Игра в класната стая

Известно училище в Ташкент организира игра между ученици и учители, за да отпразнува своята годишнина. В стая има N ученици, номерирани от 0 до $N-1$. Всеки от тях държи лист хартия, съдържащ списък от числа. В началото, всеки лист е празен, т.е. списъците са празни.

Заедно с учениците, в играта участват и M учители, номерирани от 0 до $M-1$, както и директорът. Играта се играе на M стъпки, също номерирани от 0 до $M-1$. На стъпка j, учител с номер j влиза в стаята, при което се случват следните събития:

- Няколко (може и нула) ученици си вдигат ръцете. Гарантирано е, че на всяка стъпка поне един ученик **не** си вдига ръката, както и че всеки ученик може да си вдигне ръката **най-много веднъж** по време на цялата игра.
- Учителят поглежда листовете, които в момента учениците държат, и може да **промени** листовете на тези ученици, които **не** са си вдигнали ръката на текущата стъпка. За всеки такъв ученик, учителят може да смени съдържанието на листа му с нов (може и празен) списък от числа. Списъкът има най-много 63 елемента, като всяко число в него е между 0 и 63, включително.
- След направените промени, учител с номер j напуска стаята, а учениците разменят своите листове в съответствие с тайна пермутация P. По-конкретно, ученик с номер i ($0 \leq i < N$) подава листа си на ученик с номер $P[i]$, където $P[0], P[1], \ldots, P[N-1]$ са N **различни** цели числа между 0 и $N-1$, включително. Стойностите в P са **фиксирани** по време на цялата игра, но учителите и директорът не ги знаят.

След като всички M стъпки са изпълнени и последната размяна съгласно пермутацията P е извършена, директорът влиза в стаята. Разглеждайки само листовете, които учениците държат, директорът трябва да определи за всяко i ($0 \leq i < N$), номерът на стъпката j, при която ученикът с номер i си е вдигнал ръката, или да каже, че ученикът с номер i никога не си е вдигал ръката.

Учителите работят независимо един от друг; не комуникират, нито помежду си, нито с директора, освен чрез списъците от числа, написани на листовете. Всеки учител знае стъпката, на която влиза.

Вашата задача е да измислите и имплементирате стратегия за учителите и за директора, която правилно да определя дали и кога всеки ученик си е вдигнал ръката. Вашият резултат зависи от максималната дължина на списък, написан по време на играта: по-къса максимална дължина дава по-добър или същия резултат.

Детайли по имплементацията

Трябва да имплементирате две функции – една за учителите и една за директора.

Функцията, която трябва да имплементирате за **учителите**, е:

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

- N: броят ученици.
- M: броят стъпки, който също е и броят учители.
- R: номерът на текущата стъпка (от 0 до $M-1$).
- T: масив (може и празен), съдържащ номерата на учениците, които са си вдигнали ръката по време на тази стъпка, сортиран в нарастващ ред.
- A: масив с дължина N, описващ листовете, където $A[i]$ $(0 \leq i < N)$ е масив, описващ списъка от числа, които ученик с номер i държи в началото на текущата стъпка.
- Тази функция се извиква точно M пъти на игра, в реда $R = 0, 1, \ldots, M-1$.

Функцията трябва да върне масив B с дължина N, описващ листовете след промените на учителя в същия формат като A.

- За всяко i, за което ученикът с номер i си е вдигнал ръката (т.е. i се среща в T), листът му не трябва да бъде променен, т.е. $B[i] = A[i]$.
- За всяко i, такова че $0 \leq i < N$, дължината на $B[i]$ не трябва да надвишава 63, както и всяко число в $B[i]$ трябва да бъде между 0 и 63, включително.

Функцията, която трябва да имплементирате за **директора**, е:

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

- N, M: същите като по-горе.
- A: масив с дължина N, където $A[i]$ е масив, описващ списъка от числа, записан на листа на ученика с номер i след всичките M стъпки.
- Тази функция се извиква точно веднъж на игра – след последното извикване на `process_step`.

Функцията трябва да върне масив D с дължина N. За всяко i $(0 \leq i < N)$ трябва да бъде изпълнено:

- $D[i] = j$, ако ученикът с номер i си е вдигнал ръката на стъпка j, или
- $D[i] = -1$, ако ученикът с номер i не си е вдигал ръката по време на цялата игра.

