1. prefix sum, answer each query in $O(U)$ time. $O((n+q)U)$, or $O(n \cdot U^{\epsilon} + qU)$ (using persistent U^{ϵ}-ary tree for prefix sum).
2. sweep line, for each value store its occurrences in a sorted list, so that we can check whether a value occur in an interval in $O(1)$ time. $O(n + qU)$.
3. bit packing, for each query get a U-bit vector representing the numbers in the interval.

$$O(\min\{n\alpha(n) \cdot \frac{U}{w} + q \cdot \frac{U}{w} \log w, n \cdot \frac{U}{w} + q \cdot (\frac{U}{w} \log w + \log^{(c)} n)\}).$$

4. prefix sum (mod 2) + bit packing, repeat $O(\log n)$ times to get w.h.p. correctness.

$$O((n+q) \cdot \frac{U}{w} \cdot \log n + q \cdot \frac{U}{w} \log w).$$

We can also divide into blocks with length t. $O((\frac{n}{t} + q) \cdot \frac{U}{w} \cdot \log n + q \cdot \frac{U}{w} \log w)$. (improvable using multiple levels)

Minimum Absolute Difference Queries

Submission Detail

60 / 60 test cases passed.

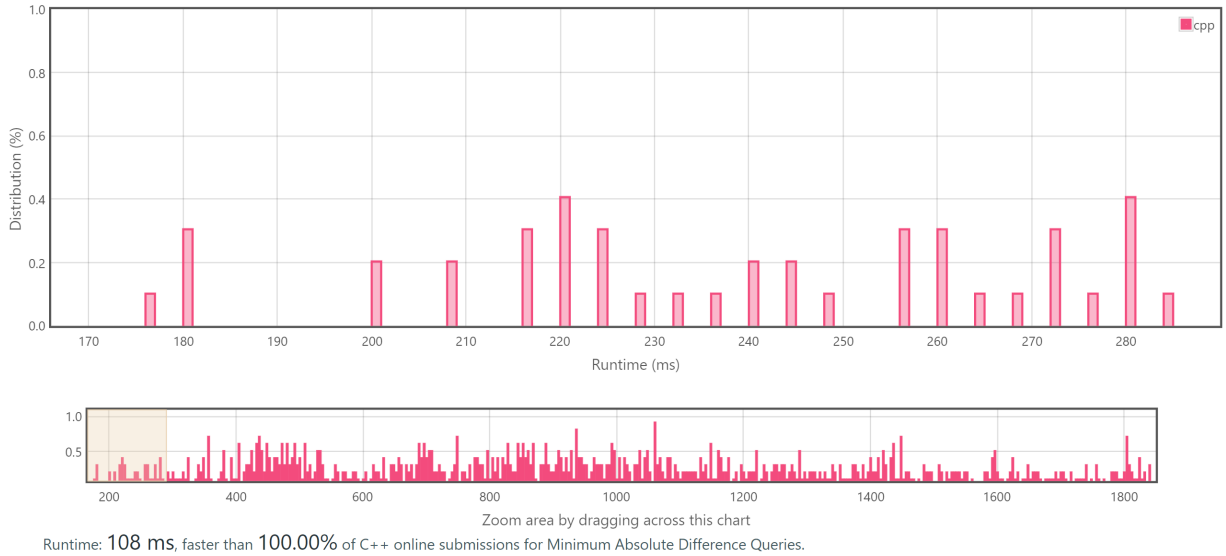
Status: Accepted

Runtime: **108 ms**

Memory Usage: **85.3 MB**

Submitted: **0 minutes ago**

Accepted Solutions Runtime Distribution



Runtime: 108 ms, faster than 100.00% of C++ online submissions for Minimum Absolute Difference Queries.

Memory Usage: 85.3 MB, less than 97.66% of C++ online submissions for Minimum Absolute Difference Queries.

Remark. algorithm for arbitrarily large U: $O(n \log n \log U)$, https://codeforces.com/contest/765/problem/F.

References