Marathon of Parallel Programming - 2021
Problem: Graph Isomorphism Detection

Wilton Jaciel Loch, Guilherme Piêgas Koslovski

October 26, 2021

Problem definition

Given two graphs $G(V, E)$ and $H(V, E)$, where V denotes the vertices and E the edges, the problem of finding isomorphic graphs consists of determining a bijection function $f : V(G) \rightarrow V(H)$ that maps the vertices of both graphs while preserving the adjacency condition. In other words, for the isomorphism property to exist, if any two vertices $u, v \in G$ are adjacent then the corresponding vertices $f(u), f(v) \in H$ must also be adjacent. In this sense, the proposed algorithm detects all the possible isomorphic relations between two undirected graphs through extensive permutations, comparing all possible vertex correspondences between G and H.

Input

An input represents only a test case. Initially, the graph G is informed followed by H. The first line contains an integer V_g $(0 < V_g < 100)$ representing the number of vertices in G, while the second line informs an integer E_g $(0 \leq E_g < (V_g * (V_g - 1))/2)$ denoting the number of edges in G. Following, E_g edges are sequentially informed, each denoted by two integers $u, v \in V_g$. The next lines repeat the same rationale for informing the graph H: an integer V_h $(0 < V_h < 100)$ representing the number of vertices in H is followed by an integer E_h $(0 \leq E_h < (V_h * (V_h - 1))/2)$ denoting the number of edges in H. Finally, E_h edges are sequentially informed, each denoted by two integers $u, v \in V_h$.

The input must be read form the standard input.

Output

The output contains N lines. If there is no possible isomorphic relation between G and H, only the message *The graphs are not isomorphic* is printed to the output. Otherwise, a number of lines equal to the amount of possible isomorphisms is printed, each containing a stream of integers $v \in V_h$ which represents

a vertex correspondence between G and H, such that $f(u) = v$, with $u \in V_g$ equal to the position of each v on the stream. Thus, the order in which vertices are printed in each of the solution lines defines the solution itself.

The output must be written to the standard output.

Example

Input example 1	Output example 1	Input example 2	Output example 2
4 6 0 2 1 1 1 3 2 2 2 3 3 3 4 6 0 0 0 1 0 2 1 1 1 3 3 3	2 3 0 1	4 6 0 3 1 1 1 3 2 2 2 3 3 3 4 6 0 0 0 1 0 2 1 1 1 3 3 3	The graphs are not isomorphic