Classroom Game

A prominent high school in Tashkent celebrates its anniversary by hosting a game between students and teachers. There are N students sitting in the classroom, numbered from 0 to $N-1$. Each student holds a piece of paper that contains an array of integers. Initially, each paper is blank, so it contains an empty array.

The students are playing the game with M teachers, numbered from 0 to $M-1$, and the headmaster. The game is played in M steps, also numbered from 0 to $M-1$. In step j, teacher j enters the classroom and the following events take place:

- Some (possibly zero) students raise their hand. It is guaranteed that in every step at least one student does **not** raise their hand, and each student may raise their hand **at most once** during the entire game.
- The teacher looks at the papers the students currently hold, and may **modify** the papers held by the students that did **not** raise their hand in this step. For each such student, the teacher can replace the contents of their paper with a new (possibly empty) array of integers. Each integer must be between 0 and 63, inclusive, and the array can have at most 63 elements.
- After these modifications, teacher j leaves and the students exchange their papers according to a secret permutation P. That is, each student i ($0 \leq i < N$) gives their paper to student $P[i]$, where $P[0], P[1], \ldots, P[N-1]$ are N **distinct** integers between 0 and $N-1$, inclusive. The values of P are **fixed** during all steps of the game, but the teachers and the headmaster do not know them.

Once all M steps have finished and the last exchange according to P has been performed, the headmaster enters. Looking only at the papers held by the students, the headmaster must determine, for every student i ($0 \leq i < N$), the step number j in which student i raised their hand, or conclude that student i never raised their hand.

The teachers cannot communicate with other teachers or the headmaster, except through the arrays of integers written on the papers. Each teacher knows the step at which they enter the classroom.

Your task is to devise and implement a strategy for the teachers and the headmaster that correctly identifies whether and when each student raised their hand. Your score will depend on the maximum length of any array written on a paper: a shorter maximum length yields a higher or equal score.

Implementation Details

You need to implement two procedures, one for the teachers and one for the headmaster.

The procedure you should implement for the **teachers** is:

```
std::vector<std::vector<int>> process_step(
    int N, int M, int R,
    std::vector<int> T,
    std::vector<std::vector<int>> A)
```

- N: the number of students.
- M: the number of steps, and also the number of teachers.
- R: the current step number (from 0 to $M-1$).
- T: a (possibly empty) array consisting of the indices of the students who raise their hand in this step, sorted in increasing order.
- A: an array of length N describing the papers, where $A[i]$ ($0 \leq i < N$) is the array of integers on the paper held by student i at the beginning of the step.
- This procedure is called exactly M times per game, in the order $R = 0, 1, \ldots, M-1$.

The procedure should return an array B of length N, representing the papers after the teacher's modifications, in the same format as A.

- For every student i that raised their hand (that is, i appears in T), the paper must not be changed, so $B[i] = A[i]$.
- For each i such that $0 \leq i < N$, the length of $B[i]$ must not exceed 63, and every integer in $B[i]$ must be between 0 and 63, inclusive.

The procedure you should implement for the **headmaster** is:

```
std::vector<int> determine_steps(
    int N, int M, std::vector<std::vector<int>> A)
```

- N, M: the same as above.
- A: an array of length N, where $A[i]$ is the array of integers on the paper held by student i after all M steps.
- This procedure is called exactly once per game, after the final call to `process_step`.

The procedure must return an array D of length N. For each i such that $0 \leq i < N$, it must hold that

- $D[i] = j$ if student i raised their hand during step j, or
- $D[i] = -1$ if student i never raised their hand during the entire game.

Your program is not allowed to store or transmit any information from one call to `process_step` to another, except through the return value of `process_step`. If your program attempts to do this, your score in CMS may be incorrect and may be reduced after the end of the contest. Note that the grader may execute multiple instances of your program simultaneously. This means that calls to `process_step` and `determine_steps` from the same game may be executed in different instances, and calls to `process_step` and `determine_steps` from different games may be executed in the same instance. Each test case includes up to 5 games.

