A. Shortest Path

Given an undirected weighted graph find its all pair shortest path cost. The graph may have multiple edges between two vertices.

Input
First Line: N (O<N<=100). number of nodes.
Second line: M (0<N<=10000), number of edges.
Next M lines, each: U V (0<=U, V<N, 0<W<=1000), defines an edge between U and V with weight W.

Output
All pair shortest path in increasing order of nodes. In case of no path, print “infinity”. See sample for clarification

Sample

Input	Output
6 5 1 3 2 1 2 5 0 1 3 3 2 1 2 5 10	0 to 1: 3 0 to 2: 6 0 to 3: 5 0 to 4: infinity 0 to 5: 16 1 to 2: 3 1 to 3: 2 1 to 4: infinity 1 to 5: 13 2 to 3: 1 2 to 4: infinity 2 to 5: 10 3 to 4: infinity 3 to 5: 11 4 to 5: infinity

B.Shortest Path II

Suppose there is a connected undirected weighted graph G_1 with vertices v_1, v_2, …, v_n. Now, you will create another graph G_2 by adding a vertex from G1 along with the edges adjacent to it in each step. Your task is to find the sum of the all pair shortest path costs after each step. The vertices will be added in increasing order.

Input
First Line: N (O<N<=100). number of nodes.
Second line: M (0<N<=10000), number of edges.
Next M lines, each: U V (0<=U, V<N, 0<W<=1000), defines an edge between U and V with weight W.

Output
Sum of the all pair shortest path costs after each step. See sample for clarification

Sample

Input	Output
4 4 0 2 1 0 1 2 1 2 4 1 3 2	0 2 6 17

In the first step, we add vertex 0 to G_2 only. So, the sum of the all pair shortest path cost in G_2 is currently 0.

In the second step, we add vertex 1 to G_2 and the edge between 0 and 1 with weight 2. The sum of the all pair shortest path cost in G_2 is currently 2.

In the third step, we add vertex 2 to G_2 and the edges (0, 2) and (1, 2) with weights 1 and 4. The sum of the all pair shortest path cost in G_2 becomes 6.

C. Shortest Path III

Given a directed weighted graph find its all pair shortest path cost. The graph may have multiple edges between two vertices.

Input
First Line: N (O<N<=500). number of nodes.
Second line: M (0<N<=1000), number of edges.
Next M lines, each: U V (0<=U, V<N, -1000<=W<=1000), defines an edge from U to V with weight W.

Output
All pair shortest path in increasing order of nodes. In case of no path, print “infinity”. In case of negative weight cycle, print “not possible”. See sample for clarification

Sample

Input	Output
4 4 0 2 1 0 1 2 1 2 4 1 3 2	0 to 1: 2 0 to 2: 1 0 to 3: 4 1 to 0: infinity 1 to 2: 4 1 to 3: 2 2 to 0: infinity 2 to 1: infinity 2 to 3: infinity 3 to 0: infinity 3 to 1: infinity 3 to 2: infinity

D. Shortest Path IV

Suppose there are n apartments in a city connected through roads. You have to deliver a product in a set of apartments maintaining a certain order. Find the minimum distance you must have to travel. Assume that, you are already in the apartment you need to deliver first.

Input
First Line: N (0<N<=100). number of apartments.
Second line: M (0<N<=10000), number of roads.
Next M lines, each: U V (0<=U, V<N, 0<W<=1000), defines an road between apartment U and V with distance W.
Next line: R (0<=R<N), the number of apartments to deliver the product.
Next R lines: A (0<=A<N), the apartment number according to the delivery order.

Output
Minimum distance you must have to travel. In case of no path, print “infinity”. See sample for clarification

Sample

Input	Output
4 4 0 2 1 0 1 2 1 2 4 1 3 2 3 1 2 3	8