**Problem Name: Equilibrium Index of an array**

**Problem Statement:**

#### Given a sequence of integers. Find the equilibrium indices of the sequence. The equilibrium index of a sequence of integers is defined as the index such that the sum of all the elements at lower indices is equal to the sum of elements at higher indices

##### Note:

1. Do not print anything, just return a sequence of equilibrium indices
2. A sequence may contain more than one equilibrium indices.
3. Consider 0 based Indexing.

##### Input format:

The first line of input contains an integer ‘T’ denoting the number of test cases.

The next ‘2\*T’ lines represent the ‘T’ test cases.

The first line of each test case contains an integer ‘n’ denoting the number of elements in

the given sequence.

The second line of each test case contains ‘n’ space-separated integers denoting the elements in the sequence.

##### Output Format

For each test case, return the minimum index present at the equilibrium indices(consider 0 based indexing). If no such indices exist, return -1.

##### Constraints:

1 <= T <= 100

1 <= N <= 10^3

1 <= A[i] <= 10^4

Where ‘T’ is the total number of test cases, ‘N’ denotes the length of the sequence.

Time limit: 1 second

##### Sample Input 1:

2

7

-7 1 5 2 -4 3 0

5

1 2 3 4 5

##### Sample Output 1:

3

-1

##### Explanation of sample input 1 :

Test Case 1:

In the given sequence of indexs, index 3 (consider 0 based indexing) is the equilibrium index, because all the sum of the indexs present at the indices lower than 3 i.e [-7 + 1 + 5 = 1 ] is equal to sum of all the indexs present at indices higher than 3 i.e [ -4 + 3 + 0 = 1]. Hence we return 3 which is the lowest equilibrium index.

Test Case 2:For each index we find the sum of all the values present at indices lower than the index and greater than the index.

For index=0, Sum of elements present at indices lower than 0 is 0, and the sum of elements present at indices higher than 0 is (2+3+4+5 = 14)0, Because the sum doesn’t match, index 0 is not at equilibrium. We can check in a similar way for all the elements and no element satisfies the equilibrium condition, therefore we return -1.

##### Sample Input 2:

2

7

-2 1 9 2 -6 3 0

4

100 -99 2 1

##### Sample Output 2:

2

2

**Approach 1**

**Problem Title: Equilibrium index of an array**

**Hint to the approach:**

Try to find the sum of indices to the lest and right of the equilibrium index.

**Approach:**

* This is a Naivé/Brute Force Approach approach.
* We can use 2 loops one which checks for each index if it is an equilibrium index or not.
* For every iteration of this loop, it will take the sum of all elements to the right of the current index and also the sum of all elements to the left of the current index and compare if the sum is equal or not.
* Each iteration will take O(n) time as we need to iterate and find the sum of elements to the left and right independently.
* This has to be done N times in the worst case, hence the complexity O(n^2).

**Time complexity:**

O(n^2) where ‘n’ is the number of indices in the array.

<>

**Space complexity:**

O(1). We are using constant space

**Approach 2**

**Problem Title:** Equilibrium index of An Array

**Hint to the approach:**

Think of the sliding window method.

**Approach:**

* The idea is to get the total sum of the array first.
* Make 2 variables LeftSum and RightSum.
* Initially LeftSum=0.
* Then Iterate through the array **from left to right(to ensure we find the lowest equilibrium index first)**  and keep updating the leftSum which is initialized as zero.
* In the loop, we can get the right sum by subtracting the elements one by one.
* If for any index, the leftSum is equal to Rightsum, we return that index as the equilibrium index.
* If we cannot find any index which satisfies the above conditions, we return -1.

**Time complexity:**

O(n), ‘n’ is the length of the array.

Iterate the sequence once to calculate the sum of the array and one more time to find the equilibrium indices.

**Space complexity:**

O(1). We are using constant space