

Classroom Game

Un liceu renumit din Taşkent își sărbătorește aniversarea găzduind un joc între elevi și profesori. În clasă stau N elevi, numerotați de la 0 la $N - 1$. Fiecare elev ține o foaie de hârtie în care poate fi înscris un tablou de numere întregi. Inițial, fiecare foaie este goală, deci conține un tablou vid.

Elevii joacă jocul cu M profesori, numerotați de la 0 la $M - 1$, și cu directorul. Jocul se desfășoară în M pași, numerotați de asemenea de la 0 la $M - 1$. În pasul j , profesorul j intră în clasă și au loc următoarele evenimente:

- Unii elevi (posibil zero) ridică mâna. Se garantează că la fiecare pas cel puțin un elev **nu** ridică mâna și fiecare elev își poate ridica mâna **cel mult o dată** pe parcursul întregului joc.
- Profesorul examinează foile de lucru pe care elevii le țin în mână în prezent și poate **modifica** foile de lucru deținute de elevii care **nu** au ridicat mâna în acest pas. Pentru fiecare astfel de elev, profesorul poate înlocui conținutul foi sale de lucru cu un nou tablou (posibil vid) de numere întregi. Fiecare număr întreg trebuie să fie între 0 și 63, inclusiv, iar tabloul poate avea cel mult 63 de elemente.
- După aceste modificări, profesorul j pleacă, iar elevii își schimbă lucrările conform unei permutări secrete P . Adică, fiecare elev i ($0 \leq i < N$) își dă lucrarea elevului $P[i]$, unde $P[0], P[1], \dots, P[N - 1]$ sunt N numere întregi **distincte** între 0 și $N - 1$, inclusiv. Valorile lui P sunt **fixe** pe parcursul tuturor etapelor jocului, însă profesorii și directorul nu le cunosc.

Odată ce toți M pași s-au finalizat și ultima interschimbare conform lui P a fost efectuată, directorul intră în scenă. Uitându-se doar la hârtiile deținute de elevi, directorul trebuie să determine, pentru fiecare elev i ($0 \leq i < N$), numărul pasului j în care elevul i a ridicat mâna sau să concluzioneze că elevul i nu a ridicat niciodată mâna.

Profesorii nu pot comunica cu alți profesori sau cu directorul, decât prin intermediul tablourilor de numere întregi scrise pe foi. Fiecare profesor știe pasul la care intră în clasă.

Sarcina ta este să elaborezi și să implementezi o strategie pentru profesori și directorul școlii care să identifice corect dacă și când fiecare elev a ridicat mâna. Scorul tău va depinde de lungimea maximă a oricărui tablou scris pe o foaie: o lungime maximă mai scurtă va duce la un scor mai mare sau egal.

Detalii de implementare

Trebuie să implementați două proceduri, una pentru profesori și una pentru director.

Procedura pe care ar trebui să o implementați pentru **profesori** este:

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

- N : numărul de studenți.
- M : numărul de pași și, de asemenea, numărul de profesori.
- R : numărul pasului curent (de la 0 la $M - 1$).
- T : un tablou (posibil gol) constând din indicii studenților care ridică mâna în acest pas, sortat crescător.
- A : un tablou de lungime N care descrie lucrările, unde $A[i]$ ($0 \leq i < N$) este tabloul de numere întregi de pe lucrarea deținută de studentul i la începutul pasului.
- Această procedură este apelată exact de M ori pe joc, în ordinea $R = 0, 1, \dots, M - 1$.

Procedura ar trebui să returneze un tablou B de lungime N , reprezentând lucrările după modificările profesorului, în același format ca și A .

- Pentru fiecare elev i care a ridicat mâna (adică i apare în T), foaia de lucru nu trebuie schimbată, deci $B[i] = A[i]$.
- Pentru fiecare i astfel încât $0 \leq i < N$, lungimea lui $B[i]$ nu trebuie să depășească 63, iar fiecare număr întreg din $B[i]$ trebuie să fie între 0 și 63, inclusiv.

Procedura pe care ar trebui să o implementați pentru **director** este:

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

- N, M : la fel ca mai sus.
- A : un tablou de lungime N , unde $A[i]$ este tabloul de numere întregi de pe foaia deținută de elevul i după toți pașii M .
- Această procedură este apelată exact o dată pe joc, după apelul final al funcției `process_step`.

Procedura trebuie să returneze un tablou D de lungime N . Pentru fiecare i astfel încât $0 \leq i < N$, trebuie să se respecte că

- $D[i] = j$ dacă elevul i a ridicat mâna în timpul pasului j sau
- $D[i] = -1$ dacă elevul i nu a ridicat mâna pe tot parcursul jocului.

