Carnival Tickets (tickets)

Ringo je na karnevalu u Singapuru. U torbi ima neke listiće lutrije koje bi želeo da iskoristi u nagradnoj igri Svaki listić dolazi u jednoj od n boja i na njemu je odštampan nenegativan cio broj. Cijeli brojevi odštampani na različitim listićima mogu biti isti. Zbog nekih čudnih karnevalskih pravila, n je garantovano **paran**.

Ringo u svojoj torbi ima m listića lutrije od svake boje, što je ukupno $n \cdot m$ listića. Na listiću j boje i odštampan je cio broj $x[i][j]$ ($0 \leq i \leq n-1$ i $0 \leq j \leq m-1$).

Nagradna igra se odvija u k rundi, označenih brojevima od 0 do $k-1$. Svaka runda se igra sljedećim redosledom::

- Ringo iz svoje torbe bira **skup** od n listića, po jedan listić od svake boje. Zatim daje izabrani skup organizatoru igre.
- Organizator igre bilježi cijele brojeve $a[0]$, $a[1]$ $\ldots$ $a[n-1]$ odštampane na kompletima listića. Redosled ovih n cijelih brojeva nije važan.
- Organizator igre izvlači listić iz bubnja i bilježi broj b odštampan na njemu.
- Organizator igre izračunava apsolutne razlike između $a[i]$ i b za svaki i od 0 do $n-1$. Neka je S zbir ovih apsolutnih razlika.
- Za ovu rundu, oganizator igre daje Ringa nagradu u vrijednosti jednakoj S.
- Listići u kompletu se odbacuju i ne mogu se koristiti u narednim rundama.

Preostale karte u Ringovoj torbi nakon k rundi igre se odbacuju.

Posmatrajući pažljivo, Ringo je shvatio da je nagradna igra namještena! Unutar bubnja za izvlačenje zapravo postoji štampač. U svakoj rundi, organizator igre određuje cio broj b koji minimizuje vrijednost nagrade u toj rundi. Vrijednost koju je odabrao organizator igre štampa se na posebnom listiću za tu rundu.

Znajući sve ovo, Ringo želi da odabere listiće za sljedeće runde igre. To jest, on želi da izabere skupove listića koje će koristiti u svakoj rundi da bi maksimizovao ukupnu vrijednost nagrada.

Detalji implementacije

Potrebno je implementirati sljedeće funkcije:

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

- k: broj rundi.
- x: niz cijelih brojeva dimenzija $n \times m$ koji opisuje cijele brojeve na listićima. Listići svake boje

su sortirani u neopadajućem poretku brojeva na njima.
- Ova funkcija se poziva tačno jednom,.
- Ova funkcija će pozvati funkciju `allocate_tickets` (vidi ispod) tačno jednom, opisujući k skupova listića, po jedan za svaku rundu. Izbor skupova treba da maksimizuje ukupnu vrijednst nagrade.
- Funkcija vraća maksimalnu ukupnu vrijednost nagrada.

Funkcija `allocate_tickets` je definisana na sljedeći način:

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

- s: niz dimenzija $n \times m$. Vrijednost $s[i][j]$ treba da je r ako je listić j boje i upotrebljen u rundi r igre ili -1 ako uopšte nije upotrebljen.
- Za svako $0 \leq i \leq n-1$, unutar $s[i][0], s[i][1], \ldots, s[i][m-1]$ svaka od vrijednosti $0, 1, 2, \ldots, k-1$ mora se pojaviti tačno jednom dok sve ostale vrijednosti moraju biti -1.
- Ako postoji više izbora listića koji daju maksimanu ukupnu nagradu, možete izabrati bilo koju.

Primjeri

Primjer 1

Posmatrajmo sljedeći poziv funkcije:

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

Ovo znači da:

- postoje $k = 2$ runde;
- brojevi na listićima boje 0 su redom 0, 2 i 5;
- brojevi na listićima boje 1 su redom 1, 1 and 3.

Mogući izbor listića koji daje maksimalnu ukupnu nagradu je:

- U rundi 0, Ringo bira listić 0 boje 0 (sa brojem 0) i listić 2 boje 1 (sa brojem 3). Najmanja moguća vrijednost nagrade u ovoj rundi je 3. Organizator igre može izabrati $b = 1$: $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- U rundi 1, Ringo bira listić 2 boje 0 (sa brojem 5) i listić 1 boje 1 (sa brojem 1).Najmanja moguća vrijednost nagrade u ovoj rundi je 4. Organizator igre može izabrati $b = 3$: $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- Dakle, ukupna vrijednost nagrada biće $3 + 4 = 7$.

Da bi izabrali ovaj skup listića, funkcija `find_maximum` pozvaće funkciju `allocate_tickets` na sljedeći način:

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

Konačno, funkcija `find_maximum` će vratiti 7.

Primjer 2

Posmatrajmo sljedeći poziv funkcije:

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

Ovo znači da:

- postoji samo jedna runda,
- brojevi na listićima boje 0 su redom 5 i 9;
- brojevi na listićima boje 1 su redom 1 i 4;
- brojevi na listićima boje 2 su redom 3 i 6;
- brojevi na listićima boje 3 su redom 2 i 7.

Mogući izbor listića koji daje maksimalnu ukupnu nagradu je:

- U rundi 0, Ringo bira listić 1 boje 0 (sa brojem 9), listić 0 boje 1 (sa brojem 1), listić 0 boje 2 (sa brojem 3) i listić 1 boje 3 (sa brojem 7). Najmanja moguća vrijednost nagrade u ovoj rundi je 12, kada organizator igre izabere $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

Da bi izabrali ovaj skup listića, funkcija `find_maximum` pozvaće funkciju `allocate_tickets` na sljedeći način:

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

Konačno, funkcija `find_maximum` će vratiti 12.

Ograničenja

- $2 \leq n \leq 1500$ i n je paran
- $1 \leq k \leq m \leq 1500$
- $0 \leq x[i][j] \leq 10^9$ (za sve $0 \leq i \leq n - 1$ i $0 \leq j \leq m - 1$)
- $x[i][j - 1] \leq x[i][j]$ (za sve $0 \leq i \leq n - 1$ i $1 \leq j \leq m - 1$)

Podzadaci

1. (11 bodova) $m = 1$
2. (16 bodova) $k = 1$
3. (14 bodova) $0 \leq x[i][j] \leq 1$ (za sve $0 \leq i \leq n - 1$ i $0 \leq j \leq m - 1$)
4. (14 bodova) $k = m$
5. (12 bodova) $n, m \leq 80$
6. (23 boda) $n, m \leq 300$
7. (10 bodova) Nema dodatnih ograničenja.

Program za testiranje (grader)

Program za testiranje (grader) čita ulaz u sljedećem formatu:

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

Program za testiranje (grader) štampa rezultat u sljedećem formatu::

- red 1: vrijednost kuju vraća funkcija `find_maximum`
- red $2+i$ ($0 \leq i \leq n-1$): $s[i][0] \;\; s[i][1] \;\; \ldots \;\; s[i][m-1]$