#include<bits/stdc++.h>

using namespace std;

vector<vector<pair<int,int> > > graph;

void shortest\_path\_dijkstra(int src, vector<int> &distance){

priority\_queue<pair<int,int>, vector<pair<int,int> >, greater<pair<int,int> > > pq;

pq.push(make\_pair(0,src));

distance[src] = 0;

while(!pq.empty()){

int dist = pq.top().first;

int node = pq.top().second;

pq.pop();

if(distance[node]!=INT\_MAX){

for(auto x: graph[node]){

if(distance[x.first]>x.second+dist){

distance[x.first] = x.second+dist;

pq.push(make\_pair(distance[x.first], x.first));

}

}

}

}

}

int main(){

int n;

int e;

cin>>n>>e;

graph.resize(n);

for(int i=0;i<e;i++){

int x,y,w;

cin>>x>>y>>w;

graph[x].push\_back(make\_pair(y,w));

graph[y].push\_back(make\_pair(x,w));

}

int src;

cin>>src;

vector<int> distance(n,INT\_MAX);

shortest\_path\_dijkstra(src,distance);

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

if(distance[i]==INT\_MAX){

cout<<i<<" inf"<<endl;

}

else{

cout<<i<<" "<<distance[i]<<endl;

}

}

return 0;

}