { 0 };

int citynum, relation, current, des;

int comp(found\_path a, found\_path b)

{

if (a.dis < b.dis)

return 1;

else if (a.dis == b.dis)

{

if (a.total\_cost < b.total\_cost)

return 1;

else

return 0;

}

else

return 0;

}

void dfs(int current\_pos,int des, int total\_value,int total\_dis)

{

if (all\_path.size())

if (total\_dis > all\_path.front().dis)

return;

if (current\_pos == des)

{

vector<int>temp(passed);

found\_path result;

result.citys = temp;

result.total\_cost = total\_value;

result.dis = total\_dis;

all\_path.push\_back(result);

return;

}

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

{

if (map[current\_pos][i].dist)

{

if(!vst[i])

{

vst[i] = 1;

passed.push\_back(i);

dfs(i, des, total\_value + map[current\_pos][i].value, total\_dis + map[current\_pos][i].dist);

passed.pop\_back();

vst[i] = 0;

}

}

}

return;

}

void output()

{

found\_path temp = all\_path.front();

cout << temp.citys[0];

for (int i = 1; i < temp.citys.size(); i++)

cout << " " << temp.citys[i];

cout << " " << temp.dis << " " << temp.total\_cost;

}

int main()

{

cin >> citynum >> relation >> current >> des;

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

{

int there, that,dist,cost;

cin >> there >> that;

cin >> dist >> cost;

map[there][that].dist = dist;

map[that][there].dist = dist;

map[there][that].value = cost;

map[that][there].value = cost;

}

//input

passed.push\_back(current);

vst[current] = 1;

dfs(current,des,0,0);

sort(all\_path.begin(), all\_path.end(), comp);

//process

output();

return 0;

}

//以上是非最短路算法深搜加上玄学剪枝的结果。