#include <iostream>

#include <vector>

using namespace std;

struct Edge {

// arista (u -> v)

int u;

int v;

int weight;

string name;

};

int main(int, char\*\*){

// // 1. Adjacent list

// vector<vector<Edge>> adjList;

// // Space = n + m, where m = # edges

// // TIme to find an edge O(n)

// int n = 5;

// adjList.resize(n);

// // Add edge (0, 1) (1, 0)

// // (0,1) -> weight = 7

// adjList[0].push\_back({1, 7, "arista\_1"});

// adjList[1].push\_back({0, 7, "arista\_1"});

// // Add edge (0, 2) (2, 0)

// // (0,2) -> weight = 5

// adjList[0].push\_back({2, 5, "arista\_2"});

// adjList[2].push\_back({0, 5, "arista\_2"});

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

// cout << "node = " << i << "\n";

// cout << "vecinos { ";

// for (auto edge : adjList[i]) {

// cout << "( "<< edge.v << ", " << edge.weight << "," << edge.name << ") ";

// }

// cout << "}\n";

// }

// // 2. Matrix

// n = 5;

// vector<vector<int>> matrix;

// matrix.assign(n, vector<int>(n, -1));

// // Space = n ^ 2

// // Time for find an edge O(1)

// // -1 es un peso invalido

// // Add edge (0, 1) (1, 0)

// // (0,1) -> weight = 7

// matrix[0][1] = 7;

// matrix[1][0] = 7;

// // Add edge (0, 2) (2, 0)

// // (0,2) -> weight = 5

// matrix[0][2] = 5;

// matrix[2][0] = 5;

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

// for (int j = 0; j < n; j++) {

// std::cout << matrix[i][j] << " ";

// }

// std::cout << "\n";

// }

// 3. Edge List

vector<Edge> edgeList;

// add edge (0,1) with weight 7 y name = "first"

// O(m)

edgeList.push\_back({0, 1, 7, "first"});

edgeList.push\_back({1, 0, 7, "first"});

}