Entradas para el carnaval (tickets)

Ringo está en un carnaval en Singapur. Tiene algunos tickets de premio en su bolso, que le gustaría usar en el puesto de juego de los premios. Cada ticket viene en uno de los n colores y tiene impreso un número entero no negativo. Los números enteros impresos en los diferentes tickets pueden ser los mismos. Debido a una peculiaridad en las reglas del carnaval, n está garantizado que sea **incluido**.

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

El juego del premio se juega en k rondas, numeradas de 0 a $k-1$. Cada ronda se juega en el siguiente orden:

- De su bolsa, Ringo selecciona un **conjunto** de n tickets, un ticket de cada color. Luego le da el conjunto al organizador del juego.
- El organizador del juego anota los números enteros $a[0],\ a[1]\ \ldots\ a[n-1]$ impresos en el conjunto de tickets. El orden de estos n números enteros no es importante.
- El organizador del juego saca una tarjeta especial de una caja de sorteo y anota el entero b impreso en esa tarjeta.
- El organizador de juego calcula las diferencias absolutas entre $a[i]$ y b por cada i desde 0 hasta $n-1$. Dejemos que S sea la suma de estas diferencias absolutas.
- En esta ronda, el organizador del juego le da a Ringo un premio con un valor igual a S.
- Los tickets del juego se descartan y no pueden ser usados en rondas futuras.

Los tickets restantes en la bolsa de Ringo después de k rondas del juego se descartan.

Al mirar de cerca, Ringo se dio cuenta de que el juego de premios está amañado! En realidad hay una impresora dentro de la caja del sorteo de la suerte. En cada ronda, el organizador del juego encuentra un entero b que minimiza el valor del premio de esa ronda. El valor elegido por el organizador de juego se imprime en la tarjeta especial de esa ronda.

Con toda esta información, Ringo desea asignar los tickets a las rondas del juego. Es decir, quiere seleccionar el ticket que se utilizará en cada ronda para maximizar el valor total de los premios.

Detalles de la Implementación

Debería implementar el siguiente procedimiento:

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

- k: el número de rondas.
- x: un arreglo $n \times m$ que describe los números enteros de cada ticket. Los tickets de cada color están ordenados en orden no decreciente de sus números enteros.
- Este procedimiento se llama exactamente una vez.
- Este procedimiento debe hacer exactamente una llamada a `allocate_tickets` (ver debajo), describiendo el conjunto de k tickets, uno para cada ronda. La asignación debe maximizar el valor total de los premios.
- Este procedimiento debe devolver el máximo valor total de los premios.

El procedimiento `allocate_tickets` se definido de la siguiente manera:

```
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 se usa en el conjunto de la ronda r en el juego, o -1 si no se usa en absoluto.
- 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 múltiples asignaciones que resultan en el máximo valor total del premio, se permite informar sobre 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 números enteros impresos en los tickets de color 0 son 0, 2 y 5, respectivamente;
- los números enteros impresos en los tickets de color 1 son 1, 1 y 3, respectivamente.

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

- En la ronda 0, Ringo escoge el ticket 0 de color 0 (con el entero 0) y el ticket 2 de color 1 (con el entero 3). El valor más bajo posible del premio en esta ronda es 3. Por ejemplo, el organizador del juego puede elegir $b = 1$: $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- En la ronda 1, Ringo escoge el ticket 2 de color 0 (con el entero 5) y el ticket 1 de color 1 (con el entero 1). El valor más bajo posible del premio en esta ronda es 4. Por ejemplo, el organizador del juego puede elegir $b = 3$: $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- Por lo tanto, el valor total de los premios debe ser $3 + 4 = 7$.

El reporte de esta asignación, el procedimiento `find_maximum` debe hacer la siguiente llamada a `allocate_tickets`:

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

Finalmente, el procedimiento `find_maximum` debería retornar 7.

Ejemplo 2

Considere la siguiente llamada:

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

Esto significa que:

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

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

- En la ronda 0, Ringo escoge el ticket 1 de color 0 (con el entero 9), el ticket 0de color 1 (con el entero 1), el ticket 0 de color 2 (con el entero 3), y el ticket 1 de color 3 (con el entero 7). El valor más bajo posible del premio en esta ronda es 12, cuando el organizador del juego selecciona $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

El reporte de 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.

Restricciones

- $2 \leq n \leq 1500$ y n es incluido.
- $1 \leq k \leq m \leq 1500$
- $0 \leq x[i][j] \leq 10^9$ (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) $0 \leq x[i][j] \leq 1$ (for all $0 \leq i \leq n-1$ and $0 \leq j \leq m-1$)
4. (14 puntos) $k = m$
5. (12 puntos) $n, m \leq 80$
6. (23 puntos) $n, m \leq 300$
7. (10 puntos) No hay restricciones adicionales.

Grader de ejemplo

El grader de ejemplo lee la entrada con el siguiente formato:

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

El grader de ejemplo imprime su respuesta em el siguiente formato:

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