Constraints

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Each student raises their hand **at most once** during the entire game. In other words, for every student i, there is at most one step j in which i raises their hand.
- At each step, at least one student does **not** raise their hand.

Subtasks and Scoring

Subtask	Score	Additional Constraints
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ for all $0 \leq i \leq N - 1$.
4	25	At most one student raises their hand at each step.
5	56	No additional constraints.

For each test case, your score is 0 (reported as `Output isn't correct` in CMS) if the return value of any call to procedure `process_step` is invalid or if the return value of any call to procedure `determine_steps` is incorrect.

Otherwise, let C be the maximum length of **any** array in **any** B returned by `process_step`. Then, the score for a test case in a subtask with score S is calculated as $S \cdot X$, where X depends on C according to the following table:

Condition	X
$C \leq 2$	1.00
$C = 3$	0.75
$C = 4$	0.55
$5 \leq C \leq 13$	$0.50 - 0.03 \cdot (C - 5)$
$14 \leq C \leq 63$	$0.19 \cdot \frac{64-C}{64-14} + 0.04$

Example

Consider a scenario with $N = 4$ students, $M = 2$ teachers, and permutation $P = [0, 3, 1, 2]$. Initially, all papers are empty.

The grader first calls:

```
process_step(4, 2, 0, [1], [[], [], [], []])
```

Student 1 has raised their hand, so their paper cannot be modified. Teacher 0 may decide to write $[10]$ on the paper of student 0, write $[1, 63, 4]$ on the paper of student 3, and leave student 2's paper empty. To do this, the procedure must return $B = [[10], [\,], [\,], [1, 63, 4]]$.

After teacher 0 leaves, the students exchange their papers according to P. After the exchange, the papers are $A = [[10], [\,], [1, 63, 4], [\,]]$.

The grader now calls:

```
process_step(4, 2, 1, [0,3], [[10], [], [1,63,4], []])
```

Students 0 and 3 have raised their hand, so their papers cannot be modified. Teacher 1 may decide to write $[0, 40]$ on the paper of student 1 and write $[1, 50]$ on the paper of student 2. To do this, the procedure must return $B = [[10], [0, 40], [1, 50], [\,]]$.

After teacher 1 leaves, the papers are again exchanged according to P. After the exchange, the papers are $A = [[10], [1, 50], [\,], [0, 40]]$.

Finally, the grader calls:

```
determine_steps(4, 2, [[10], [1,50], [], [0,40]])
```

This procedure must return the array $D = [1, 0, -1, 1]$ since students 0 and 3 raised their hand in step 1, student 1 raised their hand in step 0, and student 2 never raised their hand. In this

example, $C = 3$.

Sample Grader

Input format:

```
N M
Q[0] Q[1] ... Q[N-1]
P[0] P[1] ... P[N-1]
```

Here, Q is an array of length N, where $Q[i] = j$ if student i raises their hand during step j, or $Q[i] = -1$ if student i never raises their hand during the entire game.

Output format:

Following each call to `process_step`, the sample grader outputs the contents of the papers. Let K be the length of B and $L[i]$ be the length of $B[i]$ (for each $0 \leq i < K$). The sample grader prints B in the following format, **followed by an empty line**:

```
K
L[0] B[0][0] B[0][1] ... B[0][L[0]-1]
L[1] B[1][0] B[1][1] ... B[1][L[1]-1]
:
L[K-1] B[K-1][0] B[K-1][1] ... B[K-1][L[K-1]-1]
```

The sample grader then exchanges the papers according to permutation P. If $K \neq N$, the sample grader prints an error message and terminates instead.

After calling `determine_steps`, the sample grader outputs:

```
C H
D[0] D[1] ... D[H-1]
```

Here, H is the length of the array D returned by `determine_steps`.