Code Jam 2022 - Round 1A

Analysis: Equal Sum

In this problem we first pick half of the input, then the judge picks the other half, and then we need to solve an NP-complete problem on that input. This means we have to be really strategic on how we pick that input.

We only get to provide half of the numbers. So, it might be best to first see how good of a split we can do with using just the judge provided numbers and only afterwards come up with the numbers we provide.

A common heuristic to solve problems in which we do not know all of the input yet is to optimize locally. In this case, sort out the numbers into two sets, trying for their sums to be as close as possible. That means, we go through the input integers and assign them to the set that has the smallest current sum. After we process all of the judge's integers, this guarantees that the difference between the sets is bounded by the size of a single integer (10^9), as opposed to being bounded by the total sum of the integers we have seen (10^{11}).

We need our integers to always be able to make up for that difference, so we need to guarantee at least some of the integers we pick are big enough. Moreover, we need to be able to make up for the difference exactly, so we want to pick integers that get progressively more precise. This smells like binary. However, when writing in binary we get to sum over a subset of the powers of 2. In this case, the integers that do not end up in one subset go to the other one, so binary can just write any possible difference. That being said, powers of 2 still work, as long as we do not write in binary.

After we finish sorting out the judge's input, we know that the difference between the two subsets is at most 10^9. Therefore, if we decide where to put the largest power of 2 (2^{29}) using the same algorithm, that guarantees that the difference is now at most 2^{29}. As long as the next number is always at least half of the previous one, we can maintain the difference bounded by the last processed number. If we process the powers of 2 last, and in decreasing order, the difference between the two subsets at the end is no larger than $2^0 = 1$. Since the limits guarantee that the total sum is even, the difference is 0.

The paragraph above shows that any set of integers up to X with even sum that contains every power of 2 no greater than X can be partitioned into two subsets of equal sum. Moreover, the partition can be found efficiently. There are 30 powers of 2 between 1 and 10^9, which means we can choose the other 70 integers we are entitled in any way.