Problem statement

> Given an array A of n positive numbers. The task is to find the first Equilibium Point in the array.
> Equilibrium Point in an array is a position such that the sum of elements before it is equal to the sum of elements after it.
> **Note: Retun the index of Equilibrium point. (1-based index)**

Example:

Input:

```
n = 5
A[] = {1,3,5,2,2}
```

Output:

```
3
```

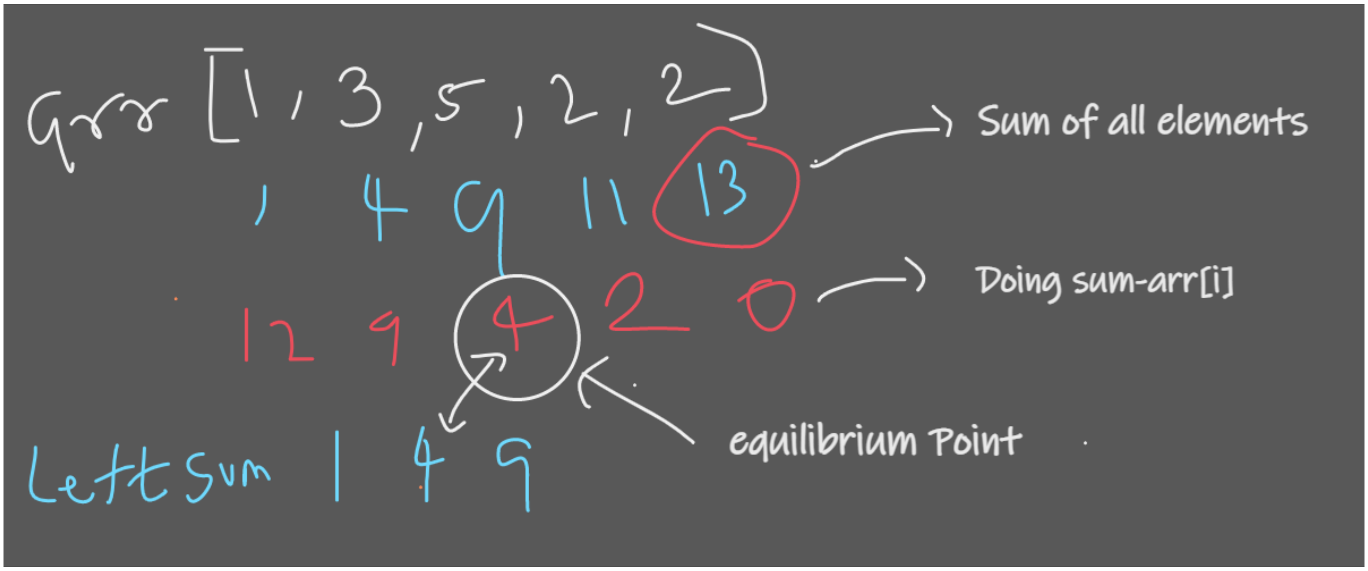
Explanation:

```
For second test case
equilibrium point is at position 3
as elements before it (1+3) =
elements after it (2+2).
```

Approach:

⟶ we will first find the sum of all elements and then traverse the array again but this time we will subtract the element from sum and also we will make one more variable `leftSum` which will also find sum of elements in this traversal and whenever leftSum becomes equal to prefixSum then the current element in equilibrium point

So the process will look like this:



Now why we are subtracting elements from our sum?

⟶ Because if we are at current element and we have subtracted the element from `sum` that means now sum contains the sum of all elements on the right side of it. Also we are using `leftSum` element which have sum of all elements which are on the left side of current element so now we just have to check if they both are same or not and if both are same then we will just return the index otherwise we will add the current element into `leftSum` and go ahead.

So final code will look like this:

```
int equilibriumPoint(long long a[], int n) {

        if(n==1){
            return 1;
        }
        long long prefixsum = 0;
        for(int i=0;i<n;i++){
            prefixsum += a[i];
        }
        long long leftSum = 0;
        for(int i=0;i<n;i++){
            prefixsum -= a[i];
            if(leftSum == prefixsum){
                return i+1;
            }
            leftSum += a[i];
        }
        return -1;
    }
```