Tickets de Carnaval (tickets)

Ringo est à un carnaval à Singapour. Il a dans son sac plusieurs tickets gagnants qu'il aimerait utiliser au stand des récompenses. Il existe n couleurs de tickets. Sur chaque ticket est imprimé un entier positif ou nul. Les entiers imprimés sur des tickets différents peuvent être égaux. Suite à une bizarrerie dans les règles du carnaval, on garantit que n est un entier **pair**.

Ringo a m tickets de chaque couleur, pour un total de $n \cdot m$ tickets. Sur le ticket j de la couleur i est imprimé le numéro $x[i][j]$ ($0 \leq i \leq n-1$ et $0 \leq j \leq m-1$).

Le jeu des récompenses se joue en k tours, numérotés de 0 à $k-1$. Chaque tour de jeu se déroule dans l'ordre suivant :

- Ringo choisit un **ensemble** de n tickets dans son sac, un de chaque couleur. Il les donne alors au maître du jeu.
- Le maître du jeu note les entiers $a[0]$, $a[1]$ $\ldots$ $a[n-1]$ imprimés sur les tickets de l'ensemble. L'ordre de ces n entiers n'est pas important.
- Le maître du jeu prend alors une carte de la boîte de tirage au sort, et note l'entier imprimé sur cette carte.
- Le maître du jeu calcule la valeur absolue de la différence entre $a[i]$ et b, pour chaque i de 0 à $n-1$. Soit S la somme des valeurs absolues des différences.
- Pour ce tour, le maître du jeu donne à Ringo un prix dont la valeur est égale à S.
- Les tickets de l'ensemble sont écartés du jeu, et ne peuvent pas être réutilisés lors des tours suivants.

Les tickets restant dans le sac de Ringo à la fin des k tours de jeu ne seront pas utilisés.

En regardant de près, Ringo a découvert que le jeu est truqué ! Il y a en fait une imprimante dans la boîte de tirage au sort. À chaque tour, le maître du jeu choisit un entier b qui minimise la valeur du prix de ce tour. La valeur choisie par le maître du jeu est alors imprimée sur la carte de ce tour.

À partir de ces informations, Ringo souhaite affecter les tickets aux tours de jeu. Plus précisément, il veut choisir l'ensemble de tickets joué chaque tour, afin de maximiser la valeur totale des prix obtenus.

Détails d'implémentation

Vous devez implémenter la fonction suivante :

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

- k : le nombre de tours de jeu.
- x : un tableau de taille $n \times m$ décrivant les entiers sur chaque ticket. Les tickets de chaque couleur sont triés par ordre croissant.
- Cette fonction est appelée exactement une fois.
- Cette fonction doit faire exactement un appel à `allocate_tickets` (voir ci-dessous), décrivant les k ensembles de tickets des k tours de jeu. Cette allocation doit maximiser la valeur totale des prix.
- Cette fonction doit renvoyer la valeur totale maximale des prix.

La fonction `allocate_tickets` est définie de la manière suivante :

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

- s : un tableau de taille $n \times m$. La valeur de $s[i][j]$ doit être égale à r si le ticket j de la couleur i doit être joué lors du tour de jeu r, ou égale à -1 s'il n'est pas utilisé.
- Pour chaque $0 \leq i \leq n-1$, parmi $s[i][0], s[i][1], \ldots, s[i][m-1]$ chaque valeur $0, 1, 2, \ldots, k-1$ doit apparaître exactement une fois, et toutes les autres valeurs doivent être -1.
- S'il existe plusieurs affectations qui maximisent la valeur totale des prix, chacune d'entre elles sera autorisée.

Exemples

Exemple 1

Considérons l'appel suivant :

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

Cela signifie que :

- il y a $k = 2$ tours de jeu ;
- les entiers imprimés sur les tickets de la couleur 0 sont 0, 2 et 5 ;
- les entiers imprimés sur les tickets de la couleur 1 sont 1, 1 et 3.

Une affectation possible pour maximiser la valeur totale des prix est :

- Au tour 0, Ringo choisit le ticket 0 de la couleur 0 (avec l'entier 0) et le ticket 2 de la couleur 1 (avec l'entier 3). Le plus petit prix pour ce tour a une valeur de 3. Le maître du jeu peut par exemple choisir $b = 1$: $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- Au tour 1, Ringo choisit le ticket 2 de la couleur 0 (avec l'entier 5) et le ticket 1 de la couleur 1 (avec l'entier 1). Le plus petit prix pour ce tour a une valeur de 4. Le maître du jeu peut par exemple choisir $b = 3$: $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- Par conséquent, la valeur totale des prix sera de $3 + 4 = 7$.

Pour renvoyer cette affectation, la fonction `find_maximum` doit faire l'appel suivant à la fonction `allocate_tickets` :

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

Enfin, la procédure `find_maximum` doit renvoyer la valeur 7.

Exemple 2

Considérons l'appel suivant :

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

Cela signifie que :

- il y a un seul tour de jeu,
- les entiers imprimés sur les tickets de la couleur 0 sont 5 et 9.
- les entiers imprimés sur les tickets de la couleur 1 sont 1 et 4.
- les entiers imprimés sur les tickets de la couleur 2 sont 3 et 6.
- les entiers imprimés sur les tickets de la couleur 3 sont 2 et 7.

Une affectation possible pour maximiser la valeur totale des prix est :

- Au tour 0, Ringo choisit le ticket 1 de la couleur 0 (avec l'entier 9), le ticket 0 de la couleur 1 (avec l'entier 1), le ticket 0 de la couleur 2 (avec l'entier 3), et le ticket 1 de la couleur 3 (avec l'entier 7). Le plus petit prix pour ce tour a une valeur de 12. Le maître du jeu peut par exemple choisir $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

Pour renvoyer cette affectation, la fonction `find_maximum` doit faire l'appel suivant à la fonction `allocate_tickets` :

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

Enfin, la procédure `find_maximum` doit renvoyer la valeur 12.

Contraintes

- $2 \leq n \leq 1500$ et n est pair.
- $1 \leq k \leq m \leq 1500$
- $0 \leq x[i][j] \leq 10^9$ (pour tous $0 \leq i \leq n - 1$ et $0 \leq j \leq m - 1$)
- $x[i][j - 1] \leq x[i][j]$ (pour tous $0 \leq i \leq n - 1$ et $1 \leq j \leq m - 1$)

Sous-tâches

1. (11 points) $m = 1$
2. (16 points) $k = 1$

3. (14 points) $0 \leq x[i][j] \leq 1$ (pour tous $0 \leq i \leq n-1$ et $0 \leq j \leq m-1$)
4. (14 points) $k = m$
5. (12 points) $n, m \leq 80$
6. (23 points) $n, m \leq 300$
7. (10 points) Aucune contrainte supplémentaire.

Évaluateur d'exemple

L'évaluateur d'exemple lit l'entrée au format suivant :

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

L'évaluateur d'exemple écrit votre réponse au format suivant :

- ligne 1 : la valeur revnoyée par `find_maximum`
- ligne $2 + i$ ($0 \leq i \leq n-1$) : $s[i][0]$ $s[i][1]$ $\ldots$ $s[i][m-1]$