# Problem 1

I approached this question from a DP perspective by first thinking carefully about the state.

- Since I need to alternate between rows while maximizing the total value collected, the main constraint is that **no two consecutive players can be from the same row**.

Initially, my idea was that for each column `i`, I could either:

- Take the player from the opposite row of the previously chosen one (which is valid),
- Or try to take a player from the same row and "replace" the previous one (which turns out to violate the problem constraints).

With this line of thinking, I implemented a relatively straightforward solution that passed some test cases, but failed on test case 11.

The problem was that my solution was **greedy** rather than truly dynamic. I was using a kind of 1D DP, where I just went through the players and picked what looked best at the moment. This led to missed opportunities, because it ignored the possibility that a worse choice now might lead to a better overall path later: the “what if I had gone the other way?” scenario.

After realising this, I changed my approach to keep track of **both choices at each step** — the value if I had taken the player from the top row and the value if I had taken the one from the bottom. By maintaining both options as separate states and propagating them forward independently, I was able to compare them at the end and pick the maximum. This corrected implementation was accepted.

In hindsight, I could have identified the issue earlier with a more robust and targeted set of test cases that better exposed the limitations of the greedy path.

Ask ChatGPT