Problem A
Connecting Cities with the Minimum *Product* of Costs
Time limit: 1 second

The ACM kingdom has n cities, numbered from 0 to $n-1$. An (unordered) pair $(i, j) \in \{0, 1, \ldots, n-1\}^2$ of distinct cities is said to be linkable if we are able to build a road linking i and j. It is guaranteed that if we build a road between i and j for all linkable pairs $(i, j) \in \{0, 1, \ldots, n-1\}^2$, then any two distinct cities will be reachable (via one or more roads) from each other. For every linkable pair $(i, j) \in \{0, 1, \ldots, n-1\}^2$, denote by $c_{i,j} \geq 0$ the cost of building a road between i and j. Mr. Smart wants to build roads with the minimum *product* of costs subject to every city being reachable from every other city. Please help him.

Technical Specification

1. There are at most 10 test cases.
2. $2 \leq n \leq 500$.
3. There are at most 25000 linkable pairs.
4. For every linkable pair $(i, j) \in \{0, 1, \ldots, n-1\}^2$, $c_{i,j} \in \{0, 1, \ldots, 999\}$.

Input Format

Each test case begins with n and then the number of linkable pairs. Every linkable pair $(i, j) \in \{0, 1, \ldots, n-1\}^2$ is specified by i, j and $c_{i,j}$, in that order. Furthermore, any two consecutive integers are separated by whitespace character(s). The last test case is "0 0", which shall not be processed.

Output Format

Do the following for each test case: Denoting by C the minimum product of costs subject to every city being reachable from every other city, please output the remainder of the division of C by 65537, i.e., C mod 65537.

Sample Input

```
4 5
0 2 996
1 0 3
1 2 4
2 3 800
1 3 998
3 3
```

```
0 2 500
0 1 0
1 2 600
5 7
0 1 100
0 2 999
1 2 100
1 3 10
2 3 100
3 4 58
4 1 999
0 0
```

Sample Output for the Sample Input

```
9600
0
32744
```

Solution

Brief (Mathematical) Description of the Problem

Find a minimum-product spanning tree.

Algorithm for Solving the Problem

If building one of the roads incurs no cost, the answer is clearly zero. In the sequel, assume all costs are positive. We only need to take logarithms to the costs and find a traditional minimum spanning tree. But even logarithms are unnecessary: Kruskal's and Prim's algorithms take care only of the *relative* ordering of the costs, and logarithms are strictly increasing functions anyway! So the algorithm is as follows:

- If there is a cost of zero, output 0.
- Otherwise, find a traditional minimum spanning tree and output accordingly.

Test Plan

I have implemented Kruskal's algorithm. I think it suffices to implement another algorithm for MST and see if the results match.