Вашата програма не трябва да съхранява или да предава каквато и да е информация от едно извикване на `process_step` към друго, освен чрез връщаната стойност на `process_step`. Ако програмата ви се опита да направи това, резултатът ви в CMS може да бъде неправилен и да бъде понижен след края на състезанието. Имайте предвид, че системата за оценяване може да изпълнява едновременно няколко инстанции на вашата програма. Това означава, че извикванията към `process_step` и `determine_steps` от една и съща игра могат да бъдат изпълнени в различни инстанции, а извикванията към `process_step` и `determine_steps` от различни игри могат да бъдат изпълнени в една и съща инстанция. Всеки тест включва до 5 игри.

Ограничения

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Всеки ученик си вдига ръката *най-много веднъж* по време на цялата игра. С други думи, за всяко i има най-много една стъпка j, в която ученикът с номер i си вдига ръката.
- На всяка стъпка поне един ученик **не** си вдига ръката.

Подзадачи и оценяване

Подзадача	Точки	Допълнителни ограничения
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ за всяко $0 \leq i \leq N - 1$.
4	25	На всяка стъпка най-много един ученик си вдига ръката.
5	56	Няма допълнителни ограничения.

Ако върнатата стойност на кое да е извикване на някоя от функциите `process_step` или `determine_steps` е невалидно/неправилно ще получите 0 точки (отчетено като `Output isn't correct` в CMS).

В противен случай, нека C е максималната дължина на **някой** списък в **някое** B, върнато от `process_step`. Тогава, резултатът, който ще получите за съответния тест в подзадача, носеща S точки, се изчислява като $S \cdot X$, където X зависи от C по начина, описан в следната таблица:

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

Пример

Да разгледаме пример с $N = 4$ ученици, $M = 2$ учители и пермутация $P = [0, 3, 1, 2]$. В началото всички листове са празни.

Грейдърът първо прави следното извикване:

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

Ученикът с номер 1 си е вдигнал ръката, така че неговия лист не може да бъде променен. Учителят с номер 0 може да реши да напише $[10]$ на листа на ученика с номер 0, да напише $[1, 63, 4]$ на листа на ученика с номер 3, и да остави листа на ученика с номер 2 празен. За да направи това, функцията трябва да върне резултат $B = [[10], [\,], [\,], [1, 63, 4]]$.

След като учителят с номер 0 излезе от стаята, учениците си разменят своите листове, в съответствие с пермутацията P. След размяната листовете стават съответно $A = [[10], [\,], [1, 63, 4], [\,]]$.

След това грейдърът прави следното извикване:

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

Учениците с номера 0 и 3 са си вдигнали ръката, така че техните листове не могат да бъдат променяни. Учителят с номер 1 може да реши да запише $[0, 40]$ на листа на ученика с номер 1 и да запише $[1, 50]$ на листа на ученика с номер 2. За да направи това, функцията трябва да върне $B = [[10], [0, 40], [1, 50], [\,]]$.

След като учителят с номер 1 излезе от стаята, листовете отново биват разменени съгласно P. След размяната листовете стават съответно $A = [[10], [1, 50], [\,], [0, 40]]$.

Накрая, грейдърът прави следното извикване:

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

Функцията трябва да върне масива $D = [1, 0, -1, 1]$, тъй като учениците с номера 0 и 3 са си вдигнали ръцете на стъпка 1, ученикът с номер 1 си е вдигнал ръката на стъпка 0, а ученикът с номер 2 не си е вдигал ръката. В този пример $C = 3$.

Локален грейдър

Формат на входа:

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

Тук Q е масив с дължина N, където $Q[i] = j$, ако ученикът с номер i си е вдигнал ръката на стъпка j, или $Q[i] = -1$, ако ученикът с номер i не си е вдигал ръката по време на цялата игра.

Формат на изхода:

След всяко извикване на функцията `process_step`, примерният грейдър извежда съдържанието на листовете. Нека K е дължината на B и $L[i]$ е дължината на $B[i]$ (за всяко $0 \leq i < K$). Локалният грейдър извежда B в следния формат, последван от **празен ред**.

```
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]
```

Локалният грейдър след това разменя листовете, в съответствие с пермутацията P. Ако $K \neq N$, локалният грейдър извежда съобщения за грешка и приключва.

След извикване на `determine_steps` локалният грейдър извежда:

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

Тук H е дължината на масива D, върнат от функцията `determine_steps`.