167. Two Sum II - Input array is sorted

Easy 3097 764 Add to List Share

Given an 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.

Return *the indices of the two numbers (**1-indexed**) as an integer array* `answer` *of size* `2`, *where* `1 <= answer[0] < answer[1] <= numbers.length`.

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

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.
```

Example 2:

```
Input: numbers = [2,3,4], target = 6
Output: [1,3]
```

Example 3:

```
Input: numbers = [-1,0], target = -1
Output: [1,2]
```

Constraints:

- `2 <= numbers.length <= 3 * 10^4`
- `-1000 <= numbers[i] <= 1000`
- `numbers` is sorted in **non-decreasing order**.

```python
class Solution:
    def twoSum(self, numbers: List[int], target: int) ->
List[int]:

        res = []

        for i in range(0,len(numbers)) :
            l,r = i+1, len(numbers)-1

            while l <= r :
                mid = l+ (r-l)//2
                if numbers[mid] == (target-numbers[i]) :
                    res.append(i+1)
                    res.append(mid+1)
                    return res
                elif (target-numbers[i]) < numbers[mid] :
                    r = mid-1
                else :
                    l = mid+1
```

Testcase | Run Code Result | Debugger

Accepted Runtime: 47 ms

Your input
[2,7,11,15]
9

Output [1,2] Diff

Expected [1,2]

Console Use Example Testcases