Editorial: Effects of anti pimples - 1876B

Avighna

June 2024

The original problem statement is abstruse. Here's a simpler, rewritten version:

> Chaneka has an array a with n elements. She'll color a nonempty subset of these elements, and then color all uncolored integer multiples of each of the colored indices. The score of such a coloring is the maximum value of the colored elements. Find the sum of scores across all $2^n - 1$ colorings, mod $998,244,353$.

I'll be using a string of zeroes and ones to denote a subset. A zero being in the i^{th} position indicates that the i^{th} element is absent in the subset, and a one indicates that it's present. To denote a group of subsets/subsets conforming to a particular form, I'll use underscores along with the aforementioned zeroes and ones. An underscore can be replaced by either a zero or a one. For example, 1_ _1 denotes $[1001, 1011, 1101, 1111]$.

Let's consider an example: $a = [19, 14, 5, 19, 9]$. 19, the largest element of the array, is present at indices 1 and 4. All divisors of these indices (1 and 4) are $1, 2$, and 4. This means that any subset of the form ? ? _ ? _, where *at least* one ? is a 1, will have a score of 19.

Since we have three ?s, we can form $2^3 - 1$ different arrangements with at least one ? being a 1 (2^3 total subsets, -1 to exclude the empty subset). The underscores can be replaced by a one or a zero, as all other elements in the array are less than 19. In total, we have $(2^3 - 1)(2^2)$ subsets that have a score of 19.

Counting how many subsets have a score of x for any $x < 19$ necessitates the consideration of only subsets with the first, second, and fourth elements zeroed out: i.e., 00_0_.

This hints at a strategy: starting from the largest number (with the current number being a_i), find out how many (previously unused) divisors its indices have: let this be d. If we then let the number of free spots we have remaining be f, the number of subsets that have a score of a_i is $2^f(2^d - 1)$.

An important observation to make at this point is that each divisor is only being assigned to a single number (we use it only once). Thus instead of factoring the indices, we can iterate over all possible divisors and assign the current divisor to the largest element at an index that is a multiple of it.

```
vector<ll> div_cnt((ll)(1e5) + 1);
for (ll div = 1; div <= n; ++div) {
  ll mx = 0;
  for (ll mult = div; mult <= n; mult += div) {
    mx = max(mx, a[mult]);
  }
  div_cnt[mx]++;
}
```

Then we simulate the above process, counting how many subsets have a score of x (s), with x descending, and adding sx to the answer.

```
ll used = 0;
```

```
ll ans = 0;
for (ll x = (ll)(1e5); x >= 1; --x) {
  ans = (ans + ((((binexp(2, div_cnt[x]) - 1) * (binexp(2, n - div_cnt[x] - used))) % mod) * x) % mod
  used += div_cnt[x];
}
```

And that's it!