

Jogo de Sala de Aula

Uma importante escola em Tashkent comemora seu aniversário promovendo um jogo entre alunos e professores. Há N alunos sentados na sala de aula, numerados de 0 a $N - 1$. Cada aluno segura um pedaço de papel que contém um vetor de números inteiros. Inicialmente, cada papel está em branco, portanto contém um vetor vazio.

Os alunos estão jogando o jogo com M professores, numerados de 0 a $M - 1$, e o diretor. O jogo é jogado em M etapas, também numeradas de 0 a $M - 1$. Na etapa j , o professor j entra na sala de aula e os seguintes eventos ocorrem:

- Alguns alunos (possivelmente nenhum) levantam a mão. É garantido que em cada etapa pelo menos um aluno **não** levanta a mão, e cada aluno pode levantar a mão **no máximo uma vez** durante todo o jogo.
- O professor observa os papéis que os alunos têm em mãos e pode **modificar** os papéis dos alunos que **não** levantaram a mão nesta etapa. Para cada aluno nessa situação, o professor pode substituir o conteúdo do papel por um novo vetor (possivelmente vazio) de números inteiros. Cada número inteiro deve estar entre 0 e 63, inclusive, e o vetor pode ter no máximo 63 elementos.
- Após essas modificações, o professor j sai e os alunos trocam seus trabalhos de acordo com uma permutação secreta P . Isto é, cada aluno i ($0 \leq i < N$) entrega seu trabalho ao aluno $P[i]$, onde $P[0], P[1], \dots, P[N - 1]$ são N inteiros **distintos** entre 0 e $N - 1$, inclusive. Os valores de P são **fixos** durante todas as etapas do jogo, mas os professores e o diretor não os conhecem.

Assim que todas M etapas forem concluídas e a última troca de acordo com P for realizada, o diretor entra. Observando apenas os papéis que os alunos possuem, o diretor deve determinar, para cada aluno i ($0 \leq i < N$), o número da etapa j em que o aluno i levantou a mão, ou concluir que o aluno i nunca levantou a mão.

Os professores não podem se comunicar uns com os outros ou com o diretor, exceto por meio de sequências de números inteiros escritas nos papéis. Cada professor sabe em qual etapa entra na sala de aula.

Sua tarefa é elaborar e implementar uma estratégia para os professores e o diretor que identifique corretamente se e quando cada aluno levantou a mão. Sua pontuação dependerá do comprimento máximo de qualquer sequência escrita em uma folha de papel: um comprimento máximo menor resulta em uma pontuação maior ou igual.

Detalhes de Implementação

Você precisa implementar dois procedimentos, um para os professores e outro para o diretor.

O procedimento que você deve implementar para os **professores** é:

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

- N : o número de alunos.
- M : o número de etapas e também o número de professores.
- R : o número da etapa atual (de 0 a $M - 1$).
- T : um vetor (possivelmente vazio) contendo os índices dos alunos que levantam a mão nesta etapa, ordenados em ordem crescente.
- A : um vetor de comprimento N descrevendo os papéis, onde $A[i]$ ($0 \leq i < N$) é o vetor de inteiros no papel que o aluno i possui no início da etapa.
- Este procedimento é chamado exatamente M vezes por jogo, na ordem $R = 0, 1, \dots, M - 1$.

O procedimento deve retornar um vetor B de comprimento N , representando os trabalhos após as modificações do professor, no mesmo formato que A .

- Para cada aluno i que levantou a mão (isto é, i aparece em T), o papel não deve ser alterado, então $B[i] = A[i]$.
- Para cada i tal que $0 \leq i < N$, o comprimento de $B[i]$ não deve exceder 63, e cada inteiro em $B[i]$ deve estar entre 0 e 63, inclusive.

O procedimento que você deve implementar para o **diretor** é:

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

- N, M : os mesmos que acima.
- A : um vetor de comprimento N , onde $A[i]$ é o vetor de inteiros no papel segurado pelo aluno i após todos os M passos.
- Este procedimento é chamado exatamente uma vez por jogo, após a chamada final para `process_step`.

O procedimento deve retornar um vetor D de comprimento N . Para cada i tal que $0 \leq i < N$, deve valer que

- $D[i] = j$ se o aluno i levantou a mão durante a etapa j , ou

- $D[i] = -1$ se o aluno i nunca levantou a mão durante todo o jogo.

