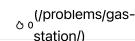
--(/) Explore(/explore/) Problems(/problemset/all/) Interview Contest







0

0

0

0

(Hard)

6290. Maximize the Minimum Powered City

My Submissions (/contest/biweekly-contest-95/problems/maximize-the-minimum-powered-city/submissions/)

Back to Contest (/contest/biweekly-contest-95/)

User Accepted:

Total Accepted:

Total Submissions:

User Tried:

Difficulty:

You are given a 0-indexed integer array stations of length n, where stations[i] represents the number of power stations in the ith city.

Each power station can provide power to every city in a fixed **range**. In other words, if the range is denoted by r, then a power station at city i can provide power to all cities j such that |i - j| <= r and 0 <= i, j <= n - 1.

• Note that |x| denotes **absolute** value. For example, |7 - 5| = 2 and |3 - 10| = 7.

The **power** of a city is the total number of power stations it is being provided power from.

The government has sanctioned building k more power stations, each of which can be built in any city, and have the same range as the pre-existing ones.

Given the two integers r and k, return the maximum possible minimum power of a city, if the additional power stations are built optimally.

Note that you can build the k power stations in multiple cities.

Example 1:

```
Input: stations = [1,2,4,5,0], r = 1, k = 2
Output: 5
Explanation:
One of the optimal ways is to install both the power stations at city 1.
So stations will become [1,4,4,5,0].
- City 0 is provided by 1 + 4 = 5 power stations.
- City 1 is provided by 1 + 4 + 4 = 9 power stations.
- City 2 is provided by 4 + 4 + 5 = 13 power stations.
- City 3 is provided by 5 + 4 = 9 power stations.
- City 4 is provided by 5 + 0 = 5 power stations.
So the minimum power of a city is 5.
Since it is not possible to obtain a larger power, we return 5.
```

Example 2:

```
Input: stations = [4,4,4,4], r = 0, k = 3
Output: 4
Explanation:
It can be proved that we cannot make the minimum power of a city greater than 4.
```

Constraints:

```
n == stations.length
1 <= n <= 10<sup>5</sup>
0 <= stations[i] <= 10<sup>5</sup>
0 <= r <= n - 1</li>
0 <= k <= 10<sup>9</sup>
```

```
JavaScript 🔻
```

```
const preSum = (a) => { let pre = [0]; for (let i = 0; i < a.length; i++) { pre.push(pre[i] + a[i]); } return pre; };
2
    const subArraySum = (a, l, r) \Rightarrow a[r + 1] - a[l];
3
 4
    let a, n, r, k, pre, p;
    const maxPower = (A, R, K) \Rightarrow {
 5 ,
        a = A, r = R, k = K, n = a.length, pre = preSum(a), p = compute();
 6
 7
        return BinarySearch(0, 2e10);
 8
    };
 9
10 v const compute = () => {
        let p = Array(n).fill(0);
11
12 •
        for (let i = 0; i < n; i++) {
            let L = Math.max(0, i - r), R = Math.min(n - 1, i + r);
13
```

```
14
             let rangeSum = subArraySum(pre, L, R);
            p[i] = rangeSum;
15
16
17
        return p;
18
    };
19
20 v const BinarySearch = (low, high) ⇒ {
        while (low <= high) {
21 🔻
22
             let mid = low + parseInt((high - low) / 2);
23 🔻
             if (possible(mid)) {
24
                 low = mid + 1;
25 🔻
             } else {
26
                 high = mid - 1;
27
        }
28
29
        return high;
30
    };
31
    const cover = (r) \Rightarrow 2 * r + 1;
32
33
34 v const possible = (v) ⇒ {
35
        let d = Array(n + 1).fill(0), cur = 0, sum = 0;
        for (let i = 0; i < n; i++) {
36 🔻
             cur += d[i];
37
             if (cur + p[i] < v) {
38 ▼
39
                 let needPower = v - p[i] - cur;
40
                 sum += needPower;
                 let idx = Math.min(i + cover(r), n);
41
                 d[idx] -= needPower;
42
43
                 cur += needPower;
44
45
        }
46
        return sum <= k;
47
    };
```

☐ Custom Testcase

Use Example Testcases

○ Run

⚠ Submit

Submission Result: Accepted (/submissions/detail/873450461/) 2

More Details > (/submissions/detail/873450461/)

Share your acceptance!

∢1

Copyright © 2023 LeetCode

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

United States (/region)