Shoes

I know a girl who has lots of shoes. By lots, I mean bucket loads of shoes - STACKS of shoes, if you will. Actually, that's exactly what they look like: five stacks of shoes, piled so high you can barely count them all!

The problem with having so many shoes is sometimes they get a little disorganized, and in this case, she can't find a single pair of shoes to wear. The problem? The left and right shoes have got all split up, and she can't figure out which pair of shoes is the most convenient to get to. In order to get to a shoe, she needs to move all the shoes above it (her version of "moving shoes" is actually picking them up and throwing them at other people), and because she's so fashionable, she needs to wear two matching shoes in order to leave the house. Types of shoes are represented by positive integers - the left and right shoes of a pair are represented by the same number. For a given shoe type, there are never more than two such shoes in the same test case, and there are never two shoes of the same type in a single stack.

Input Format
The input consists of a series of test cases, where each test case is five non-empty lines. On each line will be a nonempty sequence of integers, which represent shoes from the various pairs. The first number in a line is the highest of that stack, and the last number is at the bottom of that stack. Input ends on EOF.

Constraints
There are no more than 100000 shoes for any given test case, and no shoe is represented by an integer greater than 100000.

A warning to Java users: my friend has a lot of shoes. You might want to use the BufferedReader class to read the input.

Output Format
Output the minimum number of shoes she needs to move before finding a matching pair, one line per test case.

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