# 724. Find Pivot Index

| | |
|---|---|
| Tags | |
| Property | @August 15, 2022 |

## Question

原文：

Given an array of integers `nums`, calculate the **pivot index** of this array.

The **pivot index** is the index where the sum of all the numbers **strictly** to the left of the index is equal to the sum of all the numbers **strictly** to the index's right.

If the index is on the left edge of the array, then the left sum is `0` because there are no elements to the left. This also applies to the right edge of the array.

Return *the **leftmost pivot index***. If no such index exists, return -1.

我的理解：看不懂

翻譯：

给出一个整数数组nums，计算这个数组中樞點。

支点指数是指严格意义上在该指数左边的所有数字之和等于严格意义上在该指数右边的所有数字之和的那个指数。

如果索引在数组的左边，那么左边的和就是0，因为左边没有元素。这也适用于数组的右边缘。

返回最左边的支点索引。如果不存在这样的索引(該陣列找不到中樞點)，则返回-1。

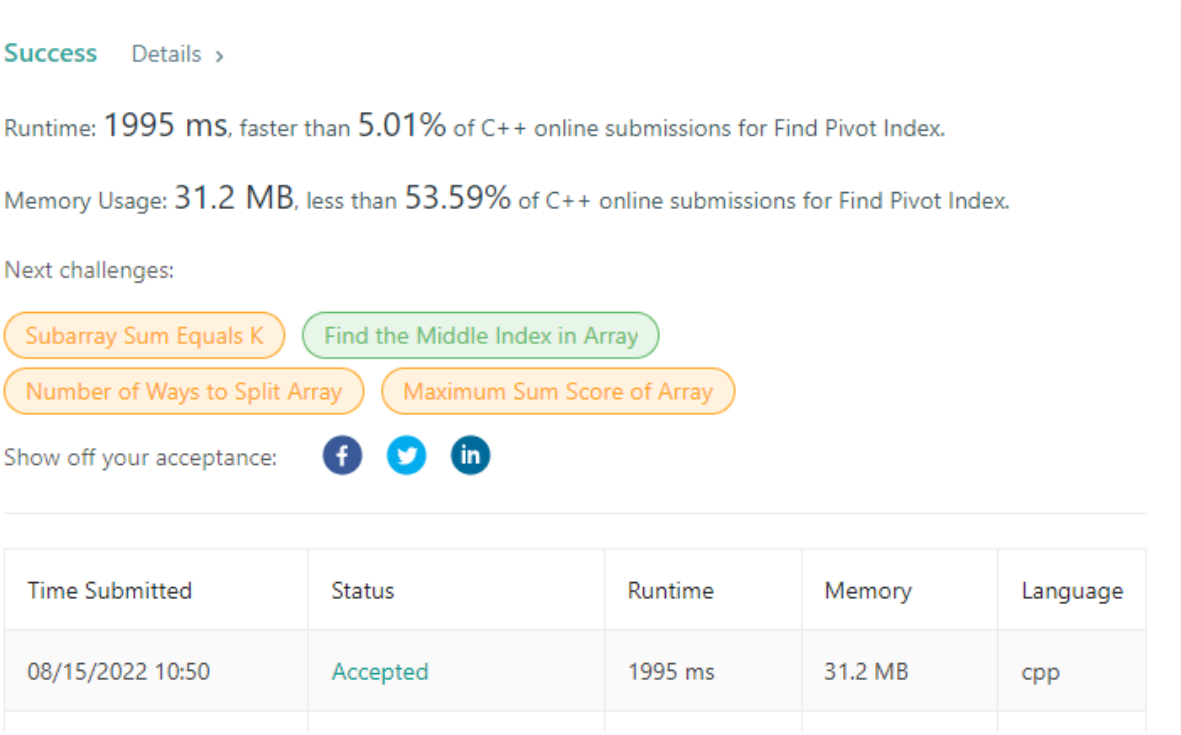
自評翻譯正確性：0

- Word Memory：
  - pivot index 支點索引、中樞點，該位置左右的所有數加起來相同
  - strictly 嚴格
  - element 元素

# Code

```
class Solution {
public:
    int pivotIndex(vector<int>& nums) {
        int Lsum,Rsum,i,L,R;
        for(i=0;i<nums.size();i++){
            Lsum=0;
            Rsum=0;
            for(L=0;L<i;L++){
                Lsum=Lsum+nums[L];
            }
            for(R=nums.size()-1;R>i;R--){
                Rsum=Rsum+nums[R];
            }
            if(Lsum==Rsum){
                return i;
            }
        }
        return -1;
    }
};
```

> 思路：依序跑陣列，跑到哪個位置就把該位址左右所有數相加，總和相等 return 該位址，如依序跑完都找不到左右相等的位址，則 return -1



## 優良code參考

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

        int partial_sum = 0;
        int total = accumulate(nums.begin(),nums.end(),0);

        for(int i=0;i<nums.size();i++)
        {
            int l = partial_sum;
            int r = total-l-nums[i];
            if(l==r)return i;
            partial_sum+=nums[i];
        }
        return -1;
    }
};
```

> 思路：先將陣列總和透過 accumulate 計算出來，依序跑陣列，累計目前位址左邊的所有數總和，右總和則等於 全-左總和-目前位址數值，在比較左右總和是否相同，相同 return 該位址，如依序跑完都找不到左右相等的位址，則 return -1