Coding 1.

Time complexity - a measure of how fast an algorithm runs.
Space complexity - a measure of how much auxiliary memory an algorithm takes up.

time& space complexity = time to run a function and memory required to run the function.

Big O notation- gives you an idea of how your function scales as the.input to the functions gets larger. i.e how time space complexity of an algorithm changes depending on the input.

n is the number of elements in the given array or the size of the given array.

For this function..

I could do it both ways but i will chose option 1 although due to time limit, i have been unable to run this code, so this is more of a pseudocode.

Option 1 has a big O Notation of 0(1) as once it runs through the first index and the last index, would return false immediately if condition is not met. I have chosen this approach as it is fast and will not need to run through the argument array/input completely. It will return false on the first search if the condition is not met

Option 2 is O(N) as it would need to iterate through the argument and reverse and split and then join and the time complexity and space complexity will depend onn the size of the string hence the o(n) where n is the size of the input.

Coding 2;
For this part, the big o(notation) will be O(N) as well as the input will need to be sorted and the time and space complexity will depend on the size of the input and because I am looping through the argument parameters hence O(N). I have chosen this approach as its the only way i can think of solving this issue however there may be a possibility of O(2log n) using a different search and sort method such as bubble sort or selection sort.