

课堂游戏

塔什干的一所著名高中为庆祝校庆，举办了一场学生和老师之间的游戏。教室里有 N 位同学，编号为 0 到 $N - 1$ 。每位同学手里都有一张纸，纸上写着一个整数数组。最开始时，每张纸都是空白的，也就是上面写着一个空数组。

同学们与 M 位老师（编号为从 0 到 $M - 1$ ）以及校长一起做游戏。游戏总共有 M 步，步数也从 0 编号到 $M - 1$ 。在第 j 步时，第 j 位老师走进教室，接下来就会发生下面的事情：

- 有某些（可能为零）同学举手。保证在每一步中，至少有一位同学**不**举手，而且在整个游戏过程中，每位同学**最多举手一次**。
- 老师查看同学们当前手中的纸，并且可以**修改**在这一步中**没有**举手的同学的纸。对于每位这样的同学，教师可以把她手上纸的内容换成一个全新的（可能为空）整数数组。数组中的每个整数必须在 0 到 63 之间（包括 0 和 63 ），而且数组最多包含 63 个元素。
- 在完成这些修改后，第 j 位老师离开教室，同学们根据一个秘密的置换 P 来交换手中的纸。也就是说，每位同学 i ($0 \leq i < N$) 将其手中的纸交给同学 $P[i]$ ，这里 $P[0], P[1], \dots, P[N - 1]$ 是 N 个 0 到 $N - 1$ 之间（包括 0 和 $N - 1$ ）的**互不相同**的整数。 P 中的值在游戏的所有步骤中都是**固定**的，但老师们和校长并不知道这些值。

当所有 M 步结束，并且最后一次根据 P 所做的交换也完成后，校长走进教室。仅通过观察同学们纸上的内容，校长必须确定每位同学 i ($0 \leq i < N$) 是在第几步举的手，或者从来没有举过手。

除了通过纸上所写的整数数组以外，老师们不能与其他老师或校长用其他途径互通信息。每位老师都知道自己是在哪一步走进了教室。

你的任务是，为老师们和校长设计并实现一套策略，以正确判断各位同学是否举过手以及在哪一步中举手了。你的得分将取决于所有写在纸上的数组的最大长度：最大长度越短的话，得分会越高或持平。

实现细节

你需要实现两个函数，一个给老师们用，一个给校长用。

你需要为**老师们**实现的函数是：

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

- N : 同学的数量。
- M : 步骤的数量, 同时也是教师的数量。
- R : 当前步骤的编号 (从 0 到 $M - 1$)。
- T : 一个 (可能为空的) 数组, 其中包含在当前步骤中举手的同学的编号, 并且按照升序给出。
- A : 一个长度为 N 的数组, 给出纸上的内容。其中 $A[i]$ ($0 \leq i < N$) 表示步骤开始时同学 i 手中纸上的整数数组。
- 在每局游戏中, 该函数将以 $R = 0, 1, \dots, M - 1$ 的顺序, 恰好被调用 M 次。

该函数应返回一个长度为 N 的数组 B , 给出经过老师修改后的纸上的内容, 其格式与 A 相同。

- 对于每位举手的同学 i (也就是说, i 出现在 T 里面), 他手中纸上的内容不得更改, 也就是 $B[i] = A[i]$ 。
- 对于每个满足 $0 \leq i < N$ 的 i , $B[i]$ 的长度不能超过 **63**, 而且 $B[i]$ 中的每个整数必须在 **0** 到 **63** 之间 (包括 **0** 和 **63**)。

你需要为**校长**实现的函数是:

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

- N 、 M : 同上。
- A : 一个长度为 N 的数组, 其中 $A[i]$ 为全部 M 步结束后同学 i 手中纸上的整数数组。
- 该函数在每局游戏中恰好被调用一次, 而且是在 `process_step` 的最后一次调用之后。

该函数必须返回一个长度为 N 的数组 D 。对于每个满足 $0 \leq i < N$ 的 i , 该数组必须满足:

