□ Two Sum II – Input Array Is Sorted

Problem Statement

Given a sorted array numbers, find two numbers that add up to target.

Return indices (1-based).

Example

Input:

numbers = [2, 7, 11, 15]

target = 9

Step-by-Step

1. Start:

```
i → 2          j → 15
|               |
[2, 7, 11, 15]
```

Sum = 2 + 15 = 17 → Too big → move j left

2. Next:

```
i → 2         j → 11
|              |
[2, 7, 11, 15]
```

Sum = 2 + 11 = 13 → Too big → move j left

3. Next:

```
i → 2      j → 7
|          |
[2, 7, 11, 15]
```

Sum = 2 + 7 = 9 → □ Match!

Return indices:

[i + 1, j + 1] → [1, 2]

Why Two Pointers?

- Sorted array → move pointers to shrink or grow sum
- O(n) time
- O(1) space

Code (JavaScript):

```
var twoSum = function(numbers, target) {
    let i = 0;
    let j = numbers.length - 1;
    while (i < j) {
        let sum = numbers[i] + numbers[j];
        if (sum < target) {
            i++;
        } else if (sum > target) {
            j--;
        } else {
            return [i + 1, j + 1]; // Return 1-based indices
        }
    }
};
```

Quick Revision Points

□ Sorted Array → Use two pointers

□ If sum < target → move i right

□ If sum > target → move j left

□ Return indices (1-based)