Three Permutations

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	64 mebibytes

You are given three permutations a, b, and c of length n. Applying permutation c to array a produces an array d such that $d[c[i]] = a[i]$ for all i.

Before applying c to a, you may perform an operation on permutation c any number of times. In one operation, you can choose two indices i and j and swap $c[i]$ and $c[j]$.

For each k from 0 to n, find the minimum number of operations after which applying the modified permutation c to a produces an array that differs from b in exactly k positions, or determine that it is impossible.

Input

The first line contains an integer t ($1 \le t \le 5 \cdot 10^4$), the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 10^6$).

The next three lines contain permutations a, b, and c of length n. Each permutation contains every integer from 1 to n exactly once.

The sum of n over all test cases does not exceed 10^6.

Output

For each test case, output $n + 1$ integers, where the k-th of them is the minimum number of operations needed to obtain exactly k differences, or -1 if it is impossible.

Example

standard input	*standard output*
2 3 1 2 3 3 1 2 2 3 1 2 2 1 1 2 1 2	0 -1 1 2 1 -1 0

Note

In the first test case, the array is $a = [1, 2, 3]$. After applying the permutation $c = [2, 3, 1]$, the element from the 1-st position moves to the 2-nd, from the 2-nd to the 3-rd, and from the 3-rd to the 1-st, and the resulting array is $d = [3, 1, 2]$. In this case, it coincides with the array b. So, the answer for $k = 0$ differences is 0 operations.