

Juego de Salón de Clases

Un reconocido colegio de secundaria de Tashkent celebra su aniversario organizando un juego entre estudiantes y profesores. Hay N estudiantes sentados en el salón de clases, numerados de 0 a $N - 1$. Cada estudiante tiene una hoja de papel que contiene un arreglo de enteros. Inicialmente, cada hoja está vacía, es decir, contiene un arreglo vacío.

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

- Algunos estudiantes (posiblemente cero) levantan su mano. Se garantiza que en cada paso al menos un estudiante **no** levanta su mano, y cada estudiante puede levantar su mano **a lo más una vez** durante todo el juego.
- El profesor mira las hojas que los estudiantes tienen actualmente y puede **modificar** las hojas de los estudiantes que **no** levantaron su mano en este paso. Para cada uno de dichos estudiantes, el profesor puede reemplazar los contenidos de su hoja con un arreglo nuevo (posiblemente vacío) de enteros. Cada entero debe estar entre 0 y 63, inclusive, y el arreglo puede tener a lo más 63 elementos.
- Después de estas modificaciones, el profesor j sale del salón y los estudiantes intercambian sus hojas de acuerdo a una permutación secreta P . Esto es, cada estudiante i ($0 \leq i < N$) le da su hoja al estudiante $P[i]$, donde $P[0], P[1], \dots, P[N - 1]$ son N enteros **distintos** entre 0 y $N - 1$, inclusive. Los valores de P son **fijos** durante todos los pasos del juego, pero los profesores y el director no los conocen.

Una vez que se han terminado los M pasos y se ha hecho el último intercambio de acuerdo a P , entra el director. Mirando solamente las hojas que tienen los estudiantes, el director debe determinar, para cada estudiante i ($0 \leq i < N$), el paso j en el cual el estudiante i levantó su mano, o concluir que el estudiante i nunca levantó su mano.

Los profesores no pueden comunicarse con otros profesores ni con el director, excepto a través de los arreglos de enteros escritos en las hojas de los estudiantes. Cada profesor conoce el paso en el cual entra al salón de clases.

Su tarea es diseñar e implementar una estrategia para los profesores y el director de modo que este último identifique correctamente si cada estudiante levantó su mano y cuando lo hizo. Su

puntaje dependerá de la longitud máxima de cualquier arreglo escrito en una hoja - una longitud máxima más pequeña producirá un puntaje mayor o igual.

Detalles de Implementación

Usted necesita implementar dos procedimientos, uno para los profesores y uno para el director.

El procedimiento que usted 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 del paso actual (de 0 a $M - 1$).
- T : un arreglo (posiblemente vacío) que contiene los índices de los estudiantes que levantaron su mano en este paso, ordenados de forma creciente.
- A : un arreglo de longitud N describiendo las hojas de los estudiantes, donde $A[i]$ ($0 \leq i < N$) es el arreglo de enteros escrito en la hoja del estudiante i al inicio del paso actual.
- 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 , en el mismo formato que A , representando las hojas después de las modificaciones del profesor.

- Para cada estudiante i que levantó su mano (es decir, i aparece en T), la hoja no debe ser cambiada, por lo cual $B[i] = A[i]$.
- Para cada i tal que $0 \leq i < N$, la longitud de $B[i]$ no debe exceder 63, y todo entero presente en $B[i]$ debe estar entre 0 y 63, 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 se describe arriba.
- A : Un arreglo de longitud N , donde $A[i]$ es el arreglo de enteros escrito en la hoja del estudiante i después de todos los M pasos.
- Este procedimiento es llamado 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 $0 \leq i < N$, se debe cumplir que:

- $D[i] = j$ si el estudiante i levantó su mano durante el paso j , o
- $D[i] = -1$ si el estudiante i nunca levantó su mano durante el juego.

Su programa no tiene permitido almacenar ni transmitir ninguna información de una llamada a `process_step` hacia otra, excepto a través del valor de retorno de `process_step`. Si intenta hacer esto, su puntuación en CMS puede ser incorrecta y puede reducirse después del final del concurso. Note que el evaluador puede ejecutar múltiples instancias de su programa simultáneamente. Esto significa que las llamadas a `process_step` y `determine_steps` del mismo juego pueden ejecutarse en diferentes instancias, y llamadas a `process_step` y `determine_steps` de diferentes juegos pueden ejecutarse en la misma instancia. Cada caso de prueba incluye a lo más 5 juegos.

Restricciones

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Cada estudiante levanta su mano **a lo más una vez** durante todo el juego. En otras palabras, para todo estudiante i , hay a lo más un paso j en el cual i levanta la mano.
- En cada paso, al menos un estudiante **no** levanta su 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	A lo más un estudiante levanta su mano en cada paso.
5	56	Sin restricciones adicionales.

Para cada caso de prueba, su puntaje es 0 (reportado como `Output isn't correct` en CMS) si el valor devuelto por cualquier llamada al procedimiento `process_step` es invalido o si el valor devuelto por cualquier llamada al procedimiento `determine_steps` es incorrecto.

De lo contrario, sea C la máxima longitud de **cualquier** arreglo en **cualquier** B devuelto por `process_step`. Entonces, la puntuación para un caso de prueba de una subtaska con puntaje S es calculada como $S \cdot X$, donde X depende de C como se muestra en 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

Considere un escenario con $N = 4$ estudiantes, $M = 2$ profesores y la permutación $P = [0, 3, 1, 2]$. Inicialmente, todas las hojas están vacías.

El evaluador llama primero:

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

El estudiante 1 ha levantado la mano, por lo que su hoja no puede ser modificada. El profesor 0 puede decidir escribir $[10]$ en la hoja del estudiante 0, escribir $[1, 63, 4]$ en la hoja del estudiante 3 y dejar la hoja del estudiante 2 en blanco. Para ello, el procedimiento debe devolver $B = [[10], [], [], [1, 63, 4]]$.

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

El evaluador ahora llama:

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

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

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

Finalmente, el evaluador llama:

```
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, $C = 3$.

Evaluador

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 evaluador muestra el contenido de las hojas. Sea K la longitud de B y $L[i]$ la longitud de $B[i]$ (para cada $0 \leq i < K$). El evaluador imprime B en el siguiente formato, **seguido de una línea vacía**:

```
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 evaluador luego intercambia las hojas según la permutación P . Si $K \neq N$, el evaluador imprime un mensaje de error y finaliza.

Después de llamar a `determine_steps`, el evaluador 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`.