



Klassrumsspelet

En framstående gymnasieskola i Tasjkent firar sitt jubileum genom att anordna en match mellan elever och lärare. Det sitter N elever i klassrummet, numrerade från 0 till $N-1$. Varje elev håller ett papper som innehåller en array med heltal. Ursprungligen är varje papper tomt, så det innehåller en tom matris.

Eleverna spelar spelet med M lärare, numrerade från 0 till $M-1$, och rektorn. Spelet spelas i M steg, även de numrerade från 0 till $M-1$. I steg j kommer läraren j in i klassrummet och följande händelser inträffar:

- Ett antal elever (möjligen noll) räcker upp handen. Det garanteras att minst en elev i varje steg **inte** räcker upp handen, och varje elev får räcka upp handen **högst en gång** under hela spelet.
- Läraren tittar på de pappren som eleverna för närvarande har och kan **modifiera** pappren som de elever som **inte** räckte upp handen i detta steg har. För varje sådan elev kan läraren ersätta innehållet i deras pappret med en ny (möjligen tom) array med heltal. Varje heltal måste vara mellan 0 och 63 , och array kan innehålla högst 63 -element.
- Efter dessa modifieringar går läraren j och eleverna byter sina papper enligt en hemlig permutation P . Det vill säga att varje elev i ($0 \leq i < N$) ger sitt papper till eleven $P[i]$, där $P[0], P[1], \dots, P[N-1]$ är N **distinkta** heltal mellan 0 och $N-1$, inklusive. Värdena på P är **fasta** under alla steg i spelet, men lärarna och rektorn känner inte till dem.

När alla M steg är avslutade och det sista utbytet enligt P har utförts, går rektorn in. Rektorn måste, genom att endast titta på de papper som eleverna har, för varje elev i ($0 \leq i < N$), bestämma stegnumret j där eleven i räckte upp handen, eller dra slutsatsen att eleven i aldrig räckte upp handen.

Lärarna kan inte kommunicera med andra lärare eller rektorn, förutom genom de heltal som står skrivna på pappren. Varje lärare vet i vilket steg de kommer in i klassrummet.

Din uppgift är att utforma och implementera en strategi för lärarna och rektorn som korrekt identifierar om och när varje elev räckte upp handen. Din poäng beror på den maximala längden på en matris som skrivs på ett papper: en kortare maximal längd ger en högre eller lika stor poäng.

Implementationsdetaljer

Du behöver implementera två funktioner, en för lärarna och en för rektorn.

Funktionen du bör implementera för **lärarna** är:

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

- N : antalet elever.
- M : antalet steg samt antalet lärare.
- R : det aktuella stegnumret (från 0 till $M - 1$).
- T : en (eventuellt tom) array bestående av indexen för de elever som räcker upp handen i detta steg, sorterade i stigande ordning.
- A : en array med längden N som beskriver papperen, där $A[i]$ ($0 \leq i < N$) är arrayen med heltal på det papper som eleven i håller i början av steget.
- Denna funktion anropas exakt M gånger per spel, i ordningen $R = 0, 1, \dots, M - 1$.

Funktionen ska returnera en array B med längden N , som representerar papprena efter lärarens ändringar, i samma format som A .

- För varje elev i som räckte upp handen (det vill säga i förekommer i T) får uppsatsen inte ändras, alltså $B[i] = A[i]$.
- För varje i så att $0 \leq i < N$ gäller att längden på $B[i]$ inte får överstiga 63, och varje heltal i $B[i]$ måste ligga mellan 0 och 63, inklusivt.

Funktionen du ska implementera för **rektorn** är:

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

- N, M : samma som ovan.
- A : en array med längden N , där $A[i]$ är den array av heltal på papperet som studenten i har kvar efter alla M steg.
- Denna funktion anropas exakt en gång per spel, efter det sista anropet till `process_step`.

Funktionen måste returnera en array D med längden N . För varje i så att $0 \leq i < N$ gäller att

- $D[i] = j$ om studenten i räckte upp handen under steg j , eller
- $D[i] = -1$ om studenten i aldrig räckte upp handen under hela spelet.

