Kick Start 2018 - Round G

Analysis: Combining Classes

Combining Classes: Analysis

Small dataset

Let us define a function f(x) as the number of students scoring more than or equal to x. We can compute f(x) in O(**N**) time by iterating through all classes.

Therefore, we can use binary search to find the largest x that satisfies f(x) ≥ **K**, which is the score of the **K**-th highest score. This solution runs in O(**Q** × **N** × log(10^9)) time.

Large dataset

We can solve the Large dataset by compressing the scores first. That is, we map the existing values of $\mathbf{L}_i$ and $\mathbf{R}_i$ onto a range [1, D], where D is the number of distinct values in $\mathbf{L}_i$ and $\mathbf{R}_i$. After compressing the scores, we can assume that the students' scores are between 1 and 2**N**.

For each possible score x, let cnt(x) be the number of students with score x. We can compute the values for all cnt(x) in O(**N** log(**N**)) using a range-update point-query data structure such as a segment tree. If we compute the suffix sum (similar to prefix sum, but we cumulate the sum backwards) of cnt(x), we will get the number of students scoring more than or equal to x, which is f(x).

Therefore, just as in our solution for the Small dataset, we can use binary search to find the largest x that satisfies f(x) ≥ **K**. However, note that the answer is not exactly x, since we compressed the scores earlier. The exact details of the implementation are left as an exercise.

This solution runs in O((**N** + **Q**) log **N**) time.