Tickets del Carnaval (tickets)

Ringo está en el carnaval en Singapur. El tiene algunos tickets de premios en su cartera, los cuales le gustaría usar en el quioso de premios. Cada ticket viene en uno de los n colores y tiene un entero no negativo impreso en él. Los enteros impresos en distintos tickets podrían ser los mismos. Debido a un capricho en las reglas del carnaval, está garantizado que n es **par**.

Ringo tiene m tickets de cada color en su cartera, ese es un total de $n \cdot m$ tickets. El ticket j del color i tiene el entero $x[i][j]$ impreso en él ($0 \leq i \leq n-1$ y $0 \leq j \leq m-1$).

El juego de premios es jugado en k rondas, numeradas de 0 a $k-1$. Cada ronda es jugada en el siguiente orden:

- De su cartera, Ringo selecciona un **conjunto** de n tickets, un ticket de cada color. El luego da el conjunto al dueño del juego.
- El dueño del juego anota los enteros $a[0]$, $a[1]$ $\ldots$ $a[n-1]$ impresos en los tickets del conjunto. El orden de esos n enteros no es importante.
- El dueño del juego saca una carta especial de un caja de dibujo de la suerte y anota el entero b impreso en esa caja.
- El dueño del juego calcula las diferencias absolutas entre $a[i]$ y b para cada i de 0 a $n-1$. Sea S la suma de esas diferencias absolutas.
- Para esta ronda, el dueño del juego le da a Ringo un premio con el valor igual a S.
- Los tickets en el conjunto son descartados y no pueden ser usados en las siguientes rondas.

Los tickets que sobran en la cartera de Ringo después de k rondas del juego son descartados.

Observando con detenimiento, Ringo se dió cuenta que el juego está arreglado! Hay una impresora dentro de la caja de dibujo de la suerte. En cada ronda, el dueño del juego busca un entero b que minimice el valor del premio de esa ronda. El valor elegido por el dueño del juego es impreso en la carta especial para esa ronda.

Teniendo toda esta información, Ringo quisiera colocar los tickets a las rondas del juego. Esto es, el quiere seleccionar el conjunto de tickets que usará en cada ronda para maximizar el valor total de los premios.

Detalles de implementación

Debes implementar el siguiente procedimiento:

```
int64 find_maximum(int k, int[][] x)
```

- k: el número de rondas.
- x: un arreglo $n \times m$ describiendo los enteros en cada ticket. Los tickets de cada color están ordenados en orden no decreciente de sus enteros.
- Este procedimiento debe ser llamado exactamente una vez.
- Este procedimiento debe hacer exactamente una llamada a `allocate_tickets` (ver abajo), describiendo k conjuntos de tickets, uno por cada ronda. La asignación debe maximizar el valor total de premios
- Este procedimiento debe retornar el máximo valor total de los premios.

El procedimiento `allocate_tickets` está definido como sigue:

```
void allocate_tickets(int[][] s)
```

- s: un arreglo $n \times m$. El valor de $s[i][j]$ debe ser r si el ticket j del color i es usado en el conjunto de la ronda r del juego, o -1 si no es usado para nada.
- Para cada $0 \leq i \leq n-1$, entre $s[i][0], s[i][1], \ldots, s[i][m-1]$ cada valor $0, 1, 2, \ldots, k-1$ debe ocurrir exactamente una vez, y todas las demás entradas deben ser -1.
- Si hay multiples asignaciones que devuelven el máximo valor total del premio, está permitido reportar cualquiera de ellas.

Ejemplos

Ejemplo 1

Considere la siguiente llamada:

```
find_maximum(2, [[0, 2, 5],[1, 1, 3]])
```

Esto significa que:

- hay $k = 2$ rondas;
- los enteros impresos en los tickets de color 0 son 0, 2 y 5 respectivamente;
- los enteros impresos en los tickets de color 1 son 1, 1 y 3, respectivamente.

