

Gra klasowa

Znana szkoła średnia w Taszkencie świętuje swoją rocznicę, organizując grę między uczniami i nauczycielami. W klasie siedzi N uczniów, ponumerowanych od 0 do $N - 1$. Każdy uczeń trzyma kartkę papieru zawierającą tablicę liczb całkowitych. Początkowo każda kartka jest pusta (czyli zawiera pustą tablicę).

Uczniowie grają w grę z M nauczycielami, ponumerowanymi od 0 do $M - 1$, oraz dyrektorem szkoły. Gra składa się z M kroków, ponumerowanych od 0 do $M - 1$. W kroku j nauczyciel j wchodzi do klasy i mają miejsce następujące zdarzenia:

- Kilku (być może zero) uczniów podnosi ręce. Gwarantowane jest, że w każdym kroku przynajmniej jeden uczeń **nie** podniesie ręki, a każdy uczeń może podnieść rękę **maksymalnie raz** podczas całej gry.
- Nauczyciel przegląda aktualnie trzymane przez uczniów kartki i może **zmodyfikować** kartki uczniów, którzy **nie** podnieśli ręki w tym kroku. Dla każdego takiego ucznia nauczyciel może zastąpić zawartość jego kartki nową (ewentualnie pustą) tablicą liczb całkowitych. Każda liczba całkowita musi mieścić się w przedziale od 0 do 63 włącznie, a tablica może mieć maksymalnie 63 elementów.
- Po tych modyfikacjach nauczyciel j odchodzi, a uczniowie wymieniają się kartkami zgodnie z tajną permutacją P . Oznacza to, że każdy uczeń i ($0 \leq i < N$) przekazuje swoją kartkę uczniowi $P[i]$, gdzie $P[0], P[1], \dots, P[N - 1]$ to N **różnych** liczb całkowitych pomiędzy 0 a $N - 1$ włącznie. Wartości P są **nie zmieniają się** podczas wszystkich kroków gry, ale nauczyciele i dyrektor szkoły ich nie znają.

Po zakończeniu wszystkich M kroków i ostatniej wymiany zgodnie z P , wchodzi dyrektor. Na podstawie wyłącznie kartek trzymanych przez uczniów, dyrektor musi określić dla każdego ucznia i ($0 \leq i < N$) numer kroku j , w którym uczeń i podniósł rękę, lub stwierdzić, że uczeń i nigdy nie podniósł ręki.

Nauczyciele nie mogą komunikować się z innymi nauczycielami ani z dyrektorem, chyba że za pomocą tablic liczb całkowitych zapisanych na kartkach. Każdy nauczyciel zna krok, w którym wchodzi do klasy.

Twoim zadaniem jest opracowanie i wdrożenie strategii dla nauczycieli i dyrektora, która prawidłowo określi, czy i kiedy każdy uczeń podniósł rękę. Twój wynik będzie zależał od maksymalnej długości tablicy zapisanej na kartce: krótsza tablica oznacza wyższy lub równy wynik.

Szczegóły implementacji

Należy zaimplementować dwie procedury: jedną dla nauczycieli i jedną dla dyrektora.

Procedura, którą powinieneś zaimplementować w przypadku **nauczycieli**, jest następująca:

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

- N : liczba studentów.
- M : liczba kroków, a także liczba nauczycieli.
- R : numer aktualnego kroku (od 0 do $M - 1$).
- T : tablica (być może pusta) zawierająca indeksy uczniów, którzy podnieśli rękę na tym etapie, posortowane w kolejności rosnącej.
- A : tablica o długości N opisująca kartki, gdzie $A[i]$ ($0 \leq i < N$) jest tablicą liczb całkowitych z kartki, którą student i miał na początku kroku.
- Tę procedurę wywołuje się dokładnie M razy na grę, w kolejności $R = 0, 1, \dots, M - 1$.

Procedura powinna zwrócić tablicę B o długości N , opisującą kartki po modyfikacjach wprowadzonych przez nauczyciela, w tym samym formacie co A .

- Dla każdego ucznia i , który podniósł rękę (czyli i pojawia się w T), kartka nie może zostać zmieniona, więc $B[i] = A[i]$.
- Dla każdego i takiego, że $0 \leq i < N$, długość $B[i]$ nie może przekraczać 63, a każda liczba całkowita w $B[i]$ musi mieścić się w przedziale od 0 do 63 włącznie.

Procedura, którą powinieneś zaimplementować w przypadku **dyrektora**, jest następująca:

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

- N, M : tak samo jak powyżej.
- A : tablica o długości N , gdzie $A[i]$ jest tablicą liczb całkowitych na kartce studenta i po wszystkich M krokach.
- Ta procedura jest wywoływana dokładnie raz na grę, po ostatnim wywołaniu `process_step`.

Procedura musi zwrócić tablicę D o długości N . Dla każdego i takiego, że $0 \leq i < N$, musi zachodzić, że

- $D[i] = j$ jeśli uczeń i podniósł rękę w kroku j lub
- $D[i] = -1$ jeśli uczeń i nie podniósł ręki przez całą grę.

