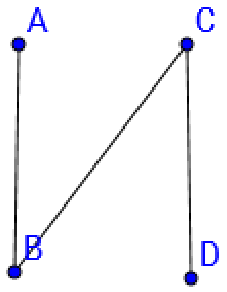
ISOMORPH

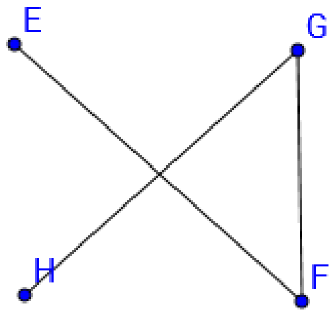
1. **1. Introduction**

   Definition : An **isomorphism** from a simple graph G to a simple graph H is a bijection f:V(G) -> V(H) such that uv belongs to E(G) if and only if f(u)f(v) belongs to E(H) . We say “G is **isomorphic to** H”, if there is an isomorphism from G to H.

   Example : The following two graphs are isomorphic.



G_1



G_2

Our objective in this problem is to find isomorphic graphs.

Input

Input is read from STDIN. Each test case consists of graphs all with same number of vertices. Your task in this assignment will be to divide the given set of graphs into classes within which graphs are pairwise isomorphic.

k v [k is the number of graphs (<=20) in the test case and v is the number of vertices (<=20) in each of these graphs]

e_1 [number of edges in first graph, call this graph ‘1’]

u_1 w_1

u_2 w_2

...

u_{e1} w_{e1}

e_2 [number of edges in second graph, call this graph ‘2’]

u_1 w_1

u_2 w_2

...

u_{e2} w_{e2}

e_3

...

...

...

e_k [number of edges in the last graph, call this graph ‘k’]

u_1 w_1

u_2 w_2

...

u_{ek} w_{ek}

[Note: all n graphs themselves are named ‘1’, ‘2’...’n’ in the order in which they appear in the input. Hence the graph with e_1 edges is called ‘1’ and so on]

[Note: all graphs are simple & undirected; vertices are labelled with numbers, not alphabets.]

Output

An isomorphic class is simply a list of graphs which belong to it. And Output is written to STDOUT.
The output should follow the below format:
1. Output is a listing of all isomorphic classes, one on each line, sorted in ascending order by the first
graph of the class.
2. Within an isomorphic class, the members are printed in ascending order.
(Thus, the class containing graph ‘1’ is always on line 1.)

Remarks

Problem statement is as described in the assignment pdf. Input output format is also descibed there.

More test cases will be added soon.

Do let us know if you find any errors/hit time limits/etc.

REMEMBER: This is an individual assignment.