

Igra v razredu

Ugledna srednja šola v Taškentu praznuje svojo obletnico in ob tem organizira igro med dijaki in učitelji. V razredu sedi N dijakov, oštevilčenih od 0 do $N - 1$. Vsak dijak ima list papirja, na katerem je zapisano polje celih števil. Na začetku je vsak list papirja prazen, t.j. vsebuje prazno polje.

Dijaki igrajo igro z M učitelji, oštevilčenimi od 0 do $M - 1$, in ravnateljem. Igra se izvaja v M korakih, ki so prav tako oštevilčeni od 0 do $M - 1$. V j -tem koraku vstopi v razred učitelj j in zgodijo se naslednji dogodki:

- Nekateri (morda noben) dijaki dvignejo roko. Zagotovljeno je, da pri vsakem koraku vsaj en dijak **ne** dvigne roke, in vsak dijak lahko dvigne roko **največ enkrat** v celotni igri.
- Učitelj pregleda liste, ki jih imajo dijaki trenutno, in lahko **spremeni** vsebino na listih tistih dijakov, ki v tem koraku **niso** dvignili roke. Pri vsakem takem dijaku lahko učitelj zamenja vsebino njegovega lista z novim (morda praznim) poljem celih števil. Vsako celo število mora biti med vključno 0 in vključno 63 in polje lahko vsebuje največ 63 elementov.
- Ko učitelj opravi te spremembe, zapusti učilnico, dijaki pa izmenjajo svoje liste po skrivni permutaciji P . To pomeni, da vsak dijak i ($0 \leq i < N$) izroči svoj list dijaku $P[i]$, kjer so $P[0], P[1], \dots, P[N - 1]$ N **različnih** celih števil med vključno 0 in vključno $N - 1$. Vrednosti P so **nespremenljive** skozi vse korake igre, vendar jih učitelji in ravnatelj ne poznajo.

Ko se izvede vseh M korakov in se izvede zadnja izmenjava po P , vstopi ravnatelj. Le s pregledom listov, katere imajo dijaki, mora ravnatelj ugotoviti za vsakega dijaka i ($0 \leq i < N$), pri katerem j -tem koraku je dvignil roko ali ugotoviti, da dijak i nikoli ni dvignil roke.

Učitelji med seboj in z ravnateljem ne smejo komunicirati, razen preko celih števil, ki so zapisana na listih. Vsak učitelj ve, v katerem koraku vstopi v razred.

Vaša naloga je zasnovati in implementirati strategijo za učitelje in ravnatelja, ki pravilno prepozna, kdaj, če sploh kdaj, je vsak dijak dvignil roko. Vaša ocena bo odvisna od največje dolžine katerega koli polja zapisanega na listih. Krajša največja dolžina prinaša višjo ali enako oceno.

Podrobnosti implementacije

Implementirati morate dve funkciji - eno za učitelje in eno za ravnatelja.

Funkcija, ki jo implementirate za **učitelje**, je:

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

- N : število dijakov.
- M : število korakov in tudi število učiteljev.
- R : trenutna številka koraka (od 0 do $M - 1$).
- T : (morda prazno) polje, ki vsebuje indekse dijakov, ki v tem koraku dvignejo roko, urejeno v naraščajočem vrstnem redu.
- A : polje dolžine N , ki opisuje liste, kjer $A[i]$ ($0 \leq i < N$) predstavlja polje celih števil na listu, ki ga ima dijak i ob začetku koraka.
- To funkcijo se kliče natanko M -krat na igro v vrstnem redu $R = 0, 1, \dots, M - 1$.

Funkcija naj vrne polje B dolžine N , ki predstavlja liste po učiteljevih spremembah. Polje B mora biti v istem formatu kot A .

- Za vsak i , ki se pojavi v T , se vsebine papirja ne sme spremeniti, zato $B[i] = A[i]$.
- Za vsak i , kjer $0 \leq i < N$, dolžina $B[i]$ ne sme presegati 63, in vsako celo število v $B[i]$ mora biti med vključno 0 in vključno 63.

Funkcija, ki jo implementirate za **ravnatelja**, je:

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

- N, M : enako kot zgoraj.
- A : polje dolžine N , kjer $A[i]$ predstavlja polje celih števil na listu, ki ga ima dijak i po vseh M -ih korakih.
- Funkcijo se kliče natanko enkrat na igro, in sicer po zadnjem klicu `process_step`.

Funkcija naj vrne polje D dolžine N . Za vsak i , kjer $0 \leq i < N$, mora veljati:

- $D[i] = j$, če je dijak i dvignil roko v j -tem koraku, oz.
- $D[i] = -1$, če dijak i nikoli ni dvignil roke.

