Brought Down the Grading Server? (balance)

A hectic first competition day has passed... While the Scientific Committee narrowly fought off the hacking attack on the grading server, they fear that it affected the scoring of the submissions. There is only one option: all submissions have to be rejudged!

The grading server has N processor cores. The committee has already assigned a list of S submissions to each core, where each submission is to one of the T tasks of the competition (numbered $1, \ldots, T$). The committee made sure that S is a power of two.* Now, in the next S minutes, each core will evaluate exactly one submission from its list per minute.

Unfortunately, the database containing the task data is quite fragile and could crash if the number of simultaneous requests for a single task's data varies widely. Therefore, the committee wants to order the submissions of each core in such a way that, during the rejudging, the maximum and minimum number of simultaneously evaluated submissions for any single task differ by at most one.

Write a program which computes such an ordered assignment of the submissions to the cores.

Input

The first line of input contains the three integers N, S, and T described above.

Then, N lines follow describing the submissions assigned to the cores. The i-th of these lines contains S integers $t_1, \ldots, t_S$ ($1 \le t_j \le T$), meaning that the i-th core is assigned submissions to the tasks $t_1, \ldots, t_S$ respectively.

Output

Your program should output N lines that describe an ordered assignment of the submissions to the cores such that the maximum and minimum number of simultaneously evaluated submissions for any single task differ by at most one: The i-th of these lines should contain S integers $r_1, \ldots, r_S$, meaning that the i-th core evaluates a submission to task r_j during the j-th minute. It is guaranteed that such an assignment exists for each testcase.

Constraints and grading

We always have that $S = 2^k$ for some positive integer k, $1 \le N, S, T \le 100\,000$ and $N \cdot S \le 500\,000$.

Subtask 1 (10 points). $S = 2$ and $N, T \le 20$

Subtask 2 (15 + 5 + 5 points). $S = 2$

Subtask 3 (15 + 5 + 5 points). $N \cdot S \le 10\,000$

Subtask 4 (20 + 10 + 10 points). No further constraints.

Partial scoring. In Subtasks 2, 3, and 4, you can obtain partial scores as indicated in the brackets:

- The first number represents the number of points awarded if you solve all testcases in which $T \le N$ and the total number of submissions to each task is divisible by S.

---

* Due to a bug,† the judging system would crash otherwise, taking down all firewalls and potentially exposing sensitive information!

† You had *one* job, Wolfgang!

- The second number represents the number of additional points awarded if you more generally solve all testcases in which $T \leq N$.
- The third number represents the number of additional points awarded if you solve all testcases.

In the web interface, this is shown as "Group 1," "Group 2," and "Group 3" of the corresponding subtask.

Examples

Input	Output
3 2 3 1 2 2 3 2 3	2 1 3 2 2 3
3 4 3 2 3 2 2 2 3 3 2 2 2 3 2	2 2 2 3 3 2 3 2 2 3 2 2

In the output of the first example, the difference between the maximum and the minimum number of simultaneously evaluated submissions is one for tasks 1 and 2 and zero for task 3. On the other hand, ordering the submissions as in the input would not have constituted a valid output because the difference between the maximum and the minimum number of simultaneously evaluations submissions for task 3 is two.

In the output of the second example, the difference between the maximum and the minimum number of simultaneously evaluated submissions is zero for all three tasks.

Limits

Time: 2 s
Memory: 1024 MiB