Problem Permuturns

Input file	`stdin`
Output file	`stdout`

(a) Past (b) Present (c) Future

Figure 1: The tarot reading of the problem *Permuturns*

Alin and Blin are playing a game called `Palatro`. Each of them has N cards numbered from 1 to N. Alin writes the number A_i on card i, and Blin writes the number B_i on card i. Each player must write every number from 1 to N on exactly one card (thus A and B are permutations).

Clin is the referee of the game. He decides the order in which the two players must reveal their cards, giving N requests of the form:

"Show the card with index C_1", then "Show the card with index C_2", and so on. Here $(C_1, C_2, \ldots, C_N)$ represents a permutation of the numbers from 1 to N.

At the beginning, Clin considers the first card shown (C_1) to be the winning card. Each time a new card C_i is revealed, Clin checks the numbers written on it: if both Alin's number and Blin's number are greater than those on the current winning card, then card C_i becomes the new winning card.

More formally, Clin uses the following algorithm to determine the winning card:

```
winner = C[1]
for i=2...N:
    if A[C[i]] > A[winner] and B[C[i]] > B[winner]:
        winner = C[i]
```

Task

For each i from 1 to N, determine in how many distinct ways Clin can give the requests such that the winning card is card i. Since the answer may be very large, output it modulo $10^9 + 7$.

Input data

The first line contains the integer N. The second line contains N integers, the elements of permutation A. The third line contains N integers, the elements of permutation B.

Output data

Print N integers, separated by spaces, the answer for each i.

Constraints

- $1 \le N \le 200\,000$.

#	Points	Constraints
1	8	$A = 1, 2, \ldots, N$ and $B = N-1, N-2, \ldots, 1, N$
2	9	$1 \le N \le 10$
3	14	$1 \le N \le 20$
4	17	$1 \le N \le 500$
5	21	A and B are generated uniformly at random
6	15	$1 \le N \le 5000$
7	16	No additional restrictions

Examples

stdin	stdout	Explanations
3 2 1 3 3 2 1	4 0 2	$P = [1,2,3] \Rightarrow$ At the end of the algorithm, pos = 1 $P = [1,3,2] \Rightarrow$ At the end of the algorithm, pos = 1 $P = [2,1,3] \Rightarrow$ At the end of the algorithm, pos = 1 $P = [2,3,1] \Rightarrow$ At the end of the algorithm, pos = 1 $P = [3,1,2] \Rightarrow$ At the end of the algorithm, pos = 3 $P = [3,2,1] \Rightarrow$ At the end of the algorithm, pos = 3