Codeforces Problem 1946F

> **Problem.** Given a permutation of length n and for q queries consisting of integers l, r, determine the number of sorted sequences of indices $t_1, t_2, \ldots, t_k$ for $k \geq 1$ such that each t_i lies between l and r, and a_{t_i} divides $a_{t_{i+1}}$.

Solution (Incorrect)**.** We shall start by precomputing some helpful variables (the time complexities for these shall be verified later).

- Let $\text{starts}[i]$ denote the number of sequences that start with i.
- Let $\text{ends}[i]$ denote the number of sequences that end with i.
- Let $\text{starts\_prefix}[i]$ be the prefix of starts and ends_prefix the prefix of ends. In other words,

$$\text{starts\_prefix}[i] = \sum_{j \leq i} \text{starts}[j],$$
$$\text{ends\_prefix}[i] = \sum_{j \leq i} \text{ends}[j].$$

Once we have these arrays precomputed, we may easily determine the number of sequences that satisfy our conditions. For a given interval of indices l, r, the answer is

$$(\text{starts\_prefix}[r] - \text{starts\_prefix}[l-1]) - (\text{ends\_prefix}[n] - \text{ends\_prefix}[r])1.$$

It is now sufficient to determine a method to compute starts and ends in sufficient time.

We shall start by determining starts. We find this rather simply by iterating backwards (note that the order of this is important) for each a_i and checking the positions of all multiples of a_i to see if they lie in strictly greater indices and then adding path counts accordingly. More formally,

$$\text{starts}[i] = 1 + \sum_{\substack{a_i | a_j, \\ j > i}} \text{starts}[j].$$

One can verify that this runs in sufficient time as

$$\sum_{i=1}^{n} \left\lfloor \frac{n}{i} \right\rfloor = O(n \log n).$$

The computation of the ends array is slightly different, and this asymmetry owes itself to the computational differences in finding divisors as opposed to the simpler multiples. Thus, we must take a different approach. We have that

$$\mathsf{ends}[i] = 1 + \sum_{\substack{a_j | a_i, \\ j<i}} \mathsf{ends}[j].$$

Iterating over each value of i linearly would result in us having to factor each a_i, but luckily we can simply over j, resulting in another $O(n \log n)$ pass over the array.

It actually appears that this solution is incomplete and incorrect, as storing the starting path and ending path prefix counts does not allow one to uniquely determine all paths (the equation given above for the answer in terms of the prefixes is obviously wrong after more thought is given). Another solution path will have to be attempted.

This being said, it is likely that the true solution follows a similar path, especially with the $O(n \log n)$ computation times.

Solution.