Online

Section B2

Time: 45 Minutes

In Graphland, there are many vertices and **undirected edges** between these vertices with specific costs. But there is also a weird constraint in Graphland, no matter what your source or destination is, you must visit the holy edge (*u, v*). Not going through this edge is a serious crime in Graphland.

Given a source and destination node, you are to output the path with the lowest cost. Your recommended path should have the holy edge (*u, v*) as an intermediate edge. If such a path doesn't exist, you must report this. Use Floyd-Warshall algorithm to solve the problem.

Input / Output

The first line of input consists of 2 integers N and M, the no. of input vertices and edges respectively. The next M lines contain 3 integers u, v and w indicating there is an edge from vertex u to v of cost w.

After these M lines, the immediate next line will contain two integers U and V, the two endpoints of the holy edge. You may assume this edge will always exist (i.e. given as input).

The next lines will contain two vertex numbers X and Y respectively. You are to give output a path of the lowest cost from X to Y with the holy edge (u, v) present as an intermediate edge.

The program terminates when -1 -1 is given as input for X and Y respectively.

Sample Input	Sample Output

8 8
1 6 5
2 5 8
2 6 7
4 6 8
5 4 2
5 7 7
7 6 7
8 4 8
2 6
5 3
1 8
8 1
3 4
2 6
8 3
7 5
5 7
-1 -1
	No path from 5 to 3 through 2 -- 6
	Shortest Path Weight: 30
	Path: 1 --> 6(5) --> 2(7) --> 5(8) --> 4(2) --> 8(8)
	Shortest Path Weight: 30
	Path: 8 --> 4(8) --> 5(2) --> 2(8) --> 6(7) --> 1(5)
	No path from 3 to 4 through 2 -- 6
	Shortest Path Weight: 7
	Path: 2 --> 6(7)
	No path from 8 to 3 through 2 -- 6
	Shortest Path Weight: 22
	Path: 7 --> 6(7) --> 2(7) --> 5(8)
	Shortest Path Weight: 22
	Path: 5 --> 2(8) --> 6(7) --> 7(7)