Twój program nie ma prawa przechowywać ani przysyłać żadnych informacji z jednego wywołania `process_step` do drugiego, z wyjątkiem wartości zwracanej przez `process_step`. Jeśli Twój program spróbuje to zrobić, Twój wynik wyświetlany w CMS może być niepoprawny i może zostać obniżony po zakończeniu konkursu. Należy pamiętać, że program sprawdzający może uruchamiać wiele instancji Twojego programu jednocześnie. Oznacza to, że wywołania `process_step` i `determine_steps` z tej samej gry mogą być wykonywane w różnych instancjach, a wywołania `process_step` i `determine_steps` z różnych gier mogą być wykonywane w tej samej instancji. Każdy przypadek testowy obejmuje do 5 gier.

Ograniczenia

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Każdy uczeń podnosi rękę **maksymalnie raz** podczas całej gry. Innymi słowy, dla każdego ucznia i istnieje maksymalnie jeden krok j , w którym i podnosi rękę.
- W każdym kroku przynajmniej jeden uczeń **nie** podnosi ręki.

Podzadania i ocenianie

Podzadanie	Punkty	Dodatkowe ograniczenia
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ dla wszystkich $0 \leq i \leq N - 1$.
4	25	W każdym kroku najwyżej jeden uczeń podnosi rękę.
5	56	Brak dodatkowych ograniczeń.

Dla każdego przypadku testowego wynik wynosi 0 (zgłaszany jako `Output isn't correct` w CMS), jeśli wartość zwracana dowolnego wywołania procedury `process_step` jest nieprawidłowa lub jeśli wartość zwracana dowolnego wywołania procedury `determine_steps` jest nieprawidłowa.

W przeciwnym razie niech C będzie maksymalną długością **dowolnej** tablicy w **dowolnym** B zwróconym przez `process_step`. Następnie wynik dla przypadku testowego w podzadaniu o wyniku S jest obliczany jako $S \cdot X$, gdzie X zależy od C zgodnie z poniższą tabelą:

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

Przykład

Rozważmy scenariusz z $N = 4$ studentami, $M = 2$ nauczycielami i permutacją $P = [0, 3, 1, 2]$. Początkowo wszystkie kartki są puste.

Program sprawdzający najpierw wywołuje:

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

Uczeń 1 podniósł rękę, więc jego kartki nie można modyfikować. Nauczyciel 0 może zdecydować się napisać $[10]$ na kartce ucznia 0, napisać $[1, 63, 4]$ na kartce ucznia 3 i pozostawić kartkę ucznia 2 pustą. Aby to zrobić, procedura musi zwrócić $B = [[10], [], [], [1, 63, 4]]$.

Po wyjściu nauczyciela 0 uczniowie wymieniają się swoimi kartkami zgodnie z P . Po wymianie zawartość kartek to $A = [[10], [], [1, 63, 4], []]$.

Program sprawdzający następnie wywołuje:

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

Uczniowie 0 i 3 podnieśli rękę, więc ich kartki nie mogą zostać zmodyfikowane. Nauczyciel 1 może zdecydować się na napisanie $[0, 40]$ na kartce ucznia 1 i $[1, 50]$ na kartce ucznia 2. Aby to zrobić, procedura musi zwrócić $B = [[10], [0, 40], [1, 50], []]$.

Po wyjściu nauczyciela 1 następuje ponowna wymiana kartek zgodnie z P . Po wymianie zawartość kartek to $A = [[10], [1, 50], [], [0, 40]]$.

Na koniec program sprawdzający wywołuje:

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

Ta procedura musi zwrócić tablicę $D = [1, 0, -1, 1]$ ponieważ uczniowie 0 i 3 podnieśli rękę w kroku 1, uczeń 1 podniósł rękę w kroku 0, a uczeń 2 nigdy nie podniósł ręki. W tym przykładzie $C = 3$.

Przykładowy program sprawdzający

Format wejścia:

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

Tutaj Q jest tablicą o długości N , gdzie $Q[i] = j$ jeśli uczeń i podniesie rękę w kroku j lub $Q[i] = -1$ jeśli uczeń i nie podniesie ręki w trakcie całej gry.

Format wyjścia:

Po każdym wywołaniu `process_step` moduł oceniający wyświetla zawartość kartek. Niech K będzie długością B i $L[i]$ będzie długością $B[i]$ (dla każdego $0 \leq i < K$). Przykładowy program sprawdzający wypisuje B w następującym formacie, **po którym następuje pusty wiersz**:

```
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]
```

Następnie przykładowy program sprawdzający wymienia kartki zgodnie z permutacją P . Jeżeli $K \neq N$, przykładowy program sprawdzający wyświetla komunikat o błędzie i kończy działanie.

Po wywołaniu `determine_steps` przykładowy program oceniający wyświetla:

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

Wartość H oznacza długość tablicy D zwróconej przez `determine_steps`.