

Classroom Game

Een vooraanstaande school in Tasjkent viert zijn verjaardag door een spel te organiseren tussen de studenten en docenten. Er zitten N studenten in een klaslokaal, genummerd van 0 tot en met $N - 1$. Elke student houdt een papiertje vast dat een lijst met gehele getallen bevat. Aan het begin van het spel is elk papiertje leeg, dus bevat het een lege lijst.

De studenten spelen het spel met M docenten, genummerd van 0 tot en met $M - 1$ en met het schoolhoofd. Het spel heeft M rondes, ook van 0 tot en met $M - 1$ genummerd. In ronde j komt docent j het klaslokaal binnen en gebeurt het volgende:

- Een aantal (mogelijk nul) studenten steekt hun hand op. Het is gegarandeerd dat in elke ronde minstens één student **niet** zijn hand opsteekt en dat elke student zijn hand **hoogstens één keer** opsteekt in de loop van het spel.
- De docent kijkt vervolgens naar de papiertjes die de studenten omhooghouden en mag ervoor kiezen om de papiertjes **aan te passen** bij de studenten die hun hand **niet** opgestoken hebben. Voor elk van deze studenten kan de docent de lijst op het papiertje vervangen door een nieuwe (mogelijk lege) lijst van gehele getallen. Elk van deze getallen moet tussen de 0 en 63 inclusief liggen en de lijst kan lengte hoogstens 63 hebben.
- Na deze aanpassingen verlaat docent j de kamer en geven de studenten hun papiertjes door volgens een geheime permutatie P . Dit betekent dat student i ($0 \leq i < N$) zijn papiertje aan student $P[i]$ geeft, waar $P[0], P[1], \dots, P[N - 1]$ **verschillende** gehele getallen tussen 0 en $N - 1$ inclusief zijn. De waardes van P **staan vast** over de hele loop van het spel, maar de docenten en het schoolhoofd kennen deze permutatie niet.

Nadat alle M rondes zijn doorlopen en de papiertjes voor de laatste keer volgens P zijn doorgegeven, komt het schoolhoofd binnen. Door alleen maar naar de papiertjes van de studenten te kijken, moet het hoofd voor elke student i ($0 \leq i < N$) het nummer van de ronde geven waarin student i zijn hand heeft opgestoken, of concluderen dat student i zijn hand nooit heeft opgestoken.

De docenten mogen niet met elkaar of met het hoofd communiceren, behalve door middel van de lijsten van gehele getallen die op de papiertjes staan. Elke docent weet het nummer van de ronde waarin ze het klaslokaal binnengaan.

Jouw taak is om een strategie te bedenken en te implementeren voor de docenten en het schoolhoofd, waarmee je voor elke student kan bepalen of die zijn hand heeft opgestoken, en zo ja, wanneer. Je score hangt af van de maximale lengte van een lijst die op een papiertje is geschreven over alle rondes. Een kleinere maximale lengte geeft een hogere of gelijke score.

Implementatiedetails

Je moet twee methodes implementeren, één voor de docenten en één voor het schoolhoofd.

De methode die je voor **docenten** moet implementeren is:

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

- N : het aantal studenten.
- M : het aantal rondes, en ook het aantal docenten.
- R : het nummer van de huidige ronde (van 0 tot en met $M - 1$).
- T : een (mogelijk lege) lijst die de indices bevat van de studenten die hun hand opsteken in de huidige ronde, in stijgende volgorde.
- A : een lijst van lengte N die de papiertjes beschrijft, waar $A[i]$ ($0 \leq i < N$) de lijst van gehele getallen is op het papiertje van student i aan het begin van de ronde.
- Deze methode wordt precies M keer aangeroepen per spel, in de volgorde $R = 0, 1, \dots, M - 1$.

De methode moet een lijst B van lengte N retourneren, die de papiertjes beschrijft na de aanpassingen van de docent, in hetzelfde formaat als A .

- Voor elke student i die zijn hand heeft opgestoken (dat wil zeggen, i komt voor in T) mag het papiertje niet zijn aangepast, dus $B[i] = A[i]$.
- Voor elke i zodat $0 \leq i < N$ moet de lengte van $B[i]$ niet groter zijn dan 63 en moet elk geheel getal in $B[i]$ tussen de 0 en 63 inclusief liggen.

De methode die je moet implementeren voor het **schoolhoofd** is:

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

- N, M , hetzelfde als hierboven.
- A : een lijst van lengte N , waar $A[i]$ de lijst van gehele getallen is op het papiertje dat student i vastheeft na alle M rondes.
- Deze methode wordt precies één keer aangeroepen per spel, na de laatste anroep van `process_step`.

