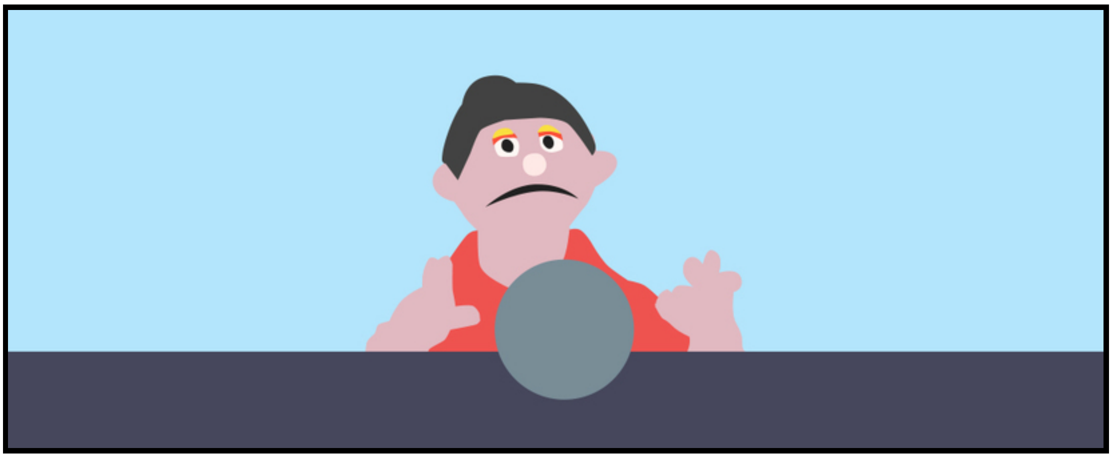
BATIBots



The year is 3000. Brainy Automated Transcendental Intelligent Bots (BATIBot for short) guard Mactan from colonizers.

Each BATIBot has a **starting** life number, a positive integer greater than or equal to 1. Starting life numbers of two different BATIBots may be the same. The life number of a BATIBot may be modified if they get attacked by evil bots.

Magell-1 wants to terrorize Mactan by sending an army of evil robots. They come in two types:

- A **Primebot** is an evil bot whose damage number is some prime number p. If it attacks a BATIBot whose current life number is not divisible by p, nothing happens. Otherwise, the BATIBot's current life number is divided by p. If the new life number is not divisible by p after the attack, the BATIBot immediately dies. **A Primebot immediately dies after attacking.**
- A **Lubot** is an evil bot whose damage number is a positive integer $m \geq 1$. If it attacks a BATIBot whose current life number is not divisible by m, nothing happens. If it attacks a BATIBot whose *starting* life number is m, nothing happens. Else, the BATIBot's current life number is divided by m. If the new life number is 1, the BATIBot dies.

A BATIBot is said to be **strong** if it is guaranteed to survive one attack of any Primebot or any number of attacks of any single Lubot.

Here are some examples:

- A BATIBot with starting life number 5 cannot be killed by a Lubot whose damage number is 5. However, it can be killed by a Primebot whose damage number is 5. Therefore, such a BATIBot is *not strong*.
- A BATIBot with starting life number 36 can be killed by two attacks of a Lubot with damage number 6. Therefore, such a BATIBot is *not strong*.
- A BATIBot with starting life number 7623 can be killed by a single attack

of a Primebot whose damage number is 7. Therefore, such a BATIBot is *not strong*.

- One can verify that a BATIBot with starting life number 9000 is *strong*.

Let us model the city of Mactan as a $w \times h$ grid drawn on the Cartesian plane with corners $(0,0)$, $(0,h)$, (w,h), and $(w,0)$. We denote by $\langle\langle x,y\rangle\rangle$ the cell which has four corners $(x-1,y-1)$, $(x-1,y)$, (x,y) and $(x,y-1)$. Cell $\langle\langle x,y\rangle\rangle$ has exactly one BATIBot with starting life number $b_{x,y}$.

A **barangay** of Mactan is defined as a rectangular subgrid of the original $w \times h$ grid. A barangay can then be represented by a pair of lattice points $\{(x_1,y_1),(x_2,y_2)\}$ with $0 \le x_1 < x_2 \le w$ and $0 \le y_1 < y_2 \le h$. With this representation, we can deduce that:

- it has four corners (x_1,y_1), (x_1,y_2), (x_2,y_2), and (x_2,y_1).
- the territory of the barangay is the set of cells $\langle\langle x,y\rangle\rangle$ such that $x_1 < x \le x_2$ and $y_1 < y \le y_2$.

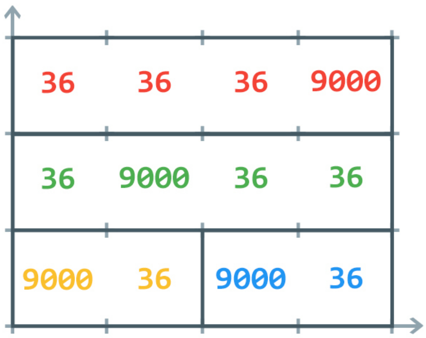
The **boundary** of barangay is defined as the set of points contained in any of its four sides, including the corners.

