Problem 1991: Find the Middle Index in Array

Problem Information

Difficulty: Easy
Acceptance Rate: 68.64%
Paid Only: No
Tags: Array, Prefix Sum

Problem Description

Given a **0-indexed** integer array `nums`, find the **leftmost** `middleIndex` (i.e., the smallest amongst all the possible ones).

A `middleIndex` is an index where `nums[0] + nums[1] + ... + nums[middleIndex-1] == nums[middleIndex+1] + nums[middleIndex+2] + ... + nums[nums.length-1]`.

If `middleIndex == 0`, the left side sum is considered to be `0`. Similarly, if `middleIndex == nums.length - 1`, the right side sum is considered to be `0`.

Return _the**leftmost** _`middleIndex` _that satisfies the condition, or_`-1` _if there is no such index_.

Example 1:

Input: nums = [2,3,-1,_8_ ,4] **Output:** 3 **Explanation:** The sum of the numbers before index 3 is: 2 + 3 + -1 = 4 The sum of the numbers after index 3 is: 4 = 4

Example 2:

Input: nums = [1,-1,_4_] **Output:** 2 **Explanation:** The sum of the numbers before index 2 is: 1 + -1 = 0 The sum of the numbers after index 2 is: 0

Example 3:

Input: nums = [2,5] **Output:** -1 **Explanation:** There is no valid middleIndex.

Constraints:

* `1 <= nums.length <= 100` * `-1000 <= nums[i] <= 1000`

Note: This question is the same as 724: <https://leetcode.com/problems/find-pivot-index/>

Code Snippets

C++:

```
class Solution {
public:
int findMiddleIndex(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int findMiddleIndex(int[] nums) {

}
}
```

Python3:

```
class Solution:
def findMiddleIndex(self, nums: List[int]) -> int:
```