Offline Problem | CSE 208 | MST

Instructions:

(1) * Please DO NOT COPY solutions from anywhere (your friends, seniors, internet etc.). Any form of plagiarism (irrespective of source or destination), will result in getting -100% marks in the online/offline.

(2) Deadline: 4th December 11:59 PM

(3) Rename all the problem solutions according to your rule. If your rule is 2005XXX, then create a folder named 2005XXX. Afterward, rename problem 1 as 2005XXX_problem1.cpp, and similarly, rename the others. Next, move all the solutions inside the folder. Create a zip file of that folder. Lastly, submit the zip file.

(4) You get 10 marks for each right answer. A viva will also be conducted. If the teacher finds that you don't know how to implement it, you'll get a score of 0.

Offline Problem 1:

Find all possible MST of a given graph.

Input

The first line contains two integers n and m — the number of the graph's vertices and edges, correspondingly. Then follow m lines, each of them contains three integers — the description of the graph's edges as "a_i b_i w_i" where a_i and b_i are the numbers of vertices connected by the i-th edge, w_i is the edge's weight. It is guaranteed that the graph is connected and doesn't contain loops or multiple edges.

Output

Print All the edges for each possible MST.

Examples

Input	Output
5 10 0 1 1 2 4 1 1 2 2 0 4 2 3 4 3 0 3 5 1 4 6 2 3 8 0 2 9 1 3 9	MSTs: 1 : [[0, 1, 1], [2, 4, 1], [1, 2, 2], [3, 4, 3]] 2 : [[0, 1, 1], [2, 4, 1], [0, 4, 2], [3, 4, 3]]

1 3 9	
7 12 3 4 1 1 5 1 1 4 1 0 6 4 2 3 8 4 5 8 2 6 10 0 5 10 3 6 10 1 6 10 1 3 12 3 5 12	**MSTs:** **1 : [[3, 4, 1], [1, 5, 1], [1, 4, 1], [0, 6, 4], [2, 3, 8], [2, 6, 10]]** **2 : [[3, 4, 1], [1, 5, 1], [1, 4, 1], [0, 6, 4], [2, 3, 8], [0, 5, 10]]** **3 : [[3, 4, 1], [1, 5, 1], [1, 4, 1], [0, 6, 4], [2, 3, 8], [3, 6, 10]]** **4 : [[3, 4, 1], [1, 5, 1], [1, 4, 1], [0, 6, 4], [2, 3, 8], [1, 6, 10]]**

Offline Problem 2:

You are given a connected weighted undirected graph without any loops and multiple edges.Your task is to determine the following for each edge of the given graph: whether it is either included in all possible MST(s), or included at least in one MST, or not included in any MST.

Input

The first line contains two integers n and m — the number of the graph's vertices and edges, correspondingly. Then follow m lines, each of them contains three integers — the description of the graph's edges as "a_i b_i w_i" where a_i and b_i are the numbers of vertices connected by the i-th edge, w_i is the edge's weight. It is guaranteed that the graph is connected and doesn't contain loops or multiple edges.

Output

Print m lines — the answers for all edges. If the i-th edge is included in any MST, print "`any`"; if the i-th edge is included at least in one MST, print "`at least one`"; if the i-th edge isn't included in any MST, print "`none`". Print the answers for the edges in the order in which the edges are specified in the input.

Examples

Input	Output
4 5 1 2 101 1 3 100 2 3 2 2 4 2 3 4 1	none any at least one at least one any
3 3 1 2 1 2 3 1 1 3 2	any any none

Offline Problem 3:

Given a weighted undirected connected graph with n vertices numbered from 0 to n - 1, and an array of edges where edges[i] = [ai, bi, weighti] represents a bidirectional and weighted edge between nodes ai and bi. A minimum spanning tree (MST) is a subset of the graph's edges that connects all vertices without cycles and with the minimum possible total edge weight.

Find all the critical and pseudo-critical edges in the given graph's minimum spanning tree (MST). An MST edge whose deletion from the graph would cause the MST weight to increase is called a critical edge. On the other hand, a pseudo-critical edge is that which can appear in some MSTs but not all.

Input

The first line contains two integers n and m — the number of the graph's vertices and edges, correspondingly. Then follow m lines, each of them contains three integers — the description of the graph's edges as "a_i b_i w_i" where a_i and b_i are the numbers of vertices connected by the i-th edge, w_i is the edge's weight. It is guaranteed that the graph is connected and doesn't contain loops or multiple edges.

Output

Print m lines — all the critical and pseudo-critical edges

Examples

Input	**Output**
5 7 0 1 1 1 2 1 2 3 2 0 3 2 0 4 3 3 4 3 1 4 6	Critical edges: [0,1], Pseudo critical edges: [2,3,4,5] **#NB: Here edge 0 means edge (0, 1 ,1)** **Edge 1 means (1,2,1) look inputs**