3 Moon Walk

You are given an undirected graph having n vertices, numbered 1 to n, and m edges. Each edge is associated with an integer weight, which is to be interpreted as a binary bit vector. That is, the decimal integer 147_{10} is equivalent to the binary bit vector 10010011_2.

A *walk* in this graph is any path that starts at vertex 1 and ends at vertex n. A walk is allowed to visit the same vertex and same edge multiple times. Our objective is to compute a minimum-weight path from 1 to n, but as our graph resides on the lunar surface the notion of "weight" is very unusual up there.

Rather than summing up weights along a path, we instead take the *exclusive-or* (xor) of the edge weights. For example, if a walk travels along three edges with decimal weights 26_{10} $(= 11010_2)$, 18_{10} $(= 10010_2)$, and 5_{10} $(= 00101_2)$ the total weight of the walk is 13_{10} $(= 11010_2 \oplus 10010_2 \oplus 00101_2)$. Your objective is to find the walk of minimum *xor-weight* from vertex 1 to vertex n.

You may assume that $1 \leq n \leq 10^5$, $1 \leq m \leq 10^5$, and each weight (in decimal) ranges from 0 to 10^9 (which is at most 30 binary bits).

Input and output files can be found at: `http://challengebox.cs.umd.edu/2019/MoonWalk`

Input:

The first line contains two space-separated integers n and m, the number of vertices and the number of edges, respectively. Each of the next m lines contains three space-separated positive integers a, b, and w, where $1 \leq a, b \leq n$ $(a \neq b)$ meaning that there is an edge between vertices a and b with weight w. The weight is represented in decimal. There are no loops or multiple edges in the given graph, and also there is at least one walk from vertex 1 to vertex n.

Output:

A single integer showing the minimum xor-weight of any walk from vertex 1 to vertex n.

Example:

Input:	Output:
4 3 1 2 26 2 3 18 3 4 5	13