FIRST & LAST OCCURRENCE

☆ In this problem, we are given an array of sorted elements and we are supposed to return the first and last occurrences of our target element as array may not be only unique elements

Approach is very similar to lower bound and upper bound. When we finally find our element we adjust for repeated occurrences using while loops.

Pseudocode :

```
firstAndLastOccurrence(arr, N, a) {
    low = 0
    high = N - 1
    a1, a2 = 0, 0
    while(low <= high) {
        mid = (low + high) / 2
        if(arr[mid] == a) {
            a1, a2 = mid, mid
            while(arr[a1] == arr[mid]) {
                a1 --
            }
            while(arr[a2] == arr[mid]) {
                a2 ++
            }
            return (a1, a2)
        } else if (arr[mid] < a) {
            low = mid + 1
        } else {
            high = mid - 1
        }
    }
    return -1
}
```

This approach relies on linear search after element is found somewhere but can be optimized to be purely binary

PseudoCode:

```
firstOccurrence(arr, N, a) {
    low = 0
    high = N - 1
    ans = -1
    while(low <= high) {
        if (arr[mid] == a) {
            ans = mid
            high = mid - 1
        } else if (arr[mid] < a) {
            low = mid + 1
        } else {
            high = mid - 1
        }
    }
    return ans
}
```