Task 1

In this task, it's just a merge sort, where the sorting is done using divide and conquer method.

Task 2

In this task, I just have divide every single element and findout max pair pairs.

Task 3

In this task, I used the merge sort method, but I modified it as I have to keep counting how many pairs are there where the height of the first things is greater. The counting is mainly in the merge function.

Task 4

Here I just use merge sort function and dividing in every single element. While there are single elements of both sides, then I wrote some conditions to build the logic. As there are no loop and it's a merge sort function, so, the time complexity is $O(n \lg n)$.

Task 5

Here, it's just a quick sort, where the sorting works with pivot.

Task 6

Here, I used the partition function of the quick sort, so that I can find the pivot which will be in the right place in the array. And if that pivot is the target number then just return it.