- 如果同学 i 在第 j 步举了手, 则 $D[i] = j$, 或者
- 如果同学 i 在整个游戏过程中从来没有举过手, 则 $D[i] = -1$ 。

你的程序不能在 `process_step` 的不同调用之间存储或传递任何信息, 除非是通过 `process_step` 的返回结果。 如果你的程序试图这样做, CMS 中的分数可能会不正确, 并且可能在比赛结束后被扣减。请注意, 评测程序可能会同时运行你的程序的多个实例。这意味着, 同一局游戏中的 `process_step` 和 `determine_steps` 调用可能在不同的实例中执行, 而不同游戏的 `process_step` 和 `determine_steps` 调用可能在同一个实例中执行。每个测试用例最多包含 **5** 局游戏。

约束条件

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- 在每局游戏的整个过程中, 每位同学**最多举手一次**。也就是说, 每位同学 i 最多在某一个步骤 j 中举手。
- 在每一步中, 至少有一位同学**没有**举手。

子任务与评分

子任务	分数	额外的约束条件
1	4	$M = 1$
2	6	$N = 2$
3	9	对于每个 $0 \leq i \leq N - 1$ ，都有 $P[i] = i$ 。
4	25	每一步最多有一位同学举手。
5	56	没有额外的约束条件。

对于每个测试用例，如果 `process_step` 的任何一次调用的返回结果不符合要求，或者 `determine_steps` 的任何一次调用的返回结果不对，则你的得分为 0（在 CMS 中将报告为 `Output isn't correct`）。

否则，令 C 为 `process_step` 所返回的**全部** B 中的**全部**数组的**最大长度**。然后，在分值为 S 的某个子任务的一个测试用例上，得分将被算为 $S \cdot X$ ，这里 X 将根据下表由 C 算出：

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

例子

设想一个有 $N = 4$ 位同学， $M = 2$ 位老师，而且置换 $P = [0, 3, 1, 2]$ 的场景。最开始时，所有的纸都是空白的。

评测程序首先调用：

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

同学 1 已经举手了，因此他手中的纸不能被修改。老师 0 决定在同学 0 的纸上写 `[10]`，在同学 3 的纸上写上 `[1, 63, 4]`，并且保持同学 2 的纸为空白。为此，该函数必须返回 $B = [[10], [], [], [1, 63, 4]]$ 。

在老师 0 离开后，同学们根据 P 交换纸张。交换后，纸张内容变为 $A = [[10], [], [1, 63, 4], []]$ 。

评测程序现在调用：

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

同学 0 和 3 已经举手，因此他们手上的纸不能被修改。老师 1 决定在同学 1 的纸上写 $[0, 40]$ ，在同学 2 的纸上写 $[1, 50]$ 。为此，该函数必须返回 $B = [[10], [0, 40], [1, 50], []]$ 。

老师 1 离开后，纸张再次根据 P 进行交换。交换后，纸张内容变为 $A = [[10], [1, 50], [], [0, 40]]$ 。

最后，评测程序调用：

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

该函数必须返回数组 $D = [1, 0, -1, 1]$ ，因为同学 0 和 3 在第 1 步举了手，同学 1 在第 0 步举了手，而同学 2 从来没有举过手。在这个例子中， $C = 3$ 。

评测程序示例

输入格式：

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

其中， Q 是一个长度为 N 的数组，如果同学 i 在第 j 步举了手，则 $Q[i] = j$ ；如果同学 i 在整个游戏过程中从来没有举过手，则 $Q[i] = -1$ 。

输出格式：

在每次调用 `process_step` 后，评测程序示例都会输出纸张的内容。令 K 为 B 的长度， $L[i]$ 为 $B[i]$ 的长度（对于每个 $0 \leq i < K$ ）。评测程序示例将按照如下格式输出 B ，并在**最后跟着一个空行**：

```
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]
```

随后，评测程序示例将根据置换 P 交换纸张。如果 $K \neq N$ ，评测程序示例将输出一条错误信息并且终止。

在调用 `determine_steps` 后，评测程序示例输出：

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

其中, H 是 `determine_steps` 所返回的数组 D 的长度。