A barangay is said to be **protected** if it contains exactly one strong BATIBot.

A partition of Mactan into barangays is **valid** if and only if:

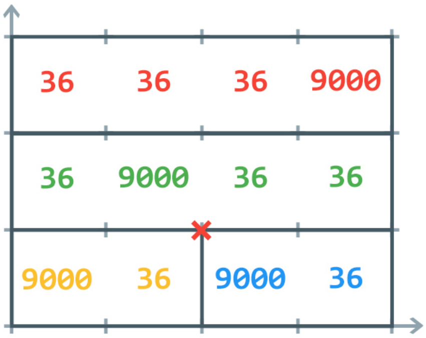
- Each cell belongs to exactly one barangay.
- No point is contained in the boundaries of exactly three barangays. It can be less than or greater than three.

For example, the following partition (into four barangays) is not valid:

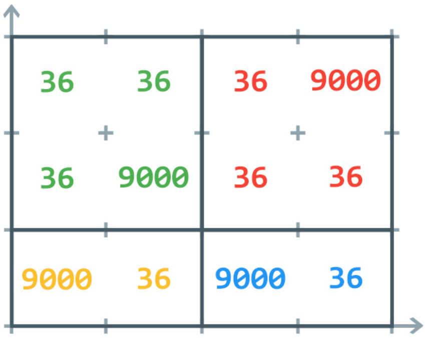


This is not a valid partition because although every cell is in exactly one barangay, the point $(2,1)$ is in the boundaries of exactly three barangays:

- $\{(0,0),(2,1)\}$.
- $\{(2,0),(4,1)\}$.
- $\{(0,1),(4,2)\}$.



On the other hand, you can verify that the following partition is valid:



Your task, assigned by Mayor Lapu-2, is to figure out how many ways one can validly partition Mactan into **protected barangays**. You decide how many barangays there are in the partition, as long as all of them turn out to be protected.

Input Format

The first line of input contains t, the number of test cases.

The first line of each test case contains two space-separated integers h and w. The next h lines contains w numbers each, and together, they describe the $w \times h$ grid in the following format:

$$\begin{array}{lllll}
b_{1,h} & b_{2,h} & \dots & b_{w-1,h} & b_{w,h} \\
b_{1,h-1} & b_{2,h-1} & \dots & b_{w-1,h-1} & b_{w,h-1} \\
\dots & \dots & \dots & \dots & \dots \\
b_{1,2} & b_{2,2} & \dots & b_{w-1,2} & b_{w,2} \\
b_{1,1} & b_{2,1} & \dots & b_{w-1,1} & b_{w,1}
\end{array}$$

In other words, $b_{x,y}$ is the xth number in the $(h-y+1)$th line.

Output Format

For each test case, output a single line containing a single integer denoting the answer modulo $10^9 + 7$.

Constraints and Subtasks

For all subtasks
$1 \leq t \leq 111$ $1 \leq w, h \leq 10^5$ $wh \leq 2 \cdot 10^5$ $1 \leq b_{x,y} \leq 10^{11}$ The sum of all wh in a single file is $\leq 4 \cdot 10^5$.

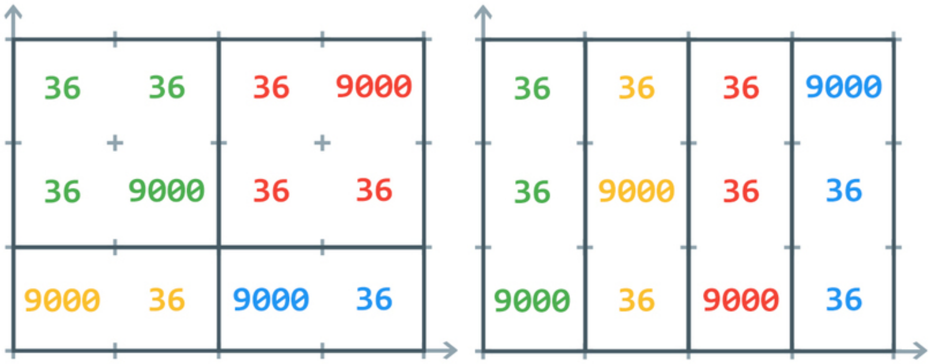
Subtask	Points	Constraints
1	**17**	$w, h \leq 60$ $\min(w, h) = 1$ $b_{x,y} \leq 100$
2	**15**	$wh \leq w + h$ $b_{x,y} \leq 10^6$
3	**18**	$wh \leq w + h$
4	**18**	$w, h \leq 60$ $b_{x,y} \leq 100$
5	**12**	$b_{x,y} \leq 100$
6	**20**	No additional constraints.

Sample I/O

Input	Output
1 3 4 36 36 36 9000 36 9000 36 36 9000 36 9000 36	2

Explanation

There are two valid partitions:



Thus, we output 2 modulo $10^9 + 7$ which is just 2.