Carnival Tickets (tickets)

Ringo este la un carnaval în Singapore. El are câteva bilete în bagaj, pe care vrea să le folosească la un stand cu jocuri. Fiecare bilet are o culoare din cele n culori posibile și are un întreg nenegativ imprimat pe el. Întregul imprimat poate fi același pentru mai multe dintre bilete. Datorită unei ciudățenii în regulile carnavalului, n este garantat **par**.

Ringo are câte m bilete de fiecare culoare în bagaj, pentru un total de $n \cdot m$ bilete. Biletul j de culoare i are întregul $x[i][j]$ imprimat pe el ($0 \leq i \leq n-1$ și $0 \leq j \leq m-1$).

Jocul se joacă în k runde, numerotate de la 0 la $k-1$. Fiecare rundă este jucată în ordinea următoare:

- Din bagaj, Ringo selectează o **mulțime** de n bilete, un bilet din fiecare culoare. El dă apoi biletele responsabilului de la stand.
- Responsabilul standului notează întregii $a[0]$, $a[1]$ $\ldots$ $a[n-1]$ imprimați pe biletele din mulțime. Ordinea acestor n întregi nu este importantă.
- Responsabilul standului extrage o carte specială dintr-o cutie și notează întregul b imprimat pe această carte.
- Responsabilul standului calculează modulul diferenței dintre $a[i]$ și b pentru fiecare i de la 0 la $n-1$. Fie S suma acestor module.
- Pentru această rundă, responsabilul standului îi dă lui Ringo un premiu în valoare egală cu S.
- Biletele din mulțime sunt aruncate și nu mai pot fi folosite în rundele următoare.

Biletele rămase în bagajul lui Ringo după k runde ale jocului sunt aruncate.

Privind cu atenție, Ringo realizează că jocul este trucat! Există de fapt o imprimantă în interiorul cutiei din care se extrage cartea specială. În fiecare rundă, responsabilul standului găsește un întreg b care minimizează valoarea premiului pentru acea rundă. Valoarea aleasă de responsabilul standului este imprimată pe cartea extrasă în acea rundă.

Având toate aceste informații, Ringo ar dori să aloce biletele pentru rundele jocului. Mai exact, acesta dorește să selecteze mulțimea de bilete folosită în fiecare rundă astfel încât să maximizeze valoarea totală a premiilor.

Detalii de implementare

Trebuie să implementați următoarea funcție:

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

- k: numărul rundelor.
- x: o matrice de dimensiuni $n \times m$ conținând întregii de pe bilete. Biletele de fiecare culoare sunt sortate în ordinea crescătoare a întregilor imprimați pe ele.
- Această funcție este apelată o singură dată.
- Această funcție trebuie să facă un singur apel către `allocate_tickets` (vezi mai jos), pentru k mulțimi de bilete, unul pentru fiecare rundă.
- Această funcție trebuie să returneze valoarea maximă a premiilor.

Funcția `allocate_tickets` este definită astfel:

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

- s: o matrice de dimensiuni $n \times m$. Valoarea lui $s[i][j]$ este r dacă biletul j de culoare i este utilizat în mulțimea rundei r a jocului, sau -1 dacă nu este folosit deloc.
- Pentru fiecare $0 \leq i \leq n-1$, printre $s[i][0], s[i][1], \ldots, s[i][m-1]$ fiecare valoare $0, 1, 2, \ldots, k-1$ trebuie să apară o singură dată, iar celelalte valori să fie -1.
- Dacă există mai multe alocări pentru valoarea maximă a premiului, se poate raporta oricare.

Exemple

Exemplul 1

Considerăm următorul apel:

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

Asta înseamnă că:

- sunt $k = 2$ runde;
- întregii imprimați pe biletele de culoare 0 sunt 0, 2 și 5;
- îngtregii imprimați pe biletele de culoare 1 sunt 1, 1, și 3.

O posibilă alocare este:

- În runda 0, Ringo alege biletul 0 de culoare 0 (cu întregul 0) și biletul 2 de culoare 1 (cu întregul 3). Cea mai mică valoare a premiului în această rundă este 3. E.g., responsabilul jocului ar putea alege $b = 1$: $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- În runda 1, Ringo alege biletul 2 de culoare 0 (cu întregul 5) și biletul 1 de culoare 1 (cu întregul 1). Cea mai mică valoare a premiului în această rundă este 4. E.g., responsabilul jocului ar putea alege $b = 3$: $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- Astfel, valoarea totală a premiilor ar fi $3 + 4 = 7$.

Pentru a raporta această alocare, funcția `find_maximum` ar trebui să facă următorul apel către `allocate_tickets`:

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

În final, funcția `find_maximum` trebuie să returneze 7.

Exemplul 2

Cosiderăm următorul apel:

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

Asta înseamnă că:

- există o singură rundă,
- întregii de pe culoarea 0 sunt 5 și 9;
- întregii de pe culoarea 1 sunt 1 și 4;
- întregii de pe culoarea 2 sunt 3 și 6;
- întregii de pe culoarea 3 sunt 2 și 7.

O posibilă alocare este:

- În runda 0, Ringo alege biletul 1 de culoare 0 (cu întregul 9), biletul 0 de culoare 1 (cu întregul 1), biletul 0 de culoare 2 (cu întregul 3), și biletul 1 de culoare 3 (cu întregul 7). Cea mai mică valoare a premiului în această rundă este 12, când responsabilul jocului alege $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

Pentru a raporta această alocare, funcția `find_maximum` ar trebui să facă următorul apel către `allocate_tickets`:

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

În final, funcția `find_maximum` trebuie să returneze 12.

Restrictii

- $2 \leq n \leq 1500$ și n este par.
- $1 \leq k \leq m \leq 1500$
- $0 \leq x[i][j] \leq 10^9$ (oricare ar fi $0 \leq i \leq n - 1$ si $0 \leq j \leq m - 1$)
- $x[i][j - 1] \leq x[i][j]$ (oricare ar fi $0 \leq i \leq n - 1$ si $1 \leq j \leq m - 1$)

Subtaskuri

1. (11 puncte) $m = 1$
2. (16 puncte) $k = 1$
3. (14 puncte) $0 \leq x[i][j] \leq 1$ (oricare ar fi $0 \leq i \leq n - 1$ și $0 \leq j \leq m - 1$)
4. (14 puncte) $k = m$
5. (12 puncte) $n, m \leq 80$

6. (23 puncte) $n, m \leq 300$
7. (10 puncte) Fără restricții suplimentare.

Sample grader

Sample graderul citește intrările în următorul format:

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

Sample graderul afișează răspunsul tău în următorul format:

- linia 1: valoarea returnata de `find_maximum`
- linia $2 + i$ ($0 \leq i \leq n - 1$): $s[i][0]$ $s[i][1]$ $\ldots$ $s[i][m - 1]$