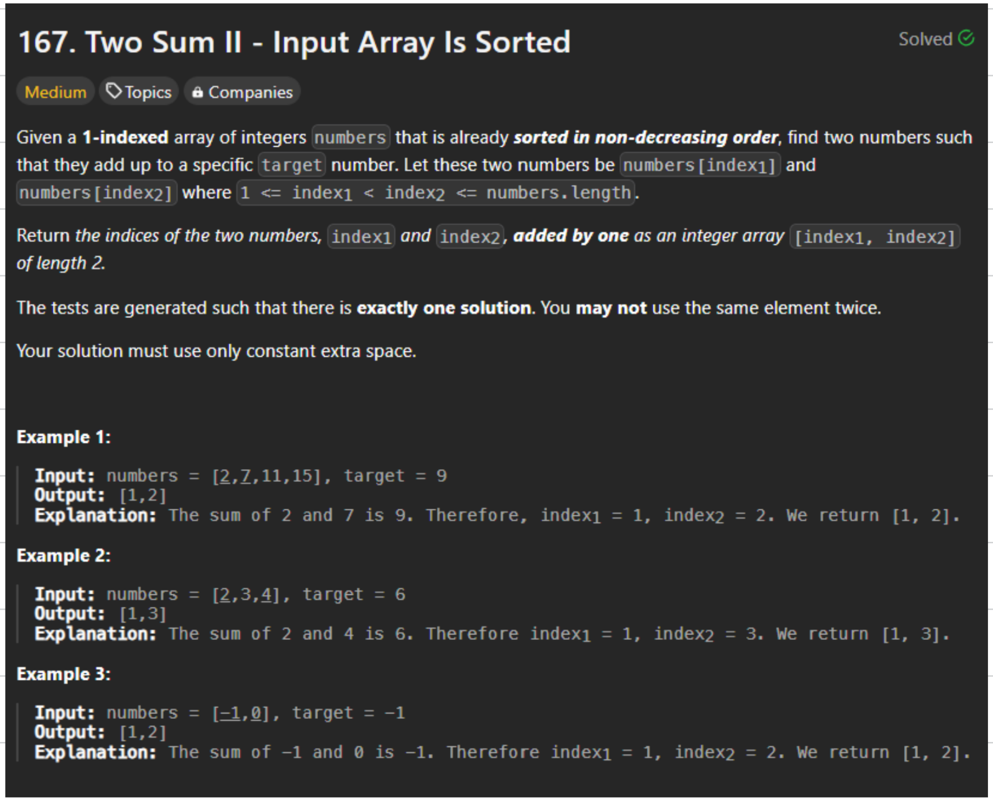
Two Sum

↳ Input array is Sorted



167. Two Sum II - Input Array Is Sorted

Solved ✓

Medium · Topics · Companies

Given a **1-indexed** array of integers `numbers` that is already ***sorted in non-decreasing order***, find two numbers such that they add up to a specific `target` number. Let these two numbers be `numbers[index1]` and `numbers[index2]` where `1 <= index1 < index2 <= numbers.length`.

Return *the indices of the two numbers,* `index1` *and* `index2`*, **added by one** as an integer array* `[index1, index2]` *of length 2.*

The tests are generated such that there is **exactly one solution**. You **may not** use the same element twice.

Your solution must use only constant extra space.

Example 1:

```
Input: numbers = [2,7,11,15], target = 9
Output: [1,2]
Explanation: The sum of 2 and 7 is 9. Therefore, index1 = 1, index2 = 2. We return [1, 2].
```

Example 2:

```
Input: numbers = [2,3,4], target = 6
Output: [1,3]
Explanation: The sum of 2 and 4 is 6. Therefore index1 = 1, index2 = 3. We return [1, 3].
```

Example 3:

```
Input: numbers = [-1,0], target = -1
Output: [1,2]
Explanation: The sum of -1 and 0 is -1. Therefore index1 = 1, index2 = 2. We return [1, 2].
```

return any 2 indices, which results the sum of both indices equals to the given target

2-pointers Technique

If you are required to find elements that satisfies certain constraints or conditions

Most commonly Used in:

1. Two Sum
2. Three Sum
3. Container with most Water
4. Sorting n Searching, String Manipulation
5. Palindrome Problems
6. Linked List problems
7. Sliding window Problems

So, as our array is sorted, we can make observations out of it

$arr[] = \{2, 7, 11, 15\}$

if you set pointer $\longrightarrow$ i at 0
j at n-1

$arr[] = \{\underset{\overset{\uparrow}{i}}{2}, 7, 11, \underset{\overset{\uparrow}{j}}{15}\}$

* if you sum up arr[i] & arr[j] $\Longrightarrow$ you will get some x as ans

Let us assume, $\boxed{sum = x}$

Case - I

If $\boxed{sum < target}$ $\longrightarrow$ it means our sum need to be increased to match target value

for that, you have to increment 'i', because array is sorted if you move i forward, the value will definetly increase

Case - II

If $\boxed{\text{Sum} > \text{target}}$ → it means our sum need to be decreased to match target Value

for that, you have to decrement 'j', because array is Sorted if you move j backward, the value will definetly decreases

Case - III

if (Sum == target)

↳ return the index Values of (i, j)

	i	j	Sum		target	sum > target	Sum < target
arr[] = {2,7,11,15}, tar = 18 (↑i at 0, ↑j at 3)	0	3	17	<	18		i++ (move to bigger value)
arr[] = {2,7,11,15}, tar = 18 (↑i at 1, ↑j at 3)	1	3	21	>	18	j-- (Sum need to reduce)	
arr[] = {2,7,11,15}, tar = 18 (↑i at 1, ↑j at 2)	1	2	18	=	18	→ return (1, 2)	