Problem Set 3 - Dynamic Programming

Problem A

Other than Mathematics - which I still don't get, Dynamic Programming was the hardest topic for me to get my head across. I'm MUCH better at it now, and honestly, it's probably the topic where I feel like I've learned the most out of all of them - but I was at the point where I needed a friend to explain the simplest practice problem on formatif to me to get.

Initially, when reading question A of problem set 3, I was completely stumped. In hindsight now, the question is quite easy, I would . I think my confusion was trying to wrap my head around whether or not the final state in the DP had to equal the final answer or not, and if so - figuring out how to reconcile the two options of choosing from either line last into one state was too much for my brain. I ended up learning here that the final state in the DP doesn't always have to be the final answer, we've got a computer, so we can calculate many different final states, and then get a final answer out of them. And so that's what I did - I calculated the max height if line 1 at position n was chosen last, and also at line 0 at the same position, and got the max of the two of those for the final answer.

In hindsight now with a better understanding of DP, I could have combined these into one final state of d[n], where d[n] is just the max height at position n, but this was a good learning regardless.

The transition between the states was fairly reasonable and easy to pick up (at least now, it probably took a little longer before) - since it was alternating, the only two maximal options were either choosing from the opposite row, 1 before, or two before. Choosing three positions before on the opposite row would net a non-maximal result as we're missing out on selections in between, and choosing one prior on the same row was nor permitted.

I ended up implementing this question top down and recursively because it felt a lot more intuitive to me. You take the top element, break it down into sub elements, solve for those, until you reach the base case.

All done :)