Subtask 1 (20 points):

Since there is only one component the second player is going to claim it. It was enough to calculate the size of that component which can be done by counting the number of already claimed boxes and subtracting from the total number.

Subtask 2 (20 points):

The number of positions players could get (which is $2^{\#\text{unjoined sides}}$) was small enough to try and determine the optimal strategy from each possible state using dynamic programming or memoization.

Subtask 3 (20 points):

The general strategy should be about claiming the boxes whenever possible but it is important to understand that sometimes when you have the opportunity to take a whole chain of length n it is better to claim n-2 boxes and leave the last 2 to the other player by making a move that doesn't complete any of those 2 boxes. Using this you are forcing the opponent to make the bad move later that will give the current player an even bigger component back. With this in mind, one could find the components (whose amount is at most 2) and find the answer using simple casework.

Subtask 4 (20 points):

Due to the constraints each component will be either a chain or a loop. The state of the game can be described as the lengths of the remaining chains and loops. There won't be many components in this subtask and one could solve it by solving each state, similar to subtask 2, this time the number of states being $O(2^{\#\text{of components}})$.

Subtask 5 (20 points):

Notice that if one player has to give up the component of a certain kind (chain or loop), it is better to give up the smallest one. Using this the number of states that optimal solutions can reach can be reduced significantly to some polynomial amount and this time dynamic programming/memoization approach can be used to achieve the full score.