**Problem A**

Given an undirected weighted graph find its minimum spanning tree. The graph is guaranteed to be connected.

**Input**

First line: N ( 0<N<=100000), number of nodes.

Second line: M ( 0<N<=300000), number of edges.

Next M lines, each: U V W(0<=U, V<N), defines an edge between U and V with weight W.

**Output**

Weight of the MST. See sample for clarification

|  |  |
| --- | --- |
| 6  6  1 3 2  1 2 5  0 1 3  3 2 1  3 4 50  2 5 10 | 66 |

Explanation:

Edges in MST are: 0-1 (3), 1-3 (2), 3-2 (1), 3-4 (50) and 2-5 (10). So the result is 3 + 2 + 1 + 50 + 10 = 66.

**Problem B**

Given an undirected weighted graph find shortest path of each node from the node 0.

**Input**

First line: N ( 0&lt;N&lt;=100000), number of nodes.

Second line: M ( 0&lt;N&lt;=300000), number of edges.

Next M lines, each: U V W(0&lt;=U, V&lt;N), defines an edge between U and V with weight W.

**Output**

Shortest path of each node. If any node is not reachable, print Infinity. See sample for clarification

|  |  |
| --- | --- |
| 7  6  1 3 2  1 2 5  0 1 3  3 2 1  3 4 50  2 5 10 | 0: 0  1: 3  2: 6  3: 5  4: 55  5: 16  6: Infinity |

**Problem C**

Given an undirected weighted graph print the shortest path of node 1 from the node 0. If there are multiple shortest paths, print the lexicographically smallest one.

**Input**

First line: N ( 0&lt;N&lt;=100000), number of nodes.

Second line: M ( 0&lt;N&lt;=300000), number of edges.

Next M lines, each: U V W(0&lt;=U, V&lt;N), defines an edge between U and V with weight W.

**Output**

Shortest path of node 1. If node 1 is not reachable, print “Not Reachable”. See sample for clarification

|  |  |
| --- | --- |
| 7  6  4 3 2  4 2 5  0 4 3  3 2 1  3 6 50  2 1 10 | 0  4  3  2  1 |

**Problem D**

You are given an connected undirected weighted graph. For this problem, cost of a path will be the highest weight of the edge traveled during the path.

You want to visit all the nodes of this graph. But you want to do it in a path, which costs minimum. You can start from any of the nodes in the graph. An edge can be visited as many time as you wish.

**Input**

First line: N ( 0&lt;N&lt;=100000), number of nodes.

Second line: M ( 0&lt;N&lt;=300000), number of edges.

Next M lines, each: U V W(0&lt;=U, V&lt;N), defines an edge between U and V with weight W.

**Output**

Minimum cost of such paths. See sample for clarification

|  |  |
| --- | --- |
| 3  3  1 2 5  0 1 3  2 0 4 | 4 |

Explanation:

There are three ways to do this:

0, 1, 2. Edge weights are 3 and 5. So the cost of this path is 5.

0, 2, 1. Edge weights are 4 and 5. So the cost of this path is 5.

1, 0, 2. Edge weights are 3 and 4. So the cost of this path is 4.

**Problem E**

Given an undirected weighted graph find its second best minimum spanning tree. The graph is guaranteed to be connected.

The second best mst, is the the spanning tree which has minimum cost, but differs with the best mst by at least a single edge.

**Input**

First line: N ( 0&lt;N&lt;=1000), number of nodes.

Second line: M ( 0&lt;N&lt;=100000), number of edges.

Next M lines, each: U V W(0&lt;=U, V&lt;N), defines an edge between U and V with weight W.

**Output**

Weight of the MST. If there is no second best MST, print “No Second Best MST”. See sample for clarification

|  |  |
| --- | --- |
| 6  6  1 3 2  1 2 5  0 1 3  3 2 1  3 4 50  2 5 10 | 69 |