
Binary Tree Restoring

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

Given two depth-first-search (DFS) sequences of a binary tree, can you find a binary tree which satisfies both of the DFS sequences?

Recall that a binary tree is a tree in which each vertex has at most two children, and the depth-first search is a tree traversing method which starts at the root and explores as far as possible along each branch before backtracking.

Input

There are multiple test cases. The first line of input contains an integer T , indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \leq n \leq 10^5$), indicating the number of vertices in the binary tree.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq n$, $\forall 1 \leq i < j \leq n, a_i \neq a_j$), indicating the first DFS sequence of the binary tree.

The third line of each test case contains n integers $b_1, b_2, \dots, b_n$ ($1 \leq b_i \leq n$, $\forall 1 \leq i < j \leq n, b_i \neq b_j$), indicating the second DFS sequence of the binary tree.

It is guaranteed that the sum of n over all test cases does not exceed 10^6 , and there always exists at least one possible binary tree.

We kindly remind you that this problem contains large I/O file, so it's recommended to use a faster I/O method. For example, you can use `scanf/printf` instead of `cin/cout` in C++.

Output

For each test case, output one line which contains n integers separated by one space. The i -th integer indicates the father of the i -th vertex in the binary tree which satisfies both of the DFS sequence. If the i -th vertex is the root of the binary tree, output 0 as its father. If there are multiple valid answers, you can output any of them.

Please, DO NOT print extra spaces at the end of each line, or your program may get a “wrong answer” verdict as this problem is special judged.

Example

standard input	standard output
2	3 4 0 3 4 1
6	0 1 2
3 4 2 5 1 6	
3 4 5 2 1 6	
3	
1 2 3	
1 2 3	