Two pointer method

Find two numbers in a sorted array that add up to a target.

Idea (Two Pointers Method)

Since the array is **already sorted**, we can:

1. Put one pointer at the **start** (`left = 0`)
2. Put another pointer at the **end** (`right = n-1`)
3. Check their sum:
    - If sum = target → found
    - If sum < target → move **left++** (increase sum)
    - If sum > target → move **right--** (decrease sum)

Time Complexity: **O(n)**
Space Complexity: **O(1)**

Example

Array = `[2, 4, 7, 11, 15]`
Target = `9`

Steps:

- 2 + 15 = 17 → too big → move right
- 2 + 11 = 13 → too big → move right
- 2 + 7 = **9**

Answer → **2 and 7**

Java Code (Easy)

```
public class TwoSumSorted {
```

```
    public static void findPair(int[] arr, int target) {
        int left = 0;
        int right = arr.length - 1;

        while (left < right) {
            int sum = arr[left] + arr[right];

            if (sum == target) {
                System.out.println("Pair found: " + arr[left] + " and " +
arr[right]);
                return;
            }
            else if (sum < target) {
                left++;
            }
            else {
                right--;
            }
        }
        System.out.println("No pair found");
    }

    public static void main(String[] args) {
        int[] arr = {2, 4, 7, 11, 15};
        int target = 9;
        findPair(arr, target);
    }
}
```

Interview Tip

If the array **is NOT sorted**, use:
👉 **HashMap approach** (Two Sum classic problem).