

Tantermi játék (Classroom game)

Taskent egyik neves középiskolája diákok és tanárok közötti játékkal ünnepli évfordulóját. N diák ül a tanteremben, 0-tól $N - 1$ -ig számozva. Minden diák egy papírlapot tart, amely egy egész számokból álló tömböt tartalmaz. Kezdetben minden papír üres, tehát egy üres tömböt tartalmaz.

A diákok M tanárral játsszák a játékot, akik 0-tól $M - 1$ -ig számozottak, valamint az igazgatóval. A játékot M körön keresztül játsszák, amelyeket szintén 0-tól $M - 1$ -ig számoznak. A j . körben a j . tanár belép a tanterembe, és a következő események történnek:

- Néhány (esetleg nulla) diák felemeli a kezét. Garantált, hogy minden körben legalább egy diák **nem** emeli fel a kezét, és minden diák **legfeljebb egyszer** emelheti fel a kezét a teljes játék során.
- A tanár megvizsgálja a diákok által jelenleg birtokolt papírlapokat, és **módosíthatja** azoknak a diákoknak a dolgozatait, akik **nem** emelték fel a kezüket ebben a körben. Minden ilyen diák esetében a tanár lecserélheti a papírlap tartalmát egy új (esetleg üres) egész számokból álló tömbre. Minden egész számnak 0 és 63 között kell lennie, a tömbnek legfeljebb 63 eleme lehet.
- A módosítások után a j . tanár távozik, és a diákok kicserélik a papírjaikat egy titkos P permutáció szerint. Vagyis minden i -re ($0 \leq i < N$) az i . diák átadja a papírját a $P[i]$ indexű diáknak, ahol $P[0], P[1], \dots, P[N - 1]$ N **különböző** egész szám 0 és $N - 1$ között. P értékei **rögzítettek** a teljes játék alatt, de a tanárok és az igazgató nem ismerik őket.

Miután mind az M kör befejeződött, és a P szerinti utolsó cserélgetés is megtörtént, belép az igazgató. Kizárólag a diákok által tartott papírokat vizsgálva az igazgatónak minden i ($0 \leq i < N$) diákra meg kell határoznia annak a körnek a j indexét, amelyben az i . diák felemelte a kezét, vagy arra a következtetésre kell jutnia, hogy az i . diák soha nem emelte fel a kezét.

A tanárok nem kommunikálhatnak más tanárokkal vagy az igazgatóval a körök között, kivéve a papírokra írt egész számokból álló tömbökön keresztül. Minden tanár tudja a saját sorszámát, amelyik körben belép a terembe.

A feladatod egy olyan stratégia kidolgozása és megvalósítása a tanárok és az igazgató számára, amely helyesen azonosítja, hogy az egyes diákok felemelték-e a kezüket, és ha igen, mikor. A pontszámod a papírra írt tömbök maximális hosszától függ: egy rövidebb maximális hosszhoz magasabb vagy egyenlő pontszám tartozik.

Megvalósítás

Két függvényt kell implementálnod, egyet a tanárok, egyet pedig az igazgató számára.

A **tanárok** esetében a következő függvényt implementáld:

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

- N : a diákok száma.
- M : a körök száma, ami megegyezik a tanárok számával.
- R : az aktuális kör indexe (0-tól $M - 1$ -ig).
- T : egy (esetleg üres) tömb, amely az ebben a körben kezűket felemelők indexeit tartalmazza növekvő sorrendben.
- A : egy N hosszúságú tömb, amely a lapokat írja le, ahol $A[i]$ ($0 \leq i < N$) az i . diák által a kör elején tartott lapon lévő egész számok tömbje.
- Ezt a függvényt játékonként pontosan M alkalommal hívjuk meg, a következő sorrendben: $R = 0, 1, \dots, M - 1$.

A függvénynek egy N hosszúságú B tömböt kell visszaadnia, amely a tanár módosításai utáni papírlapok állapotát tartalmazza, ugyanabban a formátumban, mint az A tömb.

- Ha az i . diák, felemelte a kezét (azaz i megjelenik T -ben), akkor a tömb tartalmát tilos megváltoztatni, tehát $B[i] = A[i]$.
- Minden i ($0 \leq i < N$) esetén $B[i]$ hossza nem haladhatja meg a 63-at, és minden $B[i]$ egész számnak 0 és 63 között kell lennie (a határokat is beleértve).

Az **igazgató** esetében a következő függvényt kell implementálnod:

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

- N, M : ugyanaz, mint fent.
- A : egy N hosszúságú tömb, ahol $A[i]$ az i . diák által M kör után tartott papíron lévő egész számok tömbje.
- Ez a függvény játékonként pontosan egyszer hívódik meg, a `process_step` utolsó hívása után.