Una posible asignación que da el máximo valor del premio es:ue is:

- En la ronda 0, Ringo toma el ticket 0 de color 0 (con el entero 0) y el ticket 2 de color 1 (con el entero 3). El valor mas pequeño posible del premio en esta ronda es 3. Ejemplo, el dueño dle juego podría elegir $b = 1$: $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- En la ronda 1, Ringo toma el ticket 2 de color 0 (con el entero 5) y el ticket 1 de color 1 (con el entero 1). El valor más pequeño posible del premio en esta ronda es 4. Ejemplo, el dueño del juego podría elegir $b = 3$: $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- Por lo tanto, el valor total de los premios podría ser $3 + 4 = 7$.

Para reportar esta asignación, el procedimiento `find_maximum` debería hacer la siguiente llamada a `allocate_tickets`:

- `allocate_tickets([[0, -1, 1], [-1, 1, 0]])`

Finalmente, el procedimiento `find_maximum` debe retornar 7.

Ejemplo 2

Considere la siguiente llamada:

```
find_maximum(1, [[5, 9], [1, 4], [3, 6], [2, 7]])
```

Esto significa que:

- solo hay una ronda,
- los enteros impresos en los tickets de color 0 son 5 y 9, respectivamente;
- los enteros impresos en los tickets de color 1 son 1 y 4, respectivamente;
- los enteros impresos en los tickets de color 2 son 3 y 6, respectivamente;
- los enteros impresos en los tickets de color 3 son 2 y 7, respectivamente;

Una posible asignación que da el valor máximo de premio es:

- En la ronda 0, Ringo elige el ticket 1 de color 0 (con el entero 9), el ticket 0 de color 1 (con el entero 1), ticket 0 con el color 2 (con el entero 3), y el ticket 1 de color 3 (con el entero 7). El mínimo posible valor del premio en esta ronda es 12, cuando el dueño del juego elige $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.
- In round 0, Ringo picks ticket 1 of colour 0 (with the integer 9), ticket 0 of colour 1 (with the integer 1), ticket 0 of colour 2 (with the integer 3), and ticket 1 of colour 3 (with the integer 7). The lowest possible value of the prize in this round is 12, when the game master chooses $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

Para reportar esta asignación, el procedimiento `find_maximum` debe hacer la siguiente llamada a `allocate_tickets`:

- `allocate_tickets([[-1, 0], [0, -1], [0, -1], [-1, 0]])`

Finalmente, el procedimiento `find_maximum` debe retornar 12.

Límites

- $2 \leq n \leq 1500$ y n es par.
- $1 \leq k \leq m \leq 1500$
- $0 \leq x[i][j] \leq 10^9$ (para todo $0 \leq i \leq n - 1$ y $0 \leq j \leq m - 1$)
- $x[i][j - 1] \leq x[i][j]$ (para todo $0 \leq i \leq n - 1$ y $1 \leq j \leq m - 1$)

Subtareas

1. (11 puntos) $m = 1$
2. (16 puntos) $k = 1$
3. (14 puntos) $0 \le x[i][j] \le 1$ (para todo $0 \le i \le n-1$ y $0 \le j \le m-1$)
4. (14 puntos) $k = m$
5. (12 puntos) $n, m \le 80$
6. (23 puntos) $n, m \le 300$
7. (10 puntos) Sin restricciones adicionales.

Evaluador de ejemplo

El evaluador de ejemplo lee la entrada en el siguiente formato:

- línea 1: n m k
- línea $2 + i$ ($0 \le i \le n-1$): $x[i][0]$ $x[i][1]$ $\ldots$ $x[i][m-1]$

El evaluador de ejemplo imprime tu respuesta en el siguiente formato:

- línea 1: el valor de retorno de `find_maximum`
- línea $2 + i$ ($0 \le i \le n-1$): $s[i][0]$ $s[i][1]$ $\ldots$ $s[i][m-1]$