Ditt program får inte lagra eller överföra någon information från ett anrop till `process_step` till ett annat, förutom via returvärdet från `process_step`. Om ditt program försöker göra detta kan din poäng i CMS bli felaktig och komma att sänkas efter tävlingens slut. Observera att bedömare kan köra flera instanser av ditt program samtidigt. Detta innebär att anrop till `process_step` och `determine_steps` från samma spel kan köras i olika instanser, och att anrop till

process_step och determine_steps från olika spel kan köras i samma instans. Varje testfall innehåller upp till 5 spel.

Begränsningar

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Varje elev räcker upp handen **högst en gång** under hela spelet. Med andra ord: för varje elev i finns det högst ett steg j där i räcker upp handen.
- Vid varje steg finns det minst en elev som **inte** räcker upp handen.

Deluppgifter och Poängsättning

Deluppgift	Poäng	Ytterligare begränsningar
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ for all $0 \leq i \leq N - 1$.
4	25	Högst en elev räcker upp handen vid varje steg.
5	56	Inga ytterligare begränsningar.

För varje testfall blir ditt resultat 0 (rapporteras som Output isn't correct i CMS) om returvärdet från något anrop till funktionen process_step är ogiltigt eller om returvärdet från något anrop till funktionen determine_steps är felaktigt.

I annat fall, låt C vara den maximala längden på **vilken som helst** array i **vilken som helst** B som returneras av process_step. Då beräknas poängen för ett testfall i en deluppgift med poängen S som $S \cdot X$, där X beror på C enligt följande tabell:

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

Exempel

Tänk dig ett scenario med $N = 4$ studenter, $M = 2$ lärare och permutationen $P = [0, 3, 1, 2]$. Inledningsvis är alla papper tomma.

Gradern anropar först:

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

Eleven 1 har räckt upp handen, så hennes papper kan inte ändras. Läraren 0 kan välja att skriva "[10]" på pappret som tillhör elev 0, skriva "[1,63,4]" på pappret som tillhör elev 3 och lämna pappret som tillhör elev 2 tom. För att göra detta måste funktionen returnera " $B = [[10], [], [], [1, 63, 4]]$ ".

När läraren 0 har gått ut byter eleverna sina papper enligt P . Efter utbytet befinner sig papprena $A = [[10], [], [1, 63, 4], []]$.

Gradern ropar nu:

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

Eleverna 0 och 3 har räckt upp handen, så deras papper kan inte ändras. Läraren 1 kan besluta att skriva $[0, 40]$ på pappret tillhörande elev 1 och skriva $[1, 50]$ på pappret tillhörande elev 2. För att detta ska gå att göra måste funktionen returnera $B = [[10], [0, 40], [1, 50], []]$.

När läraren 1 har gått byts papprena återigen enligt P . Efter utbytet ser papprena ut enligt $A = [[10], [1, 50], [], [0, 40]]$.

Slutligen ropar gradern:

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

Denna funktion måste returnera arrayen $D = [1, 0, -1, 1]$ eftersom eleverna 0 och 3 räckte upp handen i steg 1, eleven 1 räckte upp handen i steg 0 och eleven 2 aldrig räckte upp handen. I detta exempel gäller att $C = 3$.

Exempelgradern

Inputformat:

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

Här är Q en array med längden N , där $Q[i] = j$ gäller om eleven i räcker upp handen under steg j , eller $Q[i] = -1$ gäller om eleven i aldrig räcker upp handen under hela spelet.

Outputformat:

Efter varje anrop till `process_step` skriver exempelgradern ut innehållet i papprena. Låt K vara längden på B och $L[i]$ vara längden på $B[i]$ (för varje $0 \leq i < K$). Exempelbedömaren skriver ut B i följande format, **följt av en tom rad**:

```
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]
```

Exempelgradern byter sedan ut uppsatserna enligt permutationen P . Om $K \neq N$ skriver exempelgradern istället ut ett felmeddelande och avslutas.

Efter att ha anropat `determine_steps` skriver exempelgradern ut:

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

Här är H längden på arrayen D som returneras av `determine_steps`.