Programul dumneavoastră nu are permisiunea de a stoca sau transmite nicio informație de la un apel către `process_step` la altul, decât prin valoarea returnată de `process_step`. Dacă programul dumneavoastră încearcă să facă acest lucru, scorul dumneavoastră în CMS poate fi

incorect și poate fi redus după sfârșitul concursului. Rețineți că evaluatorul poate executa mai multe instanțe ale programului dumneavoastră simultan. Aceasta înseamnă că apelurile către `process_step` și `determine_steps` din același joc pot fi executate în instanțe diferite, iar apelurile către `process_step` și `determine_steps` din jocuri diferite pot fi executate în aceeași instanță. Fiecare caz de testare include până la 5 jocuri.

Restricții

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Fiecare elev ridică mâna **cel mult o dată** pe parcursul întregului joc. Cu alte cuvinte, pentru fiecare elev i , există cel mult un pas j în care i ridică mâna.
- La fiecare pas, cel puțin un elev **nu** ridică mâna.

Subtask-uri și punctaj

Subtask	Punctaj	Restricții suplimentare
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ pentru toate $0 \leq i \leq N - 1$.
4	25	Cel mult un elev ridică mâna la fiecare pas.
5	56	Fără restricții suplimentare.

Pentru fiecare caz de testare, scorul este 0 (raportat ca `Output isn't correct` în CMS) dacă valoarea returnată de orice apel la procedura `process_step` este invalidă sau dacă valoarea returnată de orice apel la procedura `determine_steps` este incorectă.

Altfel, fie C lungimea maximă a **oricărui** tablou din **oricare** B returnată de `process_step`. Apoi, scorul pentru un caz de testare dintr-un subtask cu scorul S este calculat ca $S \cdot X$, unde X depinde de C conform tabelului următor:

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

Exemplu

Să luăm în considerare un scenariu cu $N = 4$ elevi, $M = 2$ profesori și permutarea $P = [0, 3, 1, 2]$. Inițial, toate hârtiile sunt goale.

Graderul apelează:

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

Elevul 1 a ridicat mâna, așadar lucrarea sa nu poate fi modificată. Profesorul 0 poate decide să scrie $[10]$ pe foaia elevului 0, să scrie $[1, 63, 4]$ pe foaia elevului 3 și să lase foaia elevului 2 goală. Pentru a face acest lucru, procedura trebuie să returneze $B = [[10], [], [], [1, 63, 4]]$.

După ce profesorul 0 pleacă, elevii își schimbă foile în funcție de P . După schimb, foile sunt $A = [[10], [], [1, 63, 4], []]$.

Evaluatorul acum apelează:

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

Studentii 0 și 3 au ridicat mâna, așadar lucrările lor nu pot fi modificate. Profesorul 1 poate decide să scrie $[0, 40]$ pe foaia elevului 1 și să scrie $[1, 50]$ pe foaia elevului 2. Pentru a face acest lucru, procedura trebuie să returneze $B = [[10], [0, 40], [1, 50], []]$.

După ce profesorul 1 pleacă, foile sunt din nou schimbate conform lui P . După schimb, foile sunt $A = [[10], [1, 50], [], [0, 40]]$.

În final, evaluatorul aapelează:

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

Această procedură trebuie să returneze tabloul $D = [1, 0, -1, 1]$ deoarece elevii 0 și 3 au ridicat mâna în pasul 1, elevul 1 a ridicat mâna în pasul 0, iar elevul 2 nu a ridicat niciodată mâna. În acest exemplu, $C = 3$.

Grader local

Format de intrare:

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

Aici, Q este un tablou de lungime N , unde $Q[i] = j$ dacă elevul i ridică mâna în timpul pasului j sau $Q[i] = -1$ dacă elevul i nu ridică niciodată mâna pe tot parcursul jocului.

Format de ieșire:

După fiecare apel către `process_step`, graderul local afișează conținutul lucrărilor. Fie K lungimea lui B și $L[i]$ lungimea lui $B[i]$ (pentru fiecare $0 \leq i < K$). Graderul local afișează B în următorul format, **urmat de o linie goală**:

```
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]
```

Apoi, evaluatorul schimbă foile conform permutării P . Dacă $K \neq N$, graderul local afișează un mesaj de eroare și se termină.

După apelarea `determine_steps`, graderul local generează:

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

Aici, H este lungimea tabloului D returnat de `determine_steps`.