Seu programa não está autorizado a armazenar ou transmitir qualquer informação de uma chamada a `process_step` para outra, exceto através do valor de retorno de `process_step`.

Se o seu programa tentar fazer isso, sua pontuação no CMS poderá estar incorreta e poderá ser reduzida após o término da competição. Observe que o corretor pode executar várias instâncias do seu programa simultaneamente. Isso significa que chamadas para `process_step` e `determine_steps` do mesmo jogo podem ser executadas em instâncias diferentes, e chamadas para `process_step` e `determine_steps` de jogos diferentes podem ser executadas na mesma instância. Cada caso de teste inclui até 5 jogos.

Restrições

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Cada aluno levanta a mão **no máximo uma vez** durante todo o jogo. Em outras palavras, para cada aluno i , há no máximo uma etapa j na qual i levanta a mão.
- Em cada etapa, pelo menos um aluno **não** levanta a mão.

Subtarefas e Pontuação

Subtarefa	Pontuação	Restrições adicionais
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ para todo $0 \leq i \leq N - 1$.
4	25	No máximo um aluno levanta a mão em cada etapa.
5	56	Sem restrições adicionais.

Para cada caso de teste, sua pontuação é 0 (relatada como `Output isn't correct` no CMS) se o valor de retorno de qualquer chamada ao procedimento `process_step` for inválido ou se o valor de retorno de qualquer chamada ao procedimento `determine_steps` estiver incorreto.

Caso contrário, seja C o comprimento máximo de **qualquer** vetor em **qualquer** B retornado por `process_step`. Então, a pontuação para um caso de teste em uma subtarefa com pontuação S é calculada como $S \cdot X$, onde X depende de C de acordo com a seguinte tabela:

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

Exemplo

Considere um cenário com $N = 4$ alunos, $M = 2$ professores e permutação $P = [0, 3, 1, 2]$. Inicialmente, todos os papéis estão em branco.

O corretor primeiro chama:

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

O aluno 1 levantou a mão, portanto seu trabalho não pode ser modificado. O professor 0 pode decidir escrever $[10]$ no papel do aluno 0, escrever $[1, 63, 4]$ no papel do aluno 3 e deixar o papel do aluno 2 em branco. Para fazer isso, o procedimento deve retornar $B = [[10], [], [], [1, 63, 4]]$.

Após o professor 0 sair, os alunos trocam seus papéis de acordo com P . Após a troca, os papéis são $A = [[10], [], [1, 63, 4], []]$.

O corretor agora chama:

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

Os alunos 0 e 3 levantaram a mão, portanto seus trabalhos não podem ser modificados. O professor 1 pode decidir escrever $[0, 40]$ no papel do aluno 1 e escrever $[1, 50]$ no papel do aluno 2. Para fazer isso, o procedimento deve retornar $B = [[10], [0, 40], [1, 50], []]$.

Após o professor 1 sair, os papéis são trocados novamente de acordo com P . Após a troca, os papéis são $A = [[10], [1, 50], [], [0, 40]]$.

Finalmente, o corretor chama:

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

Este procedimento deve retornar o vetor $D = [1, 0, -1, 1]$ pois os alunos 0 e 3 levantaram a mão na etapa 1, o aluno 1 levantou a mão na etapa 0 e o aluno 2 nunca levantou a mão. Neste

exemplo, $C = 3$.

Corretor exemplo

Formato de entrada:

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

Aqui, Q é um vetor de comprimento N , onde $Q[i] = j$ se o aluno i levantar a mão durante a etapa j , ou $Q[i] = -1$ se o aluno i nunca levantar a mão durante todo o jogo.

Formato de saída:

Após cada chamada a `process_step`, o corretor exemplo exibe o conteúdo dos trabalhos. Seja K o comprimento de B e $L[i]$ o comprimento de $B[i]$ (para cada $0 \leq i < K$). O corretor exemplo imprime B no seguinte formato, **seguido de uma linha em branco**:

```
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]
```

O corretor exemplo então troca os papéis de acordo com a permutação P . Se $K \neq N$, o corretor exemplo imprime uma mensagem de erro e termina a execução.

Após chamar `determine_steps`, o corretor exemplo produz a seguinte saída:

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

Aqui, H é o comprimento do vetor D retornado por `determine_steps`.