5954. Watering Plants II

My Submissions (/contest/weekly-contest-271/problems/watering-plants-ii/submissions/) Back to Contest (/contest/weekly-contest-271/)

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Medium

Alice and Bob want to water `n` plants in their garden. The plants are arranged in a row and are labeled from `0` to `n - 1` from left to right where the `i`th plant is located at `x = i`.

Each plant needs a specific amount of water. Alice and Bob have a watering can each, **initially full**. They water the plants in the following way:

- Alice waters the plants in order from **left to right**, starting from the `0`th plant. Bob waters the plants in order from **right to left**, starting from the `(n - 1)`th plant. They begin watering the plants **simultaneously**.
- If one does not have enough water to **completely** water the current plant, he/she refills the watering can instantaneously.
- It takes the same amount of time to water each plant regardless of how much water it needs.
- One **cannot** refill the watering can early.
- Each plant can be watered either by Alice or by Bob.
- In case both Alice and Bob reach the same plant, the one with **more** water currently in his/her watering can should water this plant. If they have the same amount of water, then Alice should water this plant.

Given a **0-indexed** integer array `plants` of `n` integers, where `plants[i]` is the amount of water the `i`th plant needs, and two integers `capacityA` and `capacityB` representing the capacities of Alice's and Bob's watering cans respectively, return *the **number of times** they have to refill to water all the plants*.

Example 1:

```
Input: plants = [2,2,3,3], capacityA = 5, capacityB = 5
Output: 1
Explanation:
- Initially, Alice and Bob have 5 units of water each in their watering cans.
- Alice waters plant 0, Bob waters plant 3.
- Alice and Bob now have 3 units and 2 units of water respectively.
- Alice has enough water for plant 1, so she waters it. Bob does not have enough water for plant 2, so he refills his c
So, the total number of times they have to refill to water all the plants is 0 + 0 + 1 + 0 = 1.
```

Example 2:

```
Input: plants = [2,2,3,3], capacityA = 3, capacityB = 4
Output: 2
Explanation:
- Initially, Alice and Bob have 3 units and 4 units of water in their watering cans respectively.
- Alice waters plant 0, Bob waters plant 3.
- Alice and Bob now have 1 unit of water each, and need to water plants 1 and 2 respectively.
- Since neither of them have enough water for their current plants, they refill their cans and then water the plants.
So, the total number of times they have to refill to water all the plants is 0 + 1 + 1 + 0 = 2.
```

Example 3:

```
Input: plants = [5], capacityA = 10, capacityB = 8
Output: 0
Explanation:
- There is only one plant.
- Alice's watering can has 10 units of water, whereas Bob's can has 8 units. Since Alice has more water in her can, she
So, the total number of times they have to refill is 0.
```

Example 4:

```
Input: plants = [1,2,4,4,5], capacityA = 6, capacityB = 5
Output: 2
Explanation:
- Initially, Alice and Bob have 6 units and 5 units of water in their watering cans respectively.
- Alice waters plant 0, Bob waters plant 4.
- Alice and Bob now have 5 units and 0 units of water respectively.
- Alice has enough water for plant 1, so she waters it. Bob does not have enough water for plant 3, so he refills his c
- Alice and Bob now have 3 units and 1 unit of water respectively.
- Since Alice has more water, she waters plant 2. However, she does not have enough water to completely water this plan
So, the total number of times they have to refill to water all the plants is 0 + 0 + 1 + 1 + 0 = 2.
```

Example 5:

```
Input: plants = [2,2,5,2,2], capacityA = 5, capacityB = 5
Output: 1
Explanation:
Both Alice and Bob will reach the middle plant with the same amount of water, so Alice will water it.
She will have 1 unit of water when she reaches it, so she will refill her can.
This is the only refill needed.
```

Constraints:

- `n == plants.length`
- `1 <= n <=` 10^5
- `1 <= plants[i] <=` 10^6
- `max(plants[i]) <= capacityA, capacityB <=` 10^9

JavaScript

```javascript
const minimumRefill = (plants, capacityA, capacityB) => {
    let res = 0, pn = plants.length, middle = pn >> 1, final = 0;
    let a = plants.slice(0, middle), b = plants.slice(middle);
    if (pn & 1) final = b.shift();
    let [cntA, curA] = wateringPlantsRefillCount(a, capacityA);
    let [cntB, curB] = wateringPlantsRefillCountReverse(b, capacityB);
    res += cntA;
    res += cntB;
    // let cur = Math.max(curA, curB);
    // if (cur < final) res++;
    if (curA < curB) {
        if (curB < final) res++;
    } else {
        if (curA < final) res++;
    }
    return res;
};

const wateringPlantsRefillCount = (a, capacity) => {
    let n = a.length, refill = 0, cur = capacity;
    for (let i = 0; i < n; i++) {
        if (cur - a[i] >= 0) {
            cur -= a[i];
        } else {
            cur = capacity;
            refill++;
            cur -= a[i];
        }
    }
    return [refill, cur];
};

const wateringPlantsRefillCountReverse = (a, capacity) => {
    let n = a.length, refill = 0, cur = capacity;
    for (let i = n - 1; ~i; i--) {
        if (cur - a[i] >= 0) {
            cur -= a[i];
        } else {
            cur = capacity;
            refill++;
            cur -= a[i];
        }
    }
    return [refill, cur];
};
```

□ **Custom Testcase** Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/600553916/) More Details (/submissions/detail/600553916/)

Share your acceptance!

◀1

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)

United States (/region)