G. An express train to reveries

Time limit: 1s Memory limit: 256 MB

Sengoku still remembers the mysterious "colourful meteoroids" she discovered with Lala-chan when they were little. In particular, one of the nights impressed her deeply, giving her the illusion that all her fancies would be realized.

On that night, Sengoku constructed a permutation $p_1, p_2, ..., p_n$ of integers from 1 to n inclusive, with each integer representing a colour, wishing for the colours to see in the coming meteor outburst. Two incredible outbursts then arrived, each with n meteorids, colours of which being integer sequences $a_1, a_2, ..., a_n$ and $b_1, b_2, ..., b_n$ respectively. Meteoroids' colours were also between 1 and n inclusive, and the two sequences were not identical, that is, at least one i $(1 \le i \le n)$ exists, such that $a_i \ne b_i$ holds.

Well, she almost had it all — each of the sequences a and b matched exactly $n - 1$ elements in Sengoku's permutation. In other words, there is exactly one i $(1 \le i \le n)$ such that $a_i \ne p_i$, and exactly one j $(1 \le j \le n)$ such that $b_j \ne p_j$.

For now, Sengoku is able to recover the actual colour sequences a and b through astronomical records, but her wishes have been long forgotten. You are to reconstruct any possible permutation Sengoku could have had on that night.

Input

The first line of input contains a positive integer n $(2 \le n \le 1\,000)$ — the length of Sengoku's permutation, being the length of both meteor outbursts at the same time.

The second line contains n space-separated integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le n)$ — the sequence of colours in the first meteor outburst.

The third line contains n space-separated integers $b_1, b_2, ..., b_n$ $(1 \le b_i \le n)$ — the sequence of colours in the second meteor outburst. At least one i $(1 \le i \le n)$ exists, such that $a_i \ne b_i$ holds.

Output

Output n space-separated integers $p_1, p_2, ..., p_n$, denoting a possible permutation Sengoku could have had. If there are more than one possible answer, output any one of them.

Input guarantees that such permutation exists.

Examples

input

```
5
1 2 3 4 3
1 2 5 4 5
```

output

```
1 2 5 4 3
```

input

```
5
4 4 2 3 1
5 4 5 3 1
```

output

```
5 4 2 3 1
```

input

```
4
1 1 3 4
1 4 3 4
```

output

```
1 2 3 4
```

Note

In the first sample, both 1, 2, 5, 4, 3 and 1, 2, 3, 4, 5 are acceptable outputs.

In the second sample, 5, 4, 2, 3, 1 is the only permutation to satisfy the constraints.