[Click on the logo to find **Problem Statement**]

Intuition

The provided code implements the two-pointer approach to efficiently find a pair of indices in a sorted list where the elements add up to a given target. The `left` and `right` pointers start at the beginning and end of the list, respectively. The algorithm iteratively compares the sum of elements at these pointers with the target. If the sum is equal to the target, the function returns the indices. If the sum is less than the target, the `left` pointer is moved to the right to consider a larger element. If the sum is greater, the `right` pointer is moved to the left to consider a smaller element. This process continues until the target sum is found or the pointers converge, indicating that no such pair exists.

Approach

1. **Initialize Pointers:** Set two pointers, `left` and `right`, initially pointing to the start and end of the sorted list, respectively.

2. **While Loop:** Enter a while loop as long as the `left` pointer is less than or equal to the `right` pointer.

3. **Check Sum:**

   - Calculate the sum of elements at the `left` and `right` pointers.
   - If the sum equals the target, return the indices [left + 1, right + 1] since the list is 1-indexed.
   - If the sum is less than the target, increment the `left` pointer to consider a larger element.
   - If the sum is greater than the target, decrement the `right` pointer to consider a smaller element.

4. **Exit Condition:** If the while loop completes without finding a pair, it implies that no such pair with the given target sum exists in the list.

Complexity

- Time complexity: O(n)

- Space complexity: O(1)

Code

```
class Solution:
    def twoSum(self, numbers: List[int], target: int) -> List[int]:
        left = 0
        right = len(numbers) - 1
        while left <= right:
            if numbers[left] + numbers[right] == target:
                return [left + 1 , right + 1]
            elif numbers[left] + numbers[right] < target:
                left += 1
            else:
                right -= 1
```

If you want to see more solutions to coding problems, you can visit:

GitHub