A függvénynek egy N hosszú D tömböt kell visszaadnia. Minden i értékre ($0 \leq i < N$) teljesülnie kell, hogy

- $D[i] = j$ ha az i . diák felemelte a kezét a j . körben, vagy
- $D[i] = -1$, ha az i . diák egyszer sem emelte fel a kezét a játék során.

A programod nem tárolhat vagy továbbíthat információt az egyik `process_step` hívásból a másikba, csak és kizárólag a `process_step` visszatérési értékén keresztül. Ha a programod ezt megpróbálja megtenni, a CMS-ben elért pontszámod helytelen lehet, és a verseny vége után lehet, hogy csökkentve lesz. Vedd figyelembe, hogy az értékelő a programod több példányát is futtathatja egyszerre. Ez azt jelenti, hogy ugyanazon játék `process_step` és `determine_steps` hívásai különböző futtatások során hajtódnak végre, és különböző játékok `process_step` és `determine_steps` hívásai ugyanabban a futtatásban is végrehajthatódnak. Minden teszteset legfeljebb 5 játékot tartalmaz.

Korlátok

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Minden diák **legfeljebb egyszer** emeli fel a kezét a játék során. Más szóval, minden i diákra legfeljebb egy j kör van, amelyben i felemeli a kezét.
- Minden körnél legalább egy diák **nem** emeli fel a kezét.

Részfeladatok

Részfeladat	Pontszám	További megkötések
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ minden $0 \leq i \leq N - 1$ esetén.
4	25	Legfeljebb egy diák emeli fel a kezét körönként.
5	56	Nincsenek további megkötések.

Minden tesztesetnél a pontszám 0 (a CMS-ben Output isn't correct hibaként jelenik meg), ha a `process_step` függvény bármely hívásának visszatérési értéke érvénytelen, vagy ha a `determine_steps` függvény bármely hívásának visszatérési értéke érvénytelen.

Egyébként legyen C **bármely** `process_step` függvényhívás során visszaadott B -ben található **bármelyik** tömb maximális hossza. Ekkor egy S pontszámú részfeladat tesztesetének pontszáma a következőképpen számítható ki: $S \cdot X$, ahol X a C értékétől függ a következő táblázat szerint:

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

Példa

Tekintsünk egy olyan esetet, ahol $N = 4$ diák, $M = 2$ tanár van, és $P = [0, 3, 1, 2]$ a permutáció. Kezdetben minden papír üres.

Az értékelő először a következőt hívja:

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

Az 1. diák felemelte a kezét, így a papírlapja nem módosítható. A 0. tanár dönthet úgy, hogy $[10]$ -t ír a 0. diák papírjára, $[1, 63, 4]$ -t a 3. diák papírjára, és a 2. diák papírját üresen hagyja. Ehhez az eljárásnak $B = [[10], [], [], [1, 63, 4]]$ -t kell visszaadnia.

Miután a 0. tanár elmegy, a diákok P szerint kicserélik a papírjaikat. A csere után a papírok a következőképpen alakulnak $A = [[10], [], [1, 63, 4], []]$.

Az értékelő most a következőt hívja:

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

A 0. és 3. diákok felemelték a kezüket, így a papírlapjaik nem módosíthatók. Az 1. tanár dönthet úgy, hogy $[0, 40]$ -t ír az 1. diák papírjára, és $[1, 50]$ -t a 2. diák papírjára. Ehhez az eljárásnak $B = [[10], [0, 40], [1, 50], []]$ -t kell visszaadnia.

Miután az 1. tanár elment, a papírokat ismét kicserélik P szerint. A csere után a papírok: $A = [[10], [1, 50], [], [0, 40]]$.

Végül az értékelő meghívja:

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

Ennek az eljárásnak a $D = [1, 0, -1, 1]$ tömböt kell visszaadnia, mivel a 0. és 3. diákok felemelték a kezüket az 1. körben, az 1. diák felemelte a kezét a 0. körben, a 2. diák pedig soha nem emelte fel a

kezét. Ebben a példában $C = 3$.

Mintaértékelő

Bemeneti formátum:

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

Itt Q egy N hosszúságú tömb, ahol $Q[i] = j$ ha az i . diák felemeli a kezét j . körben, vagy $Q[i] = -1$ ha az i . diák egyszer sem emeli fel a kezét a játék során.

Kimeneti formátum:

A `process_step` minden egyes meghívását követően a mintaértékelő kimenetként kiírja a dolgozatok tartalmát. Legyen K a B hossza, $L[i]$ pedig $B[i]$ hossza (minden $0 \leq i < K$ esetén). A mintaértékelő a B értékét a következő formátumban írja ki, **utána egy üres sorral**:

```
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]
```

A mintaértékelő ezután a P permutációnak megfelelően kicseréli a dolgozatokat. Ha $K \neq N$, a mintaértékelő hibaüzenetet ír ki, és ehelyett leáll.

A `determine_steps` meghívása után a mintaértékelő a következő kimenetet adja:

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

Itt H a `determine_steps` által visszaadott D tömb hossza.