

Juego de Salon

Un prestigioso instituto de Tashkent celebra su aniversario organizando un partido entre alumnos y profesores. Hay N estudiantes sentados en el aula, numerados del 0 al $N - 1$. Cada estudiante sostiene una hoja de papel que contiene un arreglo de números enteros. Inicialmente, cada hoja está en blanco, por lo que contiene un arreglo vacío.

Los estudiantes están jugando con M profesores, numerados del 0 al $M - 1$, y el director. El juego se juega en M pasos, también numerados del 0 al $M - 1$. En el paso j , el profesor j entra al aula y tienen lugar los siguientes eventos:

- Algunos estudiantes (posiblemente ninguno) levantan la mano. Se garantiza que en cada paso al menos un estudiante **no** levanta la mano, y cada estudiante puede levantar la mano **como máximo una vez** durante todo el juego.
- El profesor revisa los papeles que los alumnos tienen en sus manos y puede **modificar** los papeles de los alumnos que **no** levantaron la mano en este paso. Para cada alumno, el profesor puede reemplazar el contenido de su papel con un nuevo arreglo de números enteros (posiblemente vacío). Cada número entero debe estar entre 0 y 63 , ambos inclusive, y el arreglo puede tener como máximo 63 elementos.
- Después de estas modificaciones, el profesor j se va y los estudiantes intercambian sus trabajos según una permutación secreta P . Es decir, cada estudiante i ($0 \leq i < N$) le da su trabajo al estudiante $P[i]$, donde $P[0], P[1], \dots, P[N - 1]$ son N enteros **distintos** entre 0 y $N - 1$, ambos inclusive. Los valores de P son **fijos** durante todas las etapas del juego, pero los profesores y el director no los conocen.

Una vez que todos los M pasos hayan terminado y se haya realizado el último intercambio según P , entra el director. Mirando solo los papeles que sostienen los estudiantes, el director debe determinar, para cada estudiante i ($0 \leq i < N$), el número de paso j en el que el estudiante i levantó la mano, o concluir que el estudiante i nunca levantó la mano.

No pueden comunicarse con otros profesores ni con el director, excepto a través de las secuencias de números enteros escritas en los papeles.

Tu tarea consiste en diseñar e implementar una estrategia para los profesores y el director que permita identificar correctamente si cada alumno levantó la mano y cuándo lo hizo. Tu puntuación dependerá de la longitud máxima de cualquier secuencia escrita en el papel; una longitud máxima menor podría resultar en una puntuación mayor o igual.

Detalles de implementación

Es necesario implementar dos procedimientos, uno para los profesores y otro para el director.

El procedimiento que debe implementar para los **profesores** es:

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

- N : el número de estudiantes.
- M : el número de pasos y también el número de profesores.
- R : el número de paso actual (de 0 a $M - 1$).
- T : un arreglo (posiblemente vacío) que consta de los índices de los estudiantes que levantan la mano en este paso, ordenados en orden ascendente.
- A : un arreglo de longitud N que describe los papeles, donde $A[i]$ ($0 \leq i < N$) es el arreglo de enteros en el papel que sostiene el estudiante i al comienzo del paso.
- Este procedimiento se llama exactamente M veces por juego, en el orden $R = 0, 1, \dots, M - 1$.

El procedimiento debe devolver un arreglo B de longitud N , que represente los papeles después de las modificaciones del profesor, en el mismo formato que A .

- Por cada estudiante i que levantó la mano (es decir, i aparece en T), el papel no debe cambiarse, por lo que $B[i] = A[i]$.
- Para cada i tal que $0 \leq i < N$, la longitud de $B[i]$ no debe exceder 63, y cada entero en $B[i]$ debe estar entre 0 y 63, ambos inclusive.

El procedimiento que debe implementar para el **director** es:

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

- N, M : lo mismo que el anterior procedimiento.
- A : un arreglo de longitud N , donde $A[i]$ es el arreglo de enteros en el papel que sostiene el estudiante i después de todos M pasos.
- Este procedimiento se llama exactamente una vez por juego, después de la llamada final a `process_step`.

El procedimiento debe devolver un arreglo D de longitud N . Para cada i tal que $0 \leq i < N$, debe cumplirse que

- $D[i] = j$ si el estudiante i levantó la mano durante el paso j , o

- $D[i] = -1$ si el estudiante i nunca levantó la mano durante todo el juego.

