

Mäng klassis

Üks kuulus keskkool Taškendis korraldab oma aastapäevaks õpilaste ja õpetajate vahelise mängu. Klassis istub N õpilast, kes on nummerdatud 0 kuni $N - 1$. Igal õpilasel on paberileht, millel on täisarvude massiiv. Algselt on iga leht tühi, ehk sellel on tühi massiiv.

Õpilased mängivad mängu M õpetajaga, kes on nummerdatud 0 kuni $M - 1$, ja direktoriga. Mängu mängitakse M sammuna, mis on samuti nummerdatud 0 kuni $M - 1$. Sammuli j siseneb klassi õpetaja j ja toimuvad järgmised sündmused:

- Mõned (võimalik, et null) õpilast tõstavad käe. On garanteeritud, et igal sammul on vähemalt üks õpilane, kes **ei** tõsta kätt, ja et iga õpilane võib kogu mängu jooksul kätt tõsta **maksimaalselt ühe korra**.
- Õpetaja vaatab üle õpilastel hetkel käes olevad paberid ja võib **muuta** nende õpilaste pabereid, kes sellel sammul kätt **ei** tõstnud. Iga sellise õpilase paberi sisu võib õpetaja asendada uue (võimalik, et tühja) arvumassiiviga. Iga arv peab olema lõigust 0 kuni 63 (kaasa arvatud) ja igas massiivis võib olla maksimaalselt 63 elementi.
- Seejärel lahkub õpetaja j klassist ja õpilased vahetavad oma pabereid vastavalt salajasele permutatsioonile P . See tähendab, et õpilane i ($0 \leq i < N$) annab oma paberi õpilasele $P[i]$, kus $P[0], P[1], \dots, P[N - 1]$ on N **erinevat** täisarvu lõigust 0 kuni $N - 1$ (kaasa arvatud). P väärtused on kogu mängu jooksul **fikseeritud**, kuid õpetajad ja direktor neid ei tea.

Kui kõik M sammu on tehtud (sealhulgas viimane paberite vahetus P kohaselt sooritatud), siseneb klassi direktor. Vaadates ainult õpilaste käes olevaid pabereid, peab direktor iga õpilase i ($0 \leq i < N$) jaoks määrama sammu numbri j , mille ajal õpilane i käe tõstis, või tuvastama, et õpilane i ei tõstnud kordagi kätt.

Õpetajad ei tohi teiste õpetajate ega direktoriga suhelda muul moel kui õpilaste paberitele kirjutatud arvumassiivide kaudu. Iga õpetaja teab, mitmendal sammul ta klassis käib.

Sinu ülesanne on välja töötada ja realiseerida õpetajatele ja direktorile strateegia, mis tuvastab õigesti, kas ja millal iga õpilane käe tõstis. Sinu tulemus sõltub paberile kirjutatud arvumassiivide maksimaalsest pikkusest: lühem maksimaalne pikkus annab kõrgema (või vähemalt sama) skoori.

Realiseerimine

Sul tuleb kirjutada kaks funktsiooni, üks õpetajatele ja teine direktorile.

Õpetajate jaoks kirjutada järgmine funktsioon:

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

- N : õpilaste arv.
- M : sammude arv ja ka õpetajate arv.
- R : praeguse sammu number (0 kuni $M - 1$).
- T : (võimalik, et tühi) kasvavalt sorteeritud massiiv, milles on sellel sammul käe tõstnud õpilaste indeksid.
- A : N -elemendiline massiiv, mis kirjeldab pabereid, kus $A[i]$ ($0 \leq i < N$) on õpilase i käes oleva paberil sammu alguses olev arvumassiiv.
- Seda funktsiooni kutsutakse välja täpselt M korda mängu kohta, järjekorras $R = 0, 1, \dots, M - 1$.

Funktsioon peab tagastama N -elemendilise massiivi B , mis kirjeldab pabereid pärast õpetaja tehtud muudatusi samas vormingus nagu funktsiooni parameetris A .

- Kui õpilane i tõstis käe (st i esineb massiivis T), ei tohi tema paberit muuta, seega $B[i] = A[i]$.
- Iga $0 \leq i < N$ korral võib $B[i]$ pikkus olla maksimaalselt 63 ja kõik $B[i]$ elemendid peavad olema lõigust 0 kuni 63 (kaasa arvatud).

Direktori jaoks kirjutada järgmine funktsioon:

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

- N, M : nagu eespool.
- A : N -elemendiline massiiv, kus $A[i]$ on õpilase i käes oleva paberil olev arvumassiiv pärast kõigi M sammu sooritamist.
- Seda funktsiooni kutsutakse välja täpselt üks kord mängu kohta, pärast viimast `process_step` väljakutset.

Funktsioon peab tagastama N -elemendilise massiivi D . Iga $0 \leq i < N$ korral peab kehtima

- $D[i] = j$, kui õpilane i tõstis sammu j ajal käe, või
- $D[i] = -1$, kui õpilane i ei tõstnud kogu mängu jooksul kordagi kätt.

