

Гра в школі

Відома середня школа в Ташкенті святкує свою річницю грою між учнями та вчителями. У класі сидять N учнів, пронумерованих від 0 до $N - 1$. Кожен учень тримає аркуш паперу, який містить масив цілих чисел. Спочатку кожен аркуш порожній, тому він містить порожній масив.

Учні грають у гру з M вчителями, пронумерованими від 0 до $M - 1$, та директором. Гра проводиться у M кроків, які також пронумеровані від 0 до $M - 1$. На кроці j вчитель j входить до класу, і відбуваються такі події:

- Деякі (можливо, нуль) учні піднімають руку. Гарантується, що на кожному кроці хоча б один учень **не** піднімає руку, і кожен учень може підняти руку **не більше одного разу** протягом усієї гри.
- Вчитель переглядає аркуші, які наразі тримають учні, і може **змінити** аркуші учнів, які **не** підняли руку на цьому кроці. Для кожного такого учня вчитель може замінити вміст його аркуша новим (можливо, порожнім) масивом цілих чисел. Кожне ціле число має бути від 0 до 63 включно, а масив може містити щонайбільше 63 елементи.
- Після цих модифікацій вчитель j йде, а учні обмінюються своїми аркушами згідно з секретною перестановкою P . Тобто, кожен учень i ($0 \leq i < N$) віддає свій аркуш учню $P[i]$, де $P[0], P[1], \dots, P[N - 1]$ це N **різних** цілих чисел від 0 до $N - 1$ включно. Значення P **фіксовані** протягом усіх кроків гри, але вчителі та директор їх не знають.

Після завершення всіх M кроків та виконання останнього обміну згідно з P , входить директор. Дивлячись лише на аркуші, що тримають учні, директор повинен визначити для кожного учня i ($0 \leq i < N$) номер кроку j , на якому учень i підняв руку, або зробити висновок, що учень i ніколи не піднімав руку.

Вчителі не можуть спілкуватися з іншими вчителями або директором, окрім як через масиви цілих чисел, написаних на аркуші. Кожен вчитель знає, на якому етапі він входить до класу.

Ваше завдання — розробити та імплементувати стратегію для вчителів і директора, яка правильно визначатиме, чи кожен учень підняв руку і коли. Ваш бал залежатиме від максимальної довжини будь-якого масиву, написаного на аркушах: менша максимальна довжина дає вищий або рівний бал.

Деталі реалізації

Вам потрібно імплементувати дві процедури: одну для вчителів, а іншу для директора.

Процедура, яку ви повинні імплементувати для **вчителів**, така:

```
std::vector<std::vector<int>> process_step(  
    int N, int M, int R,  
    std::vector<int> T,  
    std::vector<std::vector<int>> A)
```

- N : кількість учнів.
- M : кількість кроків, а також кількість вчителів.
- R : номер поточного кроку (від 0 до $M - 1$).
- T : (можливо, порожній) масив, що складається з індексів учнів, які підняли руку на цьому кроці, відсортованих у порядку зростання.
- A : масив довжини N , що описує аркуші, де $A[i]$ ($0 \leq i < N$) – це масив цілих чисел на аркуші, який мав учень i на початку кроку.
- Ця процедура викликається рівно M разів за гру, у порядку $R = 0, 1, \dots, M - 1$.

Процедура повинна повертати масив B довжиною N , що представляє аркуші після модифікацій вчителя, у тому ж форматі, що й A .

- Для кожного учня i , який підняв руку (тобто i з'являється в T), аркуш не повинен змінюватися, тому $B[i] = A[i]$.
- Для кожного i такого, що $0 \leq i < N$, довжина $B[i]$ не повинна перевищувати 63, а кожне ціле число в $B[i]$ має бути між 0 та 63 включно.

Процедура, яку ви повинні імплементувати для **директора**, така:

```
std::vector<int> determine_steps(  
    int N, int M, std::vector<std::vector<int>> A)
```

- N, M : те саме, що й вище.
- A : масив довжини N , де $A[i]$ – масив цілих чисел на аркуші паперу, який має учень i після всіх M кроків.
- Ця процедура викликається рівно один раз за гру, після останнього виклику `process_step`.

Процедура повинна повертати масив D довжини N . Для кожного i такого, що $0 \leq i < N$, має виконуватися умова

- $D[i] = j$, якщо учень i підняв руку на кроці j , або
- $D[i] = -1$, якщо учень i жодного разу не підняв руку протягом усієї гри.

