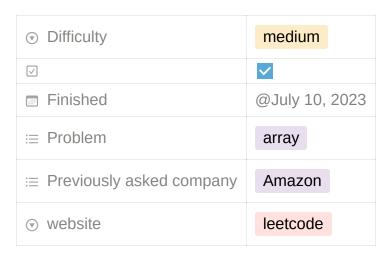
167. Two Sum II - Input Array Is Sorted



Question:

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[index 1] and numbers[index 2] where 1 <= index 1 < index 2 < numbers.length.

Return the indices of the two numbers, index 1 and index 2, added by one as an integer array [index 1, index 2] 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].
```

Optimal solution:

Time complexity: O(n)

Space complexity: O(1)

```
class Solution(object):
    def twoSum(self, numbers, target):
        n = len(numbers)
        l, r = 0, n-1
        res = []
    while l < r:
        currSum = numbers[l] + numbers[r]
        if currSum < target:
            l += 1
        elif currSum > target:
            r -= 1
        else:
            return [l+1, r+1]
```