Programmil ei ole lubatud salvestada ega edastada teavet ühelt `process_step` väljakutselt teisele, välja arvatud `process_step` tagastatava väärtuse kaudu. Kui programm proovib seda nõuet rikkuda, võib CMSis võistluse ajal näidatav skoor olla vale ja seda võidakse pärast võistluse

lõppu vähendada. Pane tähele, et hindaja võib käivitada mitu sinu programmi eksemplari samaaegselt. See tähendab, et sama mängu `process_step` ja `determine_steps` väljakutseid võidakse käivitada programmi erinevates eksemplarides ning erinevate mängude `process_step` ja `determine_steps` väljakutseid võidakse käivitada programmi samas eksemplaris. Iga test sisaldab kuni 5 mängu.

Piirangud

- $2 \leq N \leq 63$.
- $1 \leq M \leq 63$.
- Iga õpilane tõstab käe **maksimaalselt ühe korra** kogu mängu jooksul. Teisisõnu, iga õpilase i kohta on maksimaalselt üks samm j , mille ajal õpilane i käe tõstab.
- Igal sammul on vähemalt üks õpilane, kes **ei** tõsta kätt.

Alamülesanded ja hindamine

Alamülesanne	Skoor	Lisapiirangud
1	4	$M = 1$.
2	6	$N = 2$.
3	9	$P[i] = i$ iga $0 \leq i \leq N - 1$ korral.
4	25	Igal sammul tõstab käe maksimaalselt üks õpilane.
5	56	Lisapiiranguid pole.

Igas testis on lahenduse skoor 0 (CMSis kuvatakse kui `Output isn't correct`), kui funktsiooni `process_step` mistahes väljakutse tagastas lubamatu väärtuse või kui funktsiooni `determine_steps` mistahes väljakutse tagastas vale väärtuse.

Vastasel juhul olgu C maksimaalne elementide arv üle **kõigi** funktsiooni `process_step` tagastatud B väärtuste **kõigi** elementide. Siis arvutatakse S punkti väärt alamülesandes testi skoor valemiga $S \cdot X$, kus X sõltub C väärtusest vastavalt järgmisele tabelile:

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

Näide

Olgu meil $N = 4$ õpilast, $M = 2$ õpetajat ja permutatsioon $P = [0, 3, 1, 2]$. Alguses on kõik paberid tühjad.

Hindaja kutsub esmalt välja:

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

Õpilane 1 tõstis käe, seega tema paberit muuta ei tohi. Õpetaja 0 võib otsustada kirjutada õpilase 0 paberile $[10]$, õpilase 3 paberile $[1, 63, 4]$ ja jätta õpilase 2 paberi tühjaks. Selleks peab funktsioon tagastama $B = [[10], [], [], [1, 63, 4]]$.

Pärast õpetaja 0 lahkumist vahetavad õpilased oma pabereid vastavalt permutatsioonile P . Pärast vahetust on paberid $A = [[10], [], [1, 63, 4], []]$.

Hindaja kutsub nüüd välja:

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

Õpilased 0 ja 3 on käed tõstnud, seega nende pabereid muuta ei tohi. Õpetaja 1 võib otsustada kirjutada õpilase 1 paberile $[0, 40]$ ja õpilase 2 paberile $[1, 50]$. Selleks peab protseduur tagastama $B = [[10], [0, 40], [1, 50], []]$.

Pärast õpetaja 1 lahkumist vahetatakse uuesti pabereid P kohaselt. Pärast vahetust on paberid $A = [[10], [1, 50], [], [0, 40]]$.

Lõpuks kutsub hindaja välja:

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

See funktsioon peab tagastama massiivi $D = [1, 0, -1, 1]$ kuna õpilased 0 ja 3 tõstsid käed sammul 1, õpilane 1 tõstis käe sammul 0 ja õpilane 2 ei tõstnudki kätt. Selles näites $C = 3$.

Näidishindaja

Sisendi vorming

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

Siin on Q massiiv pikkusega N , kus $Q[i] = j$, kui õpilane i tõstab käe sammul j või $Q[i] = -1$, kui õpilane i ei tõsta kogu mängu jooksul kordagi kätt.

Väljundi vorming

Pärast iga `process_step` kutset väljastab näidishindaja kõigi paberite sisu. Olgu K massiivi B pikkus ja $L[i]$ iga $0 \leq i < K$ korral massiivi $B[i]$ pikkus. Näidishindaja kuvab B järgmises vormingus, **millele järgneb tühi rida**:

```
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]
```

Seejärel vahetab näidishindaja paberid vastavalt permutatsioonile P . Kui $K \neq N$, kuvab näidishindaja veateate ja lõpetab töö.

Pärast `determine_steps` kutset annab näidishindaja välja järgmise tulemuse:

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

Siin on H funktsiooni `determine_steps` tagastatud massiivi D pikkus.