Karnevalové lístky (tickets)

Keď sa v Singapure koná karneval, môžu návštevníci v rôznych hrách získať rôznofarebné lístky. Lístky sa potom v jednom stánku dajú vymeniť za rôzne ceny. Každý lístok má jednu z n možných farieb. Na každom lístku je vytlačené nejaké nezáporné celé číslo. Z obskurných dôvodov je zaručené, že číslo n je **párne**.

Miškovi sa podarilo z rôznych hier pozbierať dokopy $n \cdot m$ lístkov: z každej farby presne m kusov. Lístok farby i s indexom j má na sebe číslo $x[i][j]$.

Výherná sada lístkov je ľubovoľných n lístkov, ktoré majú navzájom rôzne farby. V stánku s cenami za ľubovoľnú výhernú sadu lístkov dostanete jednu cenu. Výdaj ceny prebieha nasledovne:

- Pracovník v stánku od vás zoberie n lístkov tvoriacich výhernú sadu a poznačí si čísla $a[0], a[1] \ \ldots \ a[n-1]$, ktoré na nich boli vytlačené.
- Pracovník v stánku siahne do svojej magickej krabičky, vytiahne z nej kartičku a poznačí si číslo b, ktoré je na nej vytlačené.
- Pre každé i od 0 po $n-1$ pracovník v stánku vypočíta absolútnu hodnotu rozdielu medzi $a[i]$ a b.
- Cena, ktorú hráč dostane, má hodnotu rovnú súčtu týchto absolútnych hodnôt.
- Pracovník znehodnotí celú výhernú sadu lístkov, aby sa tieto lístky už nedali opäť použiť.

Miško chvíľu sledoval výdaj cien iným zákazníkom. Netrvalo dlho a pochopil, ako to skutočne funguje. Zákazníci si myslia, že kartičku z magickej krabičky pracovník náhodne vyžrebuje, to ale vôbec nie je pravda! V krabičke je v skutočnosti malička tlačiareň a tá vždy na kartičku vytlačí také b, pre ktoré zákazník dostane cenu s najmenšou možnou hodnotou.

Miško má v batohu miesto na k cien. Chcel by si preto za časť svojich lístkov postupne vyzdvihnúť k cien, a to tak, aby ich celková hodnota bola najväčšia možná. (Nepoužité lístky potom zahodí.)

Implementation details

Naprogramujte nasledujúcu funkciu:

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

- k: počet cien, ktoré chceme.
- x: pole rozmerov $n \times m$ obsahujúce čísla vytlačené na lístkoch. Pre každú farbu platí, že čísla na lístkoch tejto farby budú uvedené v neklesajúcom poradí.
- Túto funkciu grader zavolá práve raz.

- Na výstupe má táto funkcia vrátiť najväčšiu možnú celkovú hodnotu k cien, ktoré vie Miško za svoje lístky získať.
- Skôr, ako túto hodnotu vráti, musí vaša funkcia práve raz zavolať funkciu gradera nazvanú `allocate_tickets` (viď nižšie) a oznámiť tak graderu, aké výherné sady lístkov má Miško poskladať, aby maximalizoval svoj zisk.

Funkcia gradera `allocate_tickets` vyzerá nasledovne:

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

- s: pole rozmerov $n \times m$, v ktorom číslo $s[i][j]$ predstavuje poradové číslo výhernej sady, do ktorej má patriť lístok farby i s indexom j.
- Jednotlivé sady majú čísla od 0 po $k-1$. Hodnota -1 zodpovedá tomu, že tento lístok nepoužijeme.
- Pre každé i $(0 \leq i \leq n-1)$ musí platiť, že medzi hodnotami $s[i][0], s[i][1], \ldots, s[i][m-1]$ sa každá z hodnôt $0, 1, 2, \ldots, k-1$ nachádza práve raz. Všetky ostatné hodnoty musia byť rovné -1.
- Ak je optimálnych riešení viac, môžete oznámiť ľubovoľné z nich.

Examples

Example 1

Grader zavolal vašu funkciu nasledovne:

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

Dozvedeli sme sa, že:

- Miško chce 2 ceny,
- lístky farby 0 majú na sebe čísla 0, 2 a 5,
- lístky farby 1 majú na sebe čísla 1, 1 a 3.

Jedno možné optimálne riešenie vyzerá nasledovne:

- Do prvej výhernej sady dá Miško z farby 0 lístok s indexom 0 (má na sebe číslo 0) a z farby 1 lístok s indexom 2 (ten má na sebe číslo 3).
- Cena, ktorú za ne dostane, bude mať hodnotu 3. Napríklad ak z krabičky vyjde kartička s $b = 1$, bude hodnota Miškovej ceny rovná $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- Do druhej výhernej sady dá Miško z farby 0 lístok s indexom 2 (má na sebe číslo 5) a z farby 1 lístok s indexom 1 (ten má na sebe číslo 1).
- Cena, ktorú za ne dostane, bude mať hodnotu 4. Napríklad pre $b = 3$ dostávame $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- Celková hodnota cien, ktoré Miško takto získal, je 7.

Vaša funkcia musí najskôr nahlásiť obe sady lístkov. To spraví tak, že zavolá funkciu gradera `allocate_tickets` nasledovne:

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

Následne vaša funkcia skončí a vráti 7.

Example 2

Grader zavolal vašu funkciu nasledovne:

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

Tentokrát sme sa dozvedeli, že:

- Miško chce len 1 cenu,
- lístky farby 0 majú na sebe čísla 5 a 9,
- lístky farby 1 majú na sebe čísla 1 a 4,
- lístky farby 2 majú na sebe čísla 3 a 6,
- lístky farby 3 majú na sebe čísla 2 a 7.

Jedna možná výherná sada lístkov, pre ktorú Miško dostane cenu najväčšej hodnoty, je:

- z farby 0 lístok s indexom 1 (číslo 9),
- z farby 1 lístok s indexom 0 (číslo 1),
- z farby 2 lístok s indexom 0 (číslo 3),
- z farby 3 lístok s indexom 1 (číslo 7).

Keď Miško odovzdá tieto lístky, krabička môže na kartičku vytlačiť napríklad číslo $b = 3$. Hodnota Miškovej ceny bude potom $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

Vyššie popísané riešenie nahlásite graderu nasledovným volaním `allocate_tickets`:

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

Na záver má vaša funkcia vrátiť hodnotu 12.

Constraints

- $2 \leq n \leq 1500$
- n je párne
- $1 \leq k \leq m \leq 1500$
- $0 \leq x[i][j] \leq 10^9$ (pre $0 \leq i \leq n-1, 0 \leq j \leq m-1$)
- $x[i][j-1] \leq x[i][j]$ (pre $0 \leq i \leq n-1, 1 \leq j \leq m-1$)

Subtasks

1. (11 points) $m = 1$
2. (16 points) $k = 1$
3. (14 points) $0 \leq x[i][j] \leq 1$ (pre $0 \leq i \leq n-1, 0 \leq j \leq m-1$)
4. (14 points) $k = m$
5. (12 points) $n, m \leq 80$
6. (23 points) $n, m \leq 300$
7. (10 points) Bez ďalších obmedzení.

Sample grader

Ukážkový grader očakáva vstup v nasledujúcom formáte:

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

Výstup z ukážkového gradera:

- line 1: návratová hodnota `find_maximum`
- line $2 + i$ ($0 \leq i \leq n-1$): $s[i][0]$ $s[i][1]$ $\ldots$ $s[i][m-1]$