Vaš program ne sme shranjevati ali prenašati kakršnih koli informacij iz enega klica `process_step` v drugega, razen preko vrnjene vrednosti `process_step`. Če vaš program to poskuša storiti, so lahko vaše ocene v sistemu CMS napačne in se lahko znižajo po koncu tekmovanja. Upoštevajte, da lahko ocenjevalec izvede več primerkov vašega programa hkrati. To pomeni, da se klici `process_step` in `determine_steps` iz iste igre lahko izvedejo v različnih primerih, klici `process_step` in `determine_steps` iz različnih iger pa se lahko izvedejo v istem primeru. Vsak testni primer vsebuje do 5 iger.

Omejitve

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Vsak dijak dvigne roko **največ enkrat** pri celotni igri. Z drugimi besedami, za vsakega dijaka i obstaja največ en korak j , pri katerem i dvigne roko.
- Pri vsakem koraku vsaj en dijak **ne** dvigne roke.

Podnaloge in ocenjevanje

Podnaloga	Točke	Dodatne omejitve
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ za vse $0 \leq i \leq N - 1$.
4	25	Pri vsakem koraku največ en dijak dvigne roko.
5	56	Brez dodatnih omejitev.

Za vsak testni primer so vaše točke 0 (v CMS-u navedeno kot `Output isn't correct`), če je vrnjena vrednost katerega koli klica procedure `process_step` neveljavna, ali če je vrnjena vrednost katerega koli klica `determine_steps` napačna.

V nasprotnem primeru naj bo C največja dolžina **katerega koli** polja v **katerem koli** B , ki ga vrne `process_step`. Potem za vsak testni primer v podnalogi s točkami S točke izračunamo kot $S \cdot X$, kjer je X odvisen od C in je določen z naslednjo tabelo:

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

Primer

Vzemite primer, kjer imamo $N = 4$ dijakov, $M = 2$ učitelja in permutacijo $P = [0, 3, 1, 2]$. Na začetku so vsi listi papirja prazni.

Ocenjevalnik najprej pokliče:

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

Dijak 1 je dvignil roko, zato njegovega lista ni mogoče spremeniti. Učitelj 0 se lahko odloči napisati [10] na list dijaku 0, napisati [1, 63, 4] na list dijaku 3, in pusti prazen list pri dijaku 2. V tem primeru mora vrnjeno polje vsebovati $B = [[10], [], [], [1, 63, 4]]$.

Ko učitelj 0 zapusti učilnico, dijaki zamenjajo svoje liste po P . Po menjavi so listi $A = [[10], [], [1, 63, 4], []]$.

Ocenjevalnik sedaj pokliče:

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

Dijaka 0 in 3 sta dvignila roko, zato njunih listov ni možno spreminjati. Učitelj 1 se lahko odloči napisati [0, 40] na list dijaku 1 in napisati [1, 50] na list dijaku 2. V tem primeru mora vrnjeno polje vsebovati $B = [[10], [0, 40], [1, 50], []]$.

Ko učitelj 1 zapusti učilnico, se listi ponovno izmenjajo po P . Po izmenjavi so listi $A = [[10], [1, 50], [], [0, 40]]$.

Na koncu ocenjevalnik pokliče:

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

Ta funkcija mora vrniti polje $D = [1, 0, -1, 1]$, ker sta dijaka 0 in 3 dvignila roko v koraku 1, dijak 1 je dvignil roko v koraku 0 in dijak 2 nikoli ni dvignil roke. V tem primeru je $C = 3$.

Vzorčni ocenjevalnik

Oblika vhoda

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

Tu je Q polje dolžine N , kjer $Q[i] = j$, če je dijak i dvignil roko v koraku j , oz. $Q[i] = -1$, če dijak i nikoli ni dvignil roke.

Oblika izhoda

Po vsakem klicu `process_step` vzorčni ocenjevalnik izpiše vsebino listov papirja. Naj bo K dolžina B in $L[i]$ dolžina $B[i]$ (za vsak $0 \leq i < K$). Vzorčni ocenjevalnik izpiše B v naslednji obliki, **ki mu sledi prazna vrstica**:

```
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]
```

Vzorčni ocenjevalnik nato izvede izmenjavo listov glede na permutacijo P . Če $K \neq N$, vzorčni ocenjevalnik izpiše sporočilo o napaki in zaključi izvajanje.

Po klicu `determine_steps` vzorčni ocenjevalnik izpiše:

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

Tu je H dolžina polja D , ki ga vrne `determine_steps`.