Minperm

The beautiful city of *Julcvari* decided to host a city tour(online, of course). They have n different attractions, the i'th one having a beauty equal to i. An initial order for the presentation of the attractions has been selected, but the people of *Julcvari* started worrying that it might not be the best one. They believe that the best order for visiting the attractions is the one for which the number of pairs of attractions (i, j), where i was visited before j but i is more beautiful than j is minimised. But organizing events is not so easy, a lot of paperwork needs to be done in order to change something like this. So, the host managed to get k swap plans: let $l_1, l_2, \ldots, l_k$ be an array of integers representing the swap plans. The host can perform any number of swaps on the initial order, such that the swapped positions are at a distance equal to one of the elements from array l. There is not much time left, so the people of *Julcvari* ask you for help in finding the best order in which the attractions are presented.

In other words, a permutation of size n and an array of size k with distinct elements are given. You can only perform swaps between positions i and j such that $|j - i|$ is equal to an element of l. What is the permutation with the minimum number of inversions that can be obtained?

Input data

The first line of the input contains n and k. The second line of the input contains n integers, representing the initial order of the attractions. The third line of the input contains k integers, representing the k available swap plans.

Output data

The only line of the output should contain n elements, representing the best order for showing the attractions.

Restrictions

$1 \leq k \leq n \leq 5000$

Subtask 1(10 points)

$1 \leq n \leq 8$

Subtask 2(30 points)

$1 \leq n \leq 100$

Subtask 3(60 points)

$1 \leq n \leq 5000$

Examples

input	output	explanations

`8 1`
`1 4 3 5 7 2 8 6`
`8`
	`1 4 3 5 7 2 8 6`	`Here, the only possible swap is between positions with difference 8. Since there are no positions like that, we cannot change the order`

`8 7`
`2 4 1 5 3 6 7 8`
`7 6 3 8 2 5 1`
	`1 2 3 4 5 6 7 8`	`Here, we can swap positions with the difference equal to any of the values from l. Since we have 1, we can perform any swap, and thus sort the permutation`