Boletos del Carnaval (tickets)

Ringo está en un carnaval en Singapur. Tiene unos tickets para premios en su bolsa, que quiere intercambiar por premios. Cada boleto viene en uno de n colores y tiene un entero no negativo impreso en él. Puede que los enteros impresos en diferentes boletos sean iguales. Por razones extrañas de las reglas del carnaval, n siempre es **par**.

Ringo tiene m boletos de cada color en su bolsa. Es decir, tiene un total de $n \cdot m$ boletos. El boleto j de color i tiene al entero $x[i][j]$ impreso en él. (Para $0 \leq i \leq n-1$ y $0 \leq j \leq m-1$.)

El juego para obtener premios tiene k rondas, numeradas de 0 a $k-1$. Cada ronda se juega en este orden:

- De su bolsa, Ringo selecciona un **conjunto** de n boletos: uno de cada color. Después le da esos boletos al anfitrión.
- El anfitrión apunta los enteros $a[0],\ a[1]\ \ \ldots\ \ a[n-1]$ impresos en los boletos que recibió. El orden de estos n enteros no es importante.
- El anfitrión saca una tarjeta especial de su caja de la suerte y apunta el entero b que esté escrito en esa tarjeta.
- El anfitrión calcula las diferencias absolutas entre $a[i]$ y b para toda i entre 0 y $n-1$. Después, calcula S: la suma de esas diferencias absolutas.
- Para esta ronda, el anfitrión le otorga a Ringo un premio con valor igual a S.
- Los boletos del conjunto se descartan y no se pueden utilizar en rondas futuras.

Los boletos restantes en la bolsa de Ringo después de k rondas también se descartan.

Después de observar con detenimiento, ¡Ringo se dio cuenta de que el anfitrión está haciendo trampa! Resulta que hay una impresora en la caja de la suerte. En cada ronda, el anfitrión escoge un entero b de tal manera que minimice el valor del premio de esa ronda. Posteriormente, el valor elegido por el anfitrión es impreso en la tarjeta especial de esa ronda.

Sabiendo todo esto, Ringo quisiera asignar boletos a las rondas del juego. Es decir, quiere escoger el conjunto de boletos que debe usar en cada ronda para maximizar el valor total de los premios.

Detalles de la implementación

Debes implementar la siguiente función:

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

- k: el número de rondas.

- x: un arreglo de $n \times m$ que describe los enteros en cada boleto. Los boletos de cada color están ordenados de forma no decreciente por sus enteros.
- Esta función es llamada exactamente una vez.
- Esta función debe hacer exactamente una llamada a `allocate_tickets` (ver debajo), describiendo k conjuntos de boletos: uno por cada ronda. Esta asignación debe maximizar el valor total de los premios.
- Esta función debe regresar el valor máximo total de los premios.

La función `allocate_tickets` se define de la siguiente manera:

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

- s: un arreglo de $n \times m$. El valor de $s[i][j]$ debe ser r si el boleto j de color i se usa en la ronda r del juego, o -1 si no se usa nunca.
- Para $0 \leq i \leq n-1$, en $s[i][0], s[i][1], \ldots, s[i][m-1]$ cada uno de los valores $0, 1, 2, \ldots, k-1$ debe ocurrir exactamente una vez, y el resto de las entradas deben ser -1.
- Si hay más de una asignación que obtenga el máximo valor total de los premios, puedes reportar cualquiera de ellas.

Ejemplos

Ejemplo 1

Considera la siguiente llamada:

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

Eso significa que:

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

Una posible asignación que da el valor máximo total posible de los premios es:

- En la ronda 0, Ringo escoge el boleto 0 de color 0 (que tiene escrito el entero 0) y el boleto 2 de color 1 (que tiene el entero 3). El mínimo valor posible de un premio en esta ronda es 3. Por ejemplo, el anfitrión puede escoger $b = 1$: $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- En la ronda 1, Ringo escoge el boleto 2 de color 0 (que tiene escrito el entero 5) y el boleto 1 de color 1 (que tiene el entero 1). El mínimo valor posible de un premio en esta ronda es 4. Por ejemplo, el anfitrión puede escoger $b = 3$: $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- Por lo tanto, el valor total de los premios sería $3 + 4 = 7$.

Para reportar esta asignación, la función `find_maximum` debe hacer la siguiente llamada a

`allocate_tickets`:

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

Por último, la función `find_maximum` debe retornar 7.

Ejemplo 2

Considera la siguiente llamada:

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

Esto significa que:

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

Una posible asignación que da el máximo valor total posible para los premios es:

- En la ronda 0, Ringo escoge el boleto 1 de color 0 (con el entero 9), el boleto 0 de color 1 (con el entero 1), el boleto 0 de color 2 (con el entero 3), y el boleto 1 de color 3 (con el entero 7). El mínimo valor posible para un premio en esa ronda es 12, si el anfitrión escoge $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

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

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

Por último, la función `find_maximum` debe regresar 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 toda $0 \leq i \leq n - 1$ y $0 \leq j \leq m - 1$)
- $x[i][j - 1] \leq x[i][j]$ (para toda $0 \leq i \leq n - 1$ y $1 \leq j \leq m - 1$)

Subtareas

1. (11 puntos) $m = 1$
2. (16 puntos) $k = 1$
3. (14 puntos) $0 \leq x[i][j] \leq 1$ (para toda $0 \leq i \leq n - 1$ y $0 \leq j \leq m - 1$)

4. (14 puntos) $k = m$
5. (12 puntos) $n, m \leq 80$
6. (23 puntos) $n, m \leq 300$
7. (10 puntos) Sin límites adicionales.

Grader de Ejemplo

El grader de ejemplo lee la entrada en 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 tu respuesta en el siguiente formato:

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