Kick Start 2019 - Round E

Analysis: Street Checkers

Test set 1 (Visible)

Let's rephrase the problem into counting the number of values X within the range [**L**, **R**] that satisfies |(# of odd divisors) - (# of even divisors)| ≤ 2.

To solve this problem under the constraint that $\mathbf{R} < 10^6$. We can simply preprocesses each X between 1 and 10^6 whether X is interesting or not. To find whether X is interesting, we can apply any $O(\sqrt{X})$ time algorithm finding out all divisors of X. After storing the result for each X, we build a prefix sum array **F** storing for counts. Thus, each query can be answered by computing **F**[**R**] - **F**[**L**-1] in constant time. The total time complexity would then be $O(\mathbf{R_{max}}\sqrt{\mathbf{R_{max}}} + \mathbf{T})$, which suffices to pass the first test set.

Test set 2 (Hidden)

We need a slightly sophisticated observation here. The intuition comes from observing that any divisor to an odd integer is still odd. For any integer X we can extract all power of 2 factors and rewrite as $X=\mathbf{A}*2^X$, where **A** is an odd integer and X is a non-negative integer.

Now, we can partition all divisors to X into sets of divisors leading with each odd divisors $\mathbf{d_1}$, $\mathbf{d_2}$, ..., $\mathbf{d_k}$ of **A**:

$$\{\mathbf{d_1}, \mathbf{d_1}*2, \mathbf{d_1}*2^2, ..., \mathbf{d_1}*2^X\},$$
$$\{\mathbf{d_2}, \mathbf{d_2}*2, \mathbf{d_2}*2^2, ..., \mathbf{d_2}*2^X\},$$
$$...$$
$$\{\mathbf{d_k}, \mathbf{d_k}*2, \mathbf{d_k}*2^2, ..., \mathbf{d_k}*2^X\}.$$

By looking at the above diagram we can infer that X has X odd divisors and X*X even divisors. The criteria to a number being interesting is now equivalent to |X*(X-1)| ≤ 2.

There are only a few cases of (X, X) pairs that satisfies the above expression:

- Case 1: X=0, X=1 or 2.
- Case 2: X=1, X can be any value.
- Case 3: X=2, X=1 or 2.
- Case 4: X=3, X=1.

So, what we really need to do here is to count the number of X's in the range [**L**, **R**] satisfying each case.

Case 1 implies that X=1 or a odd prime. One can count the number of primes within range [**L**, **R**] using Sieve of Eratosthenes. Case 2 implies that **A** can be any odd integer, so in this case X is in the form of (4*X+2) and hence can be counted in O(1) time. Case 3 implies that **A**=1 or an odd prime. One can count the number of primes within range [**L**/4, **R**/4]. Case 4 implies that X=8. Count it whenever 8 belongs to the range.

From the above case analysis, one can see that the running time is dominated by Sieve of Eratosthenes. Fortunately, we only need to use sieve method on arrays of size O(**R**-**L**+1) to count the number of odd primes within ranges [**L**, **R**] and [**L**/4, **R**/4]. If we apply sieving using numbers 2, 3, 4, ..., $\sqrt{\mathbf{R}}$, the algorithm runs in $O(\mathbf{T}*(\mathbf{R} - \mathbf{L} + 1)*\log(\sqrt{\mathbf{R}}))$ time, which suffices to pass this test set.

For a more efficient algorithm, one can sieve using prime numbers no more than $\sqrt{\mathbf{R}}$. These prime numbers again can be obtained by a sieving algorithm. At the end you will get an algorithm that runs in $O(\sqrt{\mathbf{R}} \log \log \sqrt{\mathbf{R}}) + O((\mathbf{R} - \mathbf{L} + 1) \log \log \sqrt{\mathbf{R}})$ time per test case.