De methode moet een lijst D van lengte N retourneren. Voor elke i zodat $0 \leq i < N$ moet het waar zijn dat

- $D[i] = j$ als student i zijn hand heeft opgestoken in ronde j , of

- $D[i] = -1$ als student i zijn hand nooit heeft opgestoken in de loop van het spel.

Het is niet toegestaan voor je programma om informatie door te geven tussen een aanroep van `process_step` en een andere, behalve door middel van de retourwaarde van `process_step`. Als je programma dit probeert te doen kan je score op CMS incorrect zijn en wordt deze mogelijk lager na het einde van de wedstrijd. Merk op dat de grader meerdere instanties van je programma tegelijk mag uitvoeren. Dit betekent dat aanroepen van `process_step` en `determine_steps` van hetzelfde spel uitgevoerd kunnen worden in verschillende instanties en dat aanroepen van `process_step` en `determine_steps` van verschillende spellen uitgevoerd kunnen worden in dezelfde instantie. Elk testgeval bevat hoogstens 5 spellen.

Beperkingen

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Elke student steekt zijn hand **hoogstens één keer** op in de loop van het spel. In andere woorden: voor elke student i is er hoogstens één ronde j waarin i zijn hand opeekt.
- In elke ronde is er minstens één student die zijn hand **niet** opsteekt.

Subtaken en Scores

Subtaak	Score	Extra beperkingen
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ voor alle $0 \leq i \leq N - 1$.
4	25	In elke ronde steekt hoogstens één student zijn hand op.
5	56	Geen extra beperkingen.

Voor elk testgeval is je score 0 (aangegeven met `Output isn't correct` op CMS) als de retourwaarde van een aanroep naar `process_step` ongeldig is of als de retourwaarde van een aanroep naar `determine_steps` ongeldig is.

Zo niet, laat C de maximale lengte zijn over **alle** lijsten in **alle** B die geretourneerd worden door `process_step`. De score voor een testgeval in een subtaak met score S is wordt dan berekend door $S \cdot X$, waar X afhangt van C volgens de volgende tabel:

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

Voorbeeld

Beschouw een scenario met $N = 4$ studenten, $M = 2$ docenten en een permutatie $P = [0, 3, 1, 2]$. Aanvankelijk zijn alle papiertjes leeg.

De eerste aanroep van de grader is:

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

Student 1 heeft hun hand opgestoken, dus kan hun papiertje niet aangepast worden. Docent 0 kan ervoor kiezen om $[10]$ op het papiertje van student 0 te schrijven, om $[1, 63, 4]$ op het papiertje van student 3 te schrijven en om het papiertje van student 2 leeg te laten. Om dit te doen moet de methode $B = [[10], [], [], [1, 63, 4]]$ retourneren.

Als docent 0 weg is, geven de studenten hun papiertjes door volgens P . Na het doorgeven zijn de papiertjes $A = [[10], [], [1, 63, 4], []]$.

De volgende aanroep van de grader is nu:

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

Studenten 0 en 3 hebben hun hand opgestoken, dus kunnen hun papiertjes niet aangepast worden. Docent 1 mag ervoor kiezen om $[0, 40]$ op het papiertje van student 1 te schrijven en om $[1, 50]$ op het papiertje van student 2 te schrijven. Om dit te doen moet de methode $B = [[10], [0, 40], [1, 50], []]$ retourneren.

Nadat docent 1 weg is, worden de papiertjes weer doorgegeven volgens P . Na het doorgeven zijn de papiertjes $A = [[10], [1, 50], [], [0, 40]]$.

Tenslotte roept de grader het volgende aan:

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

Deze methode moet een lijst $D = [1, 0, -1, 1]$ retourneren, omdat studenten 0 en 3 hun hand hebben opgestoken in ronde 1, student 1 hun hand heeft opgestoken in ronde 0 en student 2 hun hand nooit heeft opgestoken. In dit voorbeeld geldt $C = 3$.

Voorbeeldgrader

Invoerformaat:

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

Hier is Q is een lijst van lengte N , waar $Q[i] = j$ als student i zijn hand opsteekt tijdens ronde j , of $Q[i] = -1$ als student i zijn hand nooit op heeft gestoken tijdens het hele spel.

Uitvoerformaat:

Na elke aanroep van `process_step` print de voorbeeldgrader de inhoud van de papiertjes. Laat K de lengte van B zijn en $L[i]$ de lengte van $B[i]$ (voor elke $0 \leq i < K$). De voorbeeldgrader print B in het volgende formaat, **gevolgd door een lege regel**:

```
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]
```

De voorbeeldgrader geeft daarna de papiertjes door volgens de permutatie P . Als $K \neq N$, dan gebeurt dit niet, en print de voorbeeldgrader een error en beëindigt zichzelf.

Na het aanroepen van `determine_steps` print de voorbeeldgrader het volgende:

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

Hier is H de lengte van de lijst D die geretourneerd wordt door `determine_steps`.