517. Super Washing Machines

Add to List

Description SubmissionsSolutions

- Total Accepted: **1951**
- Total Submissions: **5483**
- Difficulty: **Hard**
- Contributors: **fallcreek**

You have **n** super washing machines on a line. Initially, each washing machine has some dresses or is empty.

For each **move**, you could choose **any m** (1 ≤ m ≤ n) washing machines, and pass **one dress** of each washing machine to one of its adjacent washing machines **at the same time** .

Given an integer array representing the number of dresses in each washing machine from left to right on the line, you should find the**minimum number of moves** to make all the washing machines have the same number of dresses. If it is not possible to do it, return -1.

Example1

```
Input: [1,0,5]


Output: 3


Explanation:
1st move:    1     0 <-- 5    =>    1     1     4
2nd move:    1 <-- 1 <-- 4    =>    2     1     3
3rd move:    2     1 <-- 3    =>    2     2     2
```

Example2

```
Input: [0,3,0]
```

```
Output: 2


Explanation:
1st move:    0 <-- 3     0    =>    1     2     0
2nd move:    1     2 --> 0    =>    1     1     1
```

Example3

```
Input: [0,2,0]


Output: -1


Explanation:
It's impossible to make all the three washing machines have the same number
of dresses.
```

Note:

1. The range of n is [1, 10000].
2. The range of dresses number in a super washing machine is [0, 1e5].

Subscribe to see which companies asked this question.

Let me use an example to briefly explain this. For example, your machines[] is [0,0,11,5]. So your total is 16 and the target value for each machine is 4. Convert the machines array to a kind of gain/lose array, we get: [-4,-4,7,1]. Now what we want to do is go from the first one and try to make all of them 0.
To make the 1st machines 0, you need to give all its "load" to the 2nd machines.
So we get: [0,-8,7,1]
then: [0,0,-1,1]
lastly: [0,0,0,0], done.
You don't have to worry about the details about how these machines give load to each other. In this process, the least steps we need to eventually finish this process is determined by the peak of abs(cnt) and the max of "gain/lose" array. In this case, the peak of abs(cnt) is 8 and the max of gain/lose array is 7. So the result is 8.

Some other example:
machines: [0,3,0]; gain/lose array: [-1,2,-1]; max = 2, cnt = 0, -1, 1, 0, its abs peak is 1. So result is 2.
machines: [1,0,5]; gain/lose array: [-1,-2,3]; max = 3, cnt = 0, -1, -3, 0, its abs peak is 3. So result is 3.

```
class Solution {
public:
    int findMinMoves(vector<int>& machines) {
        int sz = machines.size();
        int total = accumulate(machines.begin(),machines.end(),0);
        int avg = total/sz;
        if(total%sz!=0) return -1;
        int cnt = 0;int res = 0;
        for(auto load:machines)
        {
            cnt += load - avg;
            res = max(max(res,abs(cnt)),load-avg);
        }
        return res;
    }
};
```