lee215 ★ 47714 Last Edit: October 19, 2019 10:17 PM 1.3K VIEWS

64

Please reply and upvote now.
Don't have prime membership.
Cannot even read and modify my own post later, when it's locked.

Soluton 1: Arithmetic Sequence Sum

`arithmetic sequence sum = (first + last) * n / 2`
In this problem, the first and last value are not removed.
`first = min(A)`, `last = max(A)`
We can calulate the sum of arithmetic sequence.
The difference between `sum - sum(A)` is the missing number.

Complexity

Time `O(N)`
Space `O(1)`

Java:

```
public int missingNumber(int[] A) {
    int first = A[0], last = A[0], sum = 0, n = A.length;
    for (int a : A) {
        first = Math.min(first, a);
        last = Math.max(last, a);
        sum += a;
    }
    return (first + last) * (n + 1) / 2 - sum;
}
```

C++:

```
int missingNumber(vector<int>& A) {
    int first = A[0], last = A[0], sum = 0, n = A.size();
    for (int a : A) {
        first = min(first, a);
        last = max(last, a);
        sum += a;
    }
    return (first + last) * (n + 1) / 2 - sum;
}
```

Python:

```
def missingNumber(self, A):
    return (min(A) + max(A)) * (len(A) + 1) / 2 - sum(A)
```

Solution 2: Binary Search

Java:

```
public int missingNumber(int[] A) {
    int n = A.length, d = (A[n - 1] - A[0]) / n, left = 0, right = n;
    while (left < right) {
        int mid = (left + right) / 2;
        if (A[mid] == A[0] + d * mid)
            left = mid + 1;
        else
            right = mid;
    }
    return A[0] + d * left;
}
```

C++:

```
int missingNumber(vector<int>& A) {
    int n = A.size(), d = (A[n - 1] - A[0]) / n, left = 0, right = n;
    while (left < right) {
        int mid = (left + right) / 2;
        if (A[mid] == A[0] + d * mid)
            left = mid + 1;
        else
```

Python:

```
def missingNumber(self, A):
    n = len(A)
    d = (A[-1] - A[0]) / n
    left, right = 0, n
    while left < right:
        mid = (left + right) / 2
        if A[mid] == A[0] + d * mid:
            left = mid + 1
        else:
            right = mid
    return A[0] + d * left
```

Type comment here... (Markdown is supported)

Post

DennyZhang ★ 225 October 19, 2019 10:38 PM

`Cannot even read and modify my own post later, when it's locked.`

How come!!! Leetcode should give lee215 unlimited access for lifelong time.

CC @awice

Guys, please thumb up this proposal, if you also agree.

21 Show 1 reply Reply

lee215 ★ 47714 October 19, 2019 9:44 PM

Oh, actually the input is sorted. So we can binary search the answer. Sorry :)

7 Show 1 reply Reply

tlj77 ★ 273 November 7, 2019 2:46 AM

Come on man, do you really have to be so ahead of everyone? This hurts.

1 Reply

dibdidib ★ 158 Last Edit: October 20, 2019 7:59 AM

@lee215: Great ideas. We can make solution 1 "arithmetic sequence sum" ~~have both~~ ~~O(1)~~ ~~time and space~~ marginally faster like this:

```
def missingNumber(self, A):
    return ((A[0] + A[-1]) * (len(A) + 1)) / 2 - sum(A)
```

Thanks for the correction, lee215

1 Show 2 replies Reply

maverick009 ★ 174 November 10, 2019 2:11 AM

@lee215 Amazing Solution man.. Hats off

0 Reply

DenisSchmidt ★ 13 October 25, 2019 4:01 AM

JS Solution

```
const missingNumber = arr => {
  const n = arr.length;

  const s1 = ((n + 1) * (arr[0] + arr[arr.length - 1])) / 2;
  const s2 = arr.reduce((acc, v) => acc + v, 0);
  return s1 - s2;
};
```

0 Reply

xupiter007 ★ 18 October 20, 2019 7:20 AM

I guess there is a typo in your Solution 1 formula, it should be as listed below:

```
arithmetic sequence sum = (first + last) * (n + 1) / 2
```

Also, we don't need to calculate min/max because input data is already sorted.

0 Show 1 reply Reply

MichaelZ ★ 633 October 20, 2019 5:10 AM

Another solution:
There is only 1 pair which has twice the difference as other pairs

```
int missingNumber(vector<int>& arr) {
    int dif=arr[1]-arr[0];
    for(int i=2;i<arr.size();i++) {
        int cur=arr[i]-arr[i-1];
        if(cur==2*dif) return (arr[i]+arr[i-1])/2;
        else if(cur*2==dif) return (arr[1]+arr[0])/2;
    }
    return -1;
```

Read More

0 Show 2 replies Reply