(/) Explore(/explore/) Problems(/problemset/all/)

Interview (/contest/)

Discuss(/discuss/)

Store (/subscribe? ref=nb_npl)

ıt-there-ıs-a-valıdpartition-for-the-

array/)



2817. Minimum Absolute Difference Between Elements With Constraint

My Submissions (/contest/weekly-contest-358/problems/minimum-absolute-difference-between-elements-with-constraint/submissions/)

Back to Contest (/contest/weekly-contest-358/)

You are given a **0-indexed** integer array nums and an integer x.

Find the minimum absolute difference between two elements in the array that are at least x indices apart.

In other words, find two indices i and j such that abs(i - j) >= x and abs(nums[i] - nums[j]) is minimized.

Return an integer denoting the minimum absolute difference between two elements that are at least x indices apart.

User Accepted:	3735
User Tried:	9728
Total Accepted:	3901
Total Submissions:	23827
Difficulty:	Medium

Example 1:

```
Input: nums = [4,3,2,4], x = 2
Output: 0
Explanation: We can select nums[0] = 4 and nums[3] = 4.
They are at least 2 indices apart, and their absolute difference is the minimum, 0.
It can be shown that 0 is the optimal answer.
```

Example 2:

```
Input: nums = [5,3,2,10,15], x = 1
Output: 1
Explanation: We can select nums[1] = 3 and nums[2] = 2.
They are at least 1 index apart, and their absolute difference is the minimum, 1.
It can be shown that 1 is the optimal answer.
```

Example 3:

```
Input: nums = [1,2,3,4], x = 3
Output: 3
Explanation: We can select nums[0] = 1 and nums[3] = 4.
They are at least 3 indices apart, and their absolute difference is the minimum, 3.
It can be shown that 3 is the optimal answer.
```

Constraints:

- 1 <= nums.length <= 10⁵
- 1 <= nums[i] <= 10⁹
 0 <= x < nums.length

Discuss (https://leetcode.com/problems/minimum-absolute-difference-between-elements-with-constraint/discuss)

```
₫ 2 •
JavaScript
1 ▼ function SegmentTreeRMAXQ(n) {
        let h = Math.ceil(Math.log2(n)), len = 2 * 2 ** h, a = Array(len).fill(Number.MIN_SAFE_INTEGER);
2
3
       h = 2 ** h;
4
        return { update, maxx, firstle, tree }
5 ,
        function update(pos, v) {
6
            a[h + pos] = v;
7
            for (let i = parent(h + pos); i >= 1; i = parent(i)) pushup(i);
8
9,
        function pushup(i) {
10
            a[i] = Math.max(a[left(i)], a[right(i)]); // [max .... min]
11
        function maxx(l, r) { // [L, R)}
12 •
13
            let max = Number.MIN_SAFE_INTEGER;
14
            if (l >= r) return max;
15
            1 += h:
16
            r += h;
```

```
for (; l < r; l = parent(l), r = parent(r)) {
17 ▼
18
                 if (l \& 1) max = Math.max(max, a[l++]);
                 if (r \& 1) \max = Math.max(\max, a[--r]);
19
20
             }
21
             return max;
22
23 •
         function firstle(l, v) {
24
             if (l >= h) return -1;
25
             let cur = h + 1;
26 •
             while (1) {
                 if (a[cur] <= v) {
27 ▼
28
                      if (cur >= h) return cur - h;
29
                      cur = left(cur);
30 ▼
                 } else {
31
                      cur++:
32
                      if ((cur \& cur - 1) == 0) return -1;
33
                      if (cur \% 2 == 0) cur = parent(cur);
34
                 }
35
             }
36
37 ▼
         function parent(i) {
38
             return i >> 1;
39
40 ،
         function left(i) {
             return 2 * i;
41
42
43 ▼
         function right(i) {
             return 2 * i + 1;
44
45
46
         function tree() {
47
             return a;
48
        }
49
    }
50
51
    const minAbsoluteDifference = (a, x) \Rightarrow \{
        a = a.map((x, i) \Rightarrow [x, i]).sort((x, y) \Rightarrow x[0] - y[0] || x[1] - y[1]);
52
53
        let n = a.length, stmax = new SegmentTreeRMAXQ(n), res = Number.MAX_SAFE_INTEGER;
54
         for (const [v, i] of a) {
55
             stmax.update(i, v);
56
             let l = i - x, r = i + x, maxL, maxR;
             // query [0, l] [r, n-1]
57
             if (l >= 0) maxL = stmax.maxx(0, l + 1);
58
59
             if (r < n) \max R = \operatorname{stmax.maxx}(r, n);
60
             let d1 = v - maxL, d2 = v - maxR;
61
             if (!Number.isNaN(d1)) res = Math.min(res, d1);
62
             if (!Number.isNaN(d2)) res = Math.min(res, d2);
        }
63
64
        return res;
65
    };
```

☐ Custom Testcase

Use Example Testcases

○ Run (

△ Submit

Submission Result: Accepted (/submissions/detail/1019913427/) ?

More Details > (/submissions/detail/1019913427/)

Share your acceptance!

Copyright © 2023 LeetCode

Copyright © 2023 LeetCode

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

United States (/region)