CSC148, Exercise 1
Stacks and Queues **due January 26, 10 p.m.**

Question 1

Consider the list `[5, 4, 3, 1, 2]`. Using a stack s and a single iteration through the list, we can output the elements of the list in sorted order, as follows:

- Push 5 onto s
- Push 4 onto s
- Push 3 onto s
- Output 1
- Output 2
- Pop from s and output the 3
- Pop from s and output the 4
- Pop from s and output the 5

We would say here that we can output all 5 elements of the list in sorted order.

Now consider another example, this time for the list `[4, 5, 2, 1, 3]`. We cannot output the entire list in sorted order this time using a stack, but we **can** output the **smallest** 3 elements as follows:

- Push 4 onto s
- Push 5 onto s
- Push 2 onto s
- Output 1
- Pop from s and output the 2
- Output 3

... and that's all we can do. If we were to pop from s now, we'd get 5, when what we really want to output next is 4. We would say here that 3 elements of the list can be output in sorted order.

In file `q1.py`, complete the `solve` function. It takes a list, and returns the maximum number of elements of the list (starting from `1`) that can be output in sorted order by using one auxiliary stack as in the examples above. If the list is of length n, then its elements are guaranteed to be a permutation of the n integers from 1 to n. Remember never to access private instance variables of the stack! In particular, you won't know what stack we use for testing — you can be sure that it won't use `_data`, though!

Question 2

Suppose that we know the order that elements are added into an ADT, and the order that those same elements come out of that ADT. For example, we might know that elements `1, 2, 3` (in that order) are added to an ADT, and that the order they come out is `3, 2, 1` (in that order). We would then conclude that this ADT could be a stack (not a queue). As another example, if `1, 2, 1` are added to the ADT, and come out in the order `1, 2, 1`, then it's possible that the ADT is a stack **or** a queue.

In file `q2.py`, complete the `solve` function. It takes two lists: a list giving the order that elements are added to an ADT (the elements are added from the beginning of the list to the end), and a list giving the order that the elements come out of the ADT. The two lists are guaranteed to have the same number of elements. Based on the provided orderings, you should return one of four values:

- 0 means that the ADT is not a stack or queue,
- 1 means that the ADT could be a stack (but not a queue),
- 2 means that the ADT could be a queue (but not a stack),
- and 3 means that the ADT could be a stack or queue

For this question, you are not allowed to use a `Stack` or `Queue` or auxiliary lists or dictionaries; you should solve it directly using the two given lists.

Testing

We have provided one test case for Question 1 and Question 2 in `q1test.py` and `q2test.py`, respectively. You're encouraged to write further tests to help increase your confidence in your functions, though any tests that you write are **not** being marked.

Submitting

You are required to do this exercise on your own. Please submit `q1.py` and `q2.py` on MarkUs. No other files are accepted.

Your grade on these questions will be determined by our test cases. Design and other factors are not marked.