DFS Order: Extra Stage

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

You are given m depth-first search (DFS) orders of an unknown rooted tree.

We consider rooted trees with n vertices labeled from 1 to n, where vertex 1 is the root. The tree is undirected and connected, and has exactly $n - 1$ edges.

DFS definition. For a fixed rooted tree, a DFS order is obtained as follows:

- All vertices are initially unvisited.
- Start at the root 1, mark it as visited and **output** it.
- When you are at some vertex v, consider all children of v in the rooted tree. Choose its children in **any** order. For each chosen child u that is still unvisited, go to u, mark it as visited, output u, and continue the DFS from u in the same way.

For a fixed tree, different choices of the order in which the children of each vertex are processed may produce different DFS orders. All DFS orders are permutations of $1, 2, \ldots, n$ and always start with 1.

You are given m permutations of $1, 2, \ldots, n$, each starting with 1. You are told that each of them is a DFS order of the **same** rooted tree (with root 1) in the sense defined above (possibly using different choices of child orders for different DFS runs).

Your task is to determine how many different rooted trees could produce **all** of these m DFS orders.

Two rooted trees are considered different if their sets of edges are different.

Because the answer can be very large, you should output it modulo $10^9 + 7$.

Input

The first line contains two integers n and m ($1 \le n, m \le 500$).

Each of the next m lines contains a permutation $p_{k,1}, p_{k,2}, \ldots, p_{k,n}$ of the integers $1, 2, \ldots, n$, describing the k-th DFS order.

For every k, all $p_{k,i}$ are distinct and $p_{k,1} = 1$.

Output

Output a single integer: the number of rooted trees with vertex set $1, 2, \ldots, n$ and root 1 for which **every** given sequence is a valid DFS order of that tree, taken modulo $10^9 + 7$.

Examples

standard input	*standard output*
3 2 1 2 3 1 3 2	1
4 1 1 2 3 4	5
5 2 1 2 3 4 5 1 2 4 3 5	3