D Deck Randomisation

Time limit: 1s



CC-BY-SA 4.0 By Alexey Musulev on wikimedia.org

Alice and Bob love playing Don'tminion, which typically involves a lot of shuffling of decks of different sizes. Because they play so often, they are not only very quick at shuffling, but also very consistent. Each time Alice shuffles her deck, her cards get permuted in the same way, just like Bob always permutes his cards the same way when he shuffles them. This isn't good for playing games, but raises an interesting question.

They know that if they take turns shuffling, then at some point the deck will end up ordered in the same way as when they started. Alice shuffles once first, then Bob shuffles once, then Alice shuffles again, et cetera. They start with a sorted deck. What they do not know, however, is how many shuffles it will take before the deck is sorted again.

Can you help them compute how many shuffles it will take? As Alice and Bob can only do 10^{12} shuffles in the limited time they have, any number strictly larger than this should be returned as `huge` instead.

Input

- The first line contains a single integer $1 \leq n \leq 10^5$, the number of cards in the deck.
- The second line contains n distinct integers $1 \leq a_1, a_2, \ldots, a_n \leq n$, where a_i is the new position of the card previously at position i when Alice shuffles the deck.
- The third line contains n distinct integers $1 \leq b_1, b_2, \ldots, b_n \leq n$, where b_i is the new position of the card previously at position i when Bob shuffles the deck.

Output

- Output a single positive integer $m > 0$, the minimal number of shuffles required to sort the deck, or `huge` when this number is strictly larger than 10^{12}.

Sample Input 1	Sample Output 1
3 2 3 1 3 1 2	2

Sample Input 2	Sample Output 2
6 5 1 6 3 2 4 4 6 5 1 3 2	5

Sample Input 3	**Sample Output 3**
8 1 4 2 6 7 8 5 3 3 6 8 4 7 1 5 2	10