

#include <bits/stdc++.h>

#include <queue>

**using** **namespace** std;

#define SZ (31)

int lch[SZ], rch[SZ];

int border[SZ], morder[SZ];

int ans[SZ];

int ans\_len = 0;

int build\_tree(int s1, int e1, int s2, int e2) {

**if** (s1 >= e1 || s2 >= e2) **return** -1; *// 空树情况*

int root = border[e1 - 1]; *// 后序遍历的最后一个元素是根*

int mid = -1;

**for** (int i = s2; i < e2; i++) {

**if** (morder[i] == root) {

mid = i;

**break**;

}

}

*// 左子树的后序遍历起止位置和中序遍历起止位置*

lch[root] = build\_tree(s1, s1 + mid - s2, s2, mid);

*// 右子树的后序遍历起止位置和中序遍历起止位置*

rch[root] = build\_tree(s1 + mid - s2, e1 - 1, mid + 1, e2);

**return** root;

}

void bfs(int root) {

queue<int> q;

q.push(root);

int node;

**while** (!q.empty()) {

node = q.front();

ans[ans\_len++] = node;

q.pop();

**if** (lch[node] != -1)

q.push(lch[node]);

**if** (rch[node] != -1)

q.push(rch[node]);

}

}

int main() {

ios\_base::sync\_with\_stdio(false);

cin.tie(NULL);

int n;

cin >> n;

memset(lch, -1, **sizeof**(int) \* n);

memset(rch, -1, **sizeof**(int) \* n);

**for** (int i = 0; i < n; i++)

cin >> border[i];

**for** (int i = 0; i < n; i++)

cin >> morder[i];

int root = build\_tree(0, n, 0, n);

bfs(root);

**for** (int i = 0; i < n; i++) {

cout << ans[i];

**if** (i != n - 1) {

cout << ' ';

}

}

**return** 0;

}