Вашій програмі не дозволено зберігати або передавати будь-яку інформацію з одного виклику `process_step` до іншого, окрім як через повернене значення `process_step`. Якщо ваша програма спробує це зробити, ваш бал у CMS може бути неправильним і може

бути зменшений після закінчення змагання. Зверніть увагу, що градер може виконувати кілька екземплярів вашої програми одночасно. Це означає, що виклики `process_step` та `determine_steps` з однієї гри можуть виконуватися в різних екземплярах, а виклики `process_step` та `determine_steps` з різних ігор можуть виконуватися в одному екземплярі. Кожен тестовий випадок містить до 5 ігор.

Обмеження

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Кожен учень піднімає руку **не більше одного разу** протягом усієї гри. Іншими словами, для кожного учня i існує максимум один крок j , на якому i піднімає руку.
- На кожному кроці принаймні один учень **не** піднімає руку.

Підзадачі та оцінювання

Підзадача	Бали	Додаткові обмеження
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ для всіх $0 \leq i \leq N - 1$.
4	25	На кожному кроці руку піднімає максимум один учень.
5	56	Без додаткових обмежень.

Для кожного тестового випадку ваш бал становить 0 (повідомляється як `Output isn't correct` в CMS), якщо повернене значення будь-якого виклику процедури `process_step` є недійсним або якщо повернене значення будь-якого виклику процедури `determine_steps` є неправильним.

В іншому випадку, нехай C буде максимальною довжиною **будь-якого** масиву в **будь-якому** B повернутому `process_step`. Тоді оцінка для тестового випадку в підзадачі з оцінкою S розраховується як $S \cdot X$, де X залежить від C згідно з наступною таблицею:

Умова	X
$C \leq 2$	1.00
$C = 3$	0.75
$C = 4$	0.55
$5 \leq C \leq 13$	$0.50 - 0.03 \cdot (C - 5)$
$14 \leq C \leq 63$	$0.19 \cdot \frac{64-C}{64-14} + 0.04$

Приклад

Розглянемо сценарій з $N = 4$ учнями, $M = 2$ вчителями та перестановкою $P = [0, 3, 1, 2]$. Спочатку всі аркуші папери порожні.

Спочатку градер викликає:

```
process_step(4, 2, 0, [1], [[], [], [], []])
```

Учень 1 підняв руку, тому його аркуш не можна змінити. Вчитель 0 може вирішити написати [10] на аркуші учня 0, [1, 63, 4] на аркуші учня 3, а аркуш учня 2 залишити порожнім. Для цього процедура повинна повернути $B = [[10], [], [], [1, 63, 4]]$.

Після того, як вчитель 0 йде, учні обмінюються своїми аркушами згідно з перестановкою P . Після обміну аркуші мають вигляд $A = [[10], [], [1, 63, 4], []]$.

Тепер градер викликає:

```
process_step(4, 2, 1, [0, 3], [[10], [], [1, 63, 4], []])
```

Учні 0 та 3 підняли руку, тому їхні аркуші не можна змінити. Вчитель 1 може вирішити написати [0, 40] на аркуші учня 1 та [1, 50] на аркуші учня 2. Для цього процедура повинна повернути $B = [[10], [0, 40], [1, 50], []]$.

Після того, як вчитель 1 йде, аркуші знову обмінюються відповідно до P . Після обміну аркуші мають вигляд $A = [[10], [1, 50], [], [0, 40]]$.

Нарешті, градер викликає:

```
determine_steps(4, 2, [[10], [1, 50], [], [0, 40]])
```

Ця процедура має повернути масив $D = [1, 0, -1, 1]$ оскільки учні 0 та 3 підняли руку на кроці 1, учень 1 підняв руку на кроці 0, а учень 2 ніколи не піднімав руку. У цьому прикладі $C = 3$.

Приклад градера

Формат вхідних даних:

```
N M
Q[0] Q[1] ... Q[N-1]
P[0] P[1] ... P[N-1]
```

Тут Q — це масив довжини N , де $Q[i] = j$ якщо учень i піднімає руку на кроці j , або $Q[i] = -1$ якщо учень i жодного разу не піднімає руку протягом усієї гри.

Формат вихідних даних:

Після кожного виклику `process_step`, градер виводить вміст аркушів. Нехай K — довжина B , а $L[i]$ — довжина $B[i]$ (для кожного $0 \leq i < K$). Приклад градера виводить B у такому форматі, **за яким йде порожній рядок**:

```
K
L[0] B[0][0] B[0][1] ... B[0][L[0]-1]
L[1] B[1][0] B[1][1] ... B[1][L[1]-1]
:
L[K-1] B[K-1][0] B[K-1][1] ... B[K-1][L[K-1]-1]
```

Потім градер міняє місцями аркуші відповідно до перестановки P . Якщо $K \neq N$, градер виводить повідомлення про помилку та завершує роботу.

Після виклику `determine_steps`, градер виводить:

```
C H
D[0] D[1] ... D[H-1]
```

Тут H — це довжина масиву D повернутого процедурою `determine_steps`.