Kind of Bingo

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

There is a grid with n rows and m columns. The cells in the grid are numbered from 1 to $n \times m$, where the cell on the i-th row and the j-th column is numbered as $((i-1) \times m + j)$.

Given a permutation $p_1, p_2, \cdots, p_{n\times m}$ of $n \times m$, we're going to perform $n \times m$ operations according to the permutation. For the i-th operation, we'll mark cell p_i. If after the b-th operation, there is at least one row such that all the cells in that row are marked, and b is as small as possible, then we say b is the "bingo integer" of the permutation.

You're given the chance to modify the permutation at most k times (including zero times). Each time you can swap a pair of elements in the permutation. Calculate the smallest possible bingo integer after the modifications.

Recall that a sequence $p_1, p_2, \cdots, p_{n\times m}$ of length $n \times m$ is a permutation of $n \times m$ if and only if each integer from 1 to $n \times m$ (both inclusive) appears exactly once in the sequence.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^4$) indicating the number of test cases. For each test case:

The first line contains three integers n, m and k ($1 \le n, m \le 10^5$, $1 \le n \times m \le 10^5$, $0 \le k \le 10^9$), indicating the number of rows and columns of the grid and the number of modifications you can perform.

The second line contains $n \times m$ distinct integers $p_1, p_2, \cdots, p_{n\times m}$ ($1 \le p_i \le n \times m$).

It's guaranteed that the sum of $n \times m$ of all test cases will not exceed 10^5.

Output

For each test case output one line containing one integer indicating the smallest possible bingo integer after the modifications.

Example

standard input	standard output

`3`
`3 5 2`
`1 4 13 6 8 11 14 2 7 10 3 15 9 5 12`
`2 3 0`
`1 6 4 3 5 2`
`2 3 1000000000`
`1 2 3 4 5 6`
	`7`
	`5`
	`3`

Note

For the first sample test case, we can first swap 1 and 15, then swap 6 and 12 to get the sequence $[15, 4, 13, 12, 8, 11, 14, 2, 7, 10, 3, 1, 9, 5, 6]$. It's easy to see that after the 7-th operation, all cells in the 3-rd row will be marked.

For the second sample test case, it's easy to see that after the 5-th operation, all cells in the 2-nd row will be marked.

For the third sample test case, we don't need to make any modifications. It's easy to see that after the 3-rd operation, all cells in the 1-st row will be marked.