Su programa no puede almacenar ni transmitir información de una llamada a `process_step` a otra, excepto a través del valor de retorno de `process_step`. Si su programa intenta hacerlo, su puntuación en CMS podría ser incorrecta y podría reducirse al finalizar el concurso. Tenga en cuenta que el evaluador puede ejecutar varias instancias de su programa simultáneamente. Esto significa que las llamadas a `process_step` y `determine_steps` del mismo juego pueden ejecutarse en instancias diferentes, y las llamadas a `process_step` y `determine_steps` de juegos diferentes pueden ejecutarse en la misma instancia. Cada caso de prueba incluye hasta 5 juegos.

Restricciones

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Cada estudiante levanta la mano **como máximo una vez** durante todo el juego. En otras palabras, para cada estudiante i , hay como máximo un paso j en el que i levanta la mano.
- En cada paso, al menos un estudiante **no** levanta la mano.

Subtareas y Puntuación

Subtarea	Puntuación	Restricciones adicionales
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ para todo $0 \leq i \leq N - 1$.
4	25	Como máximo, un estudiante levanta la mano en cada paso.
5	56	Sin restricciones adicionales

Para cada caso de prueba, su puntuación es 0 (se informa como `Output isn't correct` en CMS) si el valor de retorno de cualquier llamada al procedimiento `process_step` no es válido o si el valor de retorno de cualquier llamada al procedimiento `determine_steps` es incorrecto.

De lo contrario, sea C la longitud máxima de **cualquier** arreglo en **cualquier** B devuelta por `process_step`. Entonces, la puntuación para un caso de prueba en una subtarea con puntuación S se calcula como $S \cdot X$, donde X depende de C según la siguiente tabla:

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

Ejemplo

Considera un escenario con $N = 4$ estudiantes, $M = 2$ profesores y permutación $P = [0, 3, 1, 2]$. Inicialmente, todos los papeles están vacíos.

El evaluador llama primero:

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

El estudiante 1 ha levantado la mano, por lo que su trabajo no puede ser modificado. El profesor 0 puede decidir escribir $[10]$ en el papel del estudiante 0, escribir $[1, 63, 4]$ en el papel del estudiante 3 y dejar el papel del estudiante 2 vacío. Para hacer esto, el procedimiento debe devolver $B = [[10], [], [], [1, 63, 4]]$.

Después de que el profesor 0 se va, los estudiantes intercambian sus trabajos según P . Después del intercambio, los trabajos son $A = [[10], [], [1, 63, 4], []]$.

El evaluador ahora llamara:

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

Los estudiantes 0 y 3 han levantado la mano, por lo que sus trabajos no pueden ser modificados. El profesor 1 puede decidir escribir $[0, 40]$ en el papel del estudiante 1 y escribir $[1, 50]$ en el papel del estudiante 2. Para hacer esto, el procedimiento debe devolver $B = [[10], [0, 40], [1, 50], []]$.

Después de que el profesor 1 se va, los papeles se intercambian de nuevo según P . Después del intercambio, los papeles son $A = [[10], [1, 50], [], [0, 40]]$.

Finalmente, el evaluador llamara:

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

Este procedimiento debe devolver el arreglo $D = [1, 0, -1, 1]$ ya que los estudiantes 0 y 3 levantaron la mano en el paso 1, el estudiante 1 levantó la mano en el paso 0 y el estudiante 2 nunca levantó la mano. En este ejemplo, $C = 3$.

Grader de ejemplo

Formato de entrada:

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

Aquí, Q es un arreglo de longitud N , donde $Q[i] = j$ si el estudiante i levanta la mano durante el paso j , o $Q[i] = -1$ si el estudiante i nunca levanta la mano durante todo el juego.

Formato de salida:

Tras cada llamada a `process_step`, el grader de ejemplo muestra el contenido de los documentos. Sea K la longitud de B y $L[i]$ la longitud de $B[i]$ (para cada $0 \leq i < K$). El grader de ejemplo imprime B en el siguiente formato, **seguido de una línea vacía**:

```
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]
```

El grader de ejemplo luego intercambia los trabajos según la permutación P . Si $K \neq N$, el grader de ejemplo imprime un mensaje de error y finaliza.

Después de llamar a `determine_steps`, el grader de ejemplo produce la siguiente salida:

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

Aquí, H es la longitud del arreglo D devuelto por `determine_steps`.