**Approach #1: Prefix Sum [Accepted]**

**Intuition and Algorithm**

We need to quickly compute the sum of values to the left and the right of every index.

Let's say we knew S as the sum of the numbers, and we are at index i. If we knew the sum of numbers leftsum that are to the left of index i, then the other sum to the right of the index would just be S - nums[i] - leftsum.

class Solution {

    public int pivotIndex(int[] nums) {

        int sum = 0, leftsum = 0;

        for (int x: nums) sum += x;

        for (int i = 0; i < nums.length; ++i) {

            if (leftsum == sum - leftsum - nums[i]) return i;

            leftsum += nums[i];

        }

        return -1;

    }

}

As such, we only need to know about leftsum to check whether an index is a pivot index in constant time. Let's do that: as we iterate through candidate indexes i, we will maintain the correct value of leftsum.

**Complexity Analysis**

* Time Complexity: O(N)O(N)*O*(*N*), where NN*N* is the length of nums.
* Space Complexity: O(1)O(1)*O*(1), the space used by leftsum and S.