Editorial: Battle Cows - 1951B

Avighna

June 2024

With the right data structure, we can pull off a brute-force solution.

> Brute-force: Calculate the number of times your cow wins without any swaps. Then try swapping your cow with each cow and calculate the number of times you win in each of those cases. The answer is the maximum of the initial value (without swaps) and the maximum of each case.

If we could (somehow) figure out the maximum value in a range of our array in $\mathcal{O}(T_1)$, swap values in $\mathcal{O}(T_2)$, and be able to find the location of the first number greater than x in $\mathcal{O}(T_3)$, we'd be able to solve our problem in $\mathcal{O}(n(T_1 + T_2 + T_3))$. The algorithm to do so is given below.

Swap your cow with every other cow. Now, if any cow behind your cow has a greater rating than yours, you won't be able to win any battles. If not, then you've already won 1 battle given that your cow isn't the first cow: the battle between your cow and the winner of the battles among cows before yours. You'll keep winning battles with cows ahead of yours until a cow has a greater rating than yours, so if you find the location (j) of the first number greater than your cow's rating, the number of battles you'll win with cows ahead of yours is $j - i - 1$, where i is your cow's location.

With a segment tree, we can do all three of the aforementioned operations in $\mathcal{O}(\log n)$. The first and second operations are elementary, but the third might not be obvious. All we need to do is walk down the segment tree, a form of binary searching. This is a classic technique. If you're unfamiliar with it, consult the editorial to this CSES problem (or try solving it yourself!).

This will solve the problem in $\mathcal{O}(n \log n)$ time. Since $n \leq 10^5$, this fits in the time limit of 1 second.