1228. Missing Number In Arithmetic Progression

Easy 217 26 Add to List Share

In some array `arr`, the values were in arithmetic progression: the values `arr[i + 1] - arr[i]` are all equal for every `0 <= i < arr.length - 1`.

A value from `arr` was removed that **was not the first or last value in the array**.

Given `arr`, return *the removed value*.

Example 1:

```
Input: arr = [5,7,11,13]
Output: 9
Explanation: The previous array was [5,7,9,11,13].
```

Example 2:

```
Input: arr = [15,13,12]
Output: 14
Explanation: The previous array was [15,14,13,12].
```

Constraints:

- `3 <= arr.length <= 1000`
- `0 <= arr[i] <= 10^5`
- The given array is **guaranteed** to be a valid array.

Accepted 18,571 | Submissions 36,265

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months | 6 months ~ 1 year | 1 year ~ 2 years

VMware | 2 | Facebook | 2

Related Topics

Array Math

Hide Hint 1

Assume the sequence is increasing, what if we find the largest consecutive difference?

Hide Hint 2

Is the missing element in the middle of the segment with the largest consecutive difference?

Hide Hint 3

For decreasing sequences, just reverse the array and do a similar process.

```java
class Solution {
    public int missingNumber(int arr[]) {
        int n = arr.length;

        // 1. Get the difference `difference`.
        int difference = (arr[n - 1] - arr[0]) / n;
        int lo = 0;
        int hi = n - 1;

        // Basic binary search template.
        while (lo < hi) {
            int mid = (lo + hi) / 2;

            // All numbers upto `mid` have no missing number, so search on the right side.
            if (arr[mid] == arr[0] + mid * difference) {
                lo = mid + 1;
            }

            // A number is missing before `mid` inclusive of `mid` itself.
            else {
                hi = mid;
            }
        }

        // Index `lo` will be the position with the first incorrect number.
        // Return the value that was supposed to be at this index.
        return arr[0] + difference * lo;
    }
}
```