#include <iostream>

#include <vector>

#include <queue>

#include <climits>

#include <set>

using namespace std;

struct Edge {

int to, weight;

bool operator>(const Edge& other) const {

return weight > other.weight;

}

};

void prim(int V, vector<vector<Edge>>& graph) {

priority\_queue<Edge, vector<Edge>, greater<Edge>> pq;

vector<bool> inMST(V, false);

vector<int> minWeight(V, INT\_MAX);

vector<int> parent(V, -1);

minWeight[0] = 0;

pq.push({0, 0}); // Start from vertex 0

while (!pq.empty()) {

int u = pq.top().to;

pq.pop();

if (inMST[u]) continue;

inMST[u] = true;

for (auto& edge : graph[u]) {

int v = edge.to;

int weight = edge.weight;

if (!inMST[v] && minWeight[v] > weight) {

minWeight[v] = weight;

pq.push({v, weight});

parent[v] = u;

}

}

graph[u].clear();

}

for (int i = 1; i < V; ++i)

cout << parent[i] << " - " << i << endl;

}

int main() {

int V, E;

cin >> V >> E;

vector<vector<Edge>> graph(V);

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

int u, v, weight;

cin >> u >> v >> weight;

graph[u].push\_back({v, weight});

graph[v].push\_back({u, weight});

}

prim(V, graph);

return 0;

}