CF1698D - Fixed Point Guessing

`https://codeforces.com/contest/1698/problem/D`

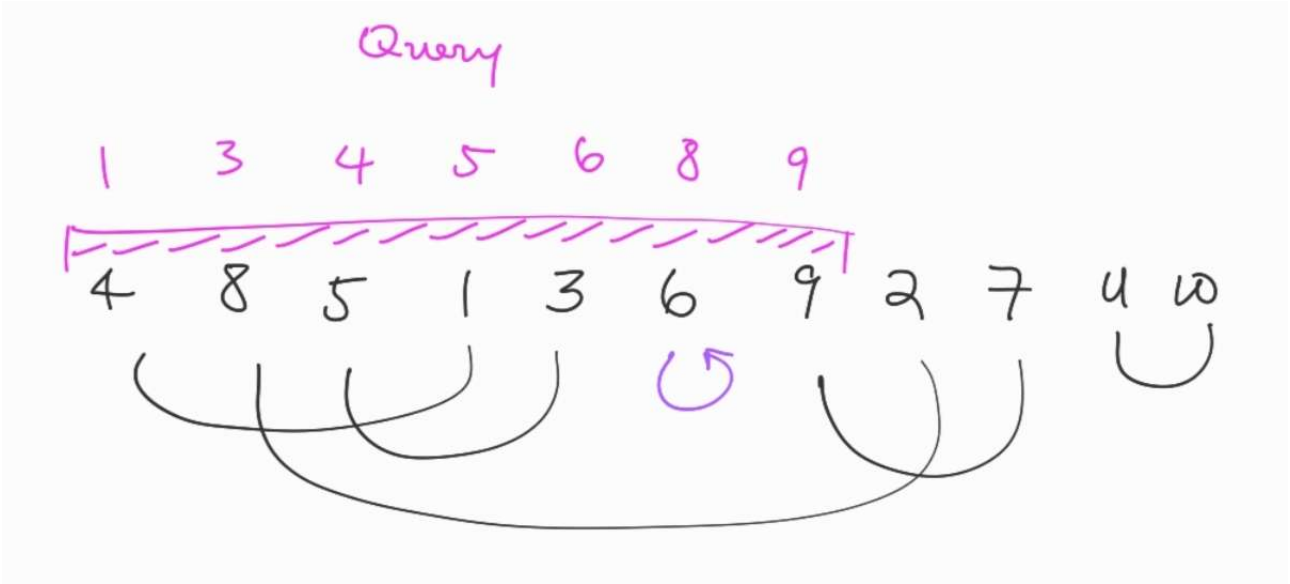
The first thing that catches my eye is that $\log_2(10^4) < 15$. Binary search is quite common in interactive problems, and since the constraints permit it, that might be the solution here.

If I can answer the question, "Is the fixed point in $[1, m]$?" in a single query, then I can binary search for the location of the fixed point; it's the first m for which the answer to that question is yes.

Anyway, I return to general exploration, but I keep in mind that the binary search path seems fruitful and that I should be on the lookout for insights that contribute towards my Big If.

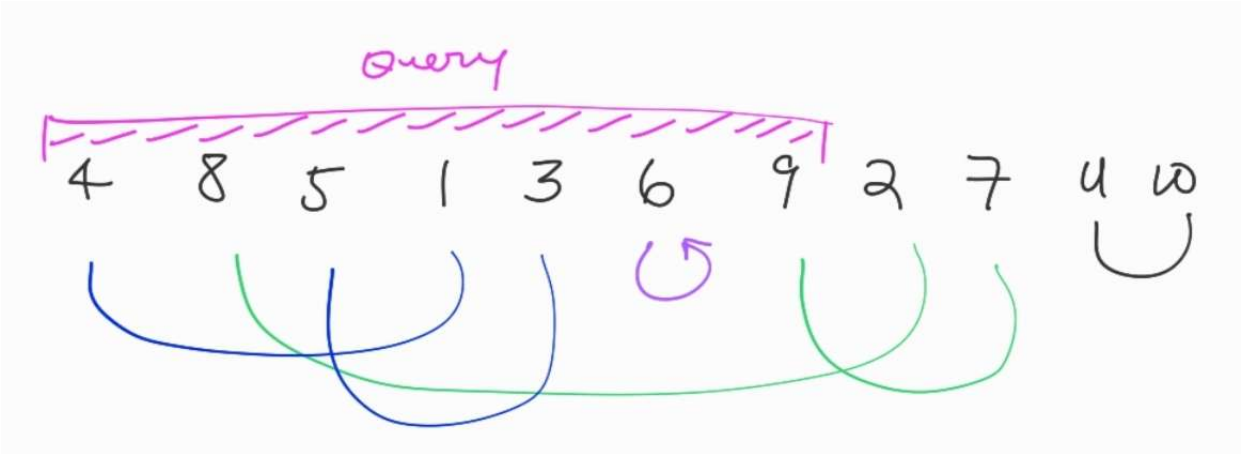
So, I try the problem-solving strategy of playing around with an actual concrete case, with pencil and paper. I try various small arrays on pen and paper, and I try different queries, and I see if I can spot a pattern.

Consider the following array with 11 elements, where I drew lines to connect the swapped pairs of elements. I also drew a query for $[1, 7]$ (and wrote the response).



After having it visually laid out like that, my eyes are drawn to the fact that for some query, I can partition each swapped pair into one of two categories:

- Both elements are in the range of the query
- One element is in the range of the query, and the other is not



For convenience, let's call these *enclosed pairs* (links colored blue) and *dangling elements* (links colored green). Apart from these two categories, the only other possibility for an element is that it is a fixed point. For the $[1, 7]$ query:

- $1, 3, 4, 5$ are from enclosed pairs
- $8, 9$ are dangling elements
- 6 is the fixed point

With these concrete numbers, I make some observations that just pop out at me.

- The elements from enclosed pairs $1, 3, 4, 5$ are in $[1, 7]$
- The fixed point 6 is also in $[1, 7]$
- The dangling elements $8, 9$ are **not** in $[1, 7]$
- There are an even number of elements from enclosed pairs

I conjecture that all of these should hold in more general cases, and after thinking about it, I realize that all these conjectures are true! The statements of the generalizations + their proofs are left as an exercise to the reader ;)

What does this tell me? **I can identify all the dangling elements** since they are simply the ones outside the range of the query that showed up in the response anyway.

Unfortunately, for the remaining elements, I can't tell if it's from an enclosed pair or if it's the fixed point. But after trying a few more example queries, I realize something:

- Perform a query, and remove all dangling elements from the response.
- If the number of remaining elements is odd, then the fixed point is **sure to be in this range**
- If even, then it is for sure not in this range

For example, for the query on $[1, 7]$ in our example, after removing the dangling elements, we are left with $1, 3, 4, 5, 6$. There are 5 remaining element, and 5 is odd, so we conclude that the fixed point is in this range (correct).

And I realize why—the only elements left are from enclosed pairs, or are the fixed point. But the number of elements from enclosed pairs is always even. So the *parity* of the number of remaining elements is solely decided by the presence of the fixed point.

Aha! This leads to a simple test that allows us to determine if the fixed point is in some range $[1, m]$ using a single query. Thus, we can binary search for the location of the fixed point. This completes the solution.

Useful takeaways for this problem:

- Drawing inspiration from solutions to past related problems
- Playing with concrete cases in order to find generalizable patterns
- Finding patterns visually, by making a diagram
- Minimizing mental load by blackboxing standard algorithms
- Using parity