Problem Set 2 - Data Structures & Paradigms

Problem D

The Problem Set was a magnificently satisfying question implementation wise. All it took was an aggregation of all of the range tree code, and then an additional new union find query in order to solve this question.

When reading this question, maybe I was just post lecture, or maybe it was just that clearly range trees (definitely the latter), I pretty much immediately knew it was range trees. Looking through the queries that they were asking for, it was pretty clear that all of them would be trivial and copied from the lectures other than the last one where we're checking if everything is increasing in a segment.

Initially, I had an incorrect thought of trying to use longest increasing subsequence to check if one of the subsequences . I do believe I ran into some issues when actually looking into the LIS code, as the LIS code only ran across the entire range tree, and I wanted to query subsections of this. The idea was, we find the longest increasing subsequence in a range in the range tree, and if it is equal to the number of problems in that subsequence, then we print 1. The same with the decreasing subsequence. After looking into the lecture code though, I pretty swiftly realised this was wrong.

The actual correct solution I came up with was more straightforward, all we need to do is store (inc_tree[i]) at each node to represent whether the range it covers is purely increasing. If the node corresponds to just a leaf, then it is trivially increasing, but if it's a region, two conditions need to be met:

- Both Subranges Are Increasing: The left and right child nodes of the current node must each be increasing.
- Boundary Check Between Children: The rightmost element of the left child must be less than or equal to the leftmost element of the right child.

The second condition required storing extra information about each range, which is what the left and rightmost values are, stored in left_tree[i] and right_tree[i]. This allowed me to do comparisons between the leftmost of the upper half in a range of responsibility, and the rightmost of the lower half, and check if the former is greater than the latter. If so, then all we need to check is if both halves are purely increasing with our inc query, and we know that the whole range we're querying is purely increasing :).

Done deal!