```
#include<bits/stdc++.h>
using namespace std;
typedef pair<int, int> P;
const int inf = numeric_limits<int>::max()/2;

int v,e;
vector< vector<P> >graph;
vector<int> dist;

void dijkstra(int s){
    priority_queue <P, vector<P>, greater<P> > que;
    dist.clear();
    dist.resize(v);
    fill(dist.begin(),dist.end(),inf);
    que.push(P(0,s));
    while(!que.empty()){
      int cost = que.top().first;
      int node = que.top().second;
      que.pop();
      if(dist[node] > cost)dist[node] = cost;
      for(int i = 0;i < (int)graph[node].size();++i){
        int ncost = graph[node][i].first;
        int nnode = graph[node][i].second;
        if(dist[node] + ncost < dist[nnode])
          que.push(P(dist[node] + ncost,nnode));
      }
    }
}

int main(void){
    int r;
    cin  >> v >> e >> r;
    graph = vector< vector<P> >(v);
    for(int i = 0;i< e;++i){
        int s,t,d;
        cin >> s >> t >> d;
        graph[s].push_back(P(d,t));
    }
    dijkstra(r);
    for(int i = 0;i < v;++i){
        if(dist[i] == inf)cout << "INF" << endl;
        else cout << dist[i] << endl;
    }
    return 0;
}
```