Code Jam 2013 - Round 1B

Analysis: Osmos

We can simplify the problem by making the following observation:

> There is an optimal solution where Armin chooses to absorb motes in order from smallest to largest (skipping removed motes).

If Armin's solution absorbs mote X before mote Y, and mote X is larger than mote Y, then he could change his solution to absorb Y right before X, without needing to perform any extra "add" or "remove" operations. So if Armin has an optimal solution, we can always change it into an optimal solution that absorbs motes in order of size.

Now we can limit our search to solutions that absorb motes in order of size. We could use a dynamic programming algorithm where the state is the number of motes considered, and Armin's mote's current size or the number of operations performed, but there is a simpler algorithm based on the following observation:

> In an optimal solution, if Armin removes a mote, he also removes all motes of equal or greater size.

To see this, consider a solution where there exist motes X and Y where X is smaller than or equal in size to Y, X is removed, and Y is absorbed. X could instead be absorbed immediately before Y is absorbed, which would save an operation by not removing X. So the solution cannot be optimal.

So to find an optimal solution, we only need to consider **N**+1 cases -- those where we try to absorb 0, 1, ... **N** of the original motes and remove the remainder.

To find how many operations are needed for each of these cases, we simulate Armin trying to absorb each mote in turn. If Armin's mote is not yet large enough to absorb the next mote, we add motes of size one less than Armin's mote's current size and absorb them, until Armin's mote is large enough.

This solution takes $O(\mathbf{N}^2)$ time to run as written, which is fast enough given the input size limits. There is a small adjustment to it that will make it linear, though. Can you see it?

One final case to handle is when Armin's mote is of size 1, and so is unable to absorb any motes at all. We were generous and added this as a case in the sample input!

This mechanics for this problem were inspired by Osmos by Hemisphere Games.