6325. Minimum Time to Repair Cars

My Submissions (/contest/biweekly-contest-100/problems/minimum-time-to-repair-cars/submissions/) Back to Contest (/contest/biweekly-contest-100/)

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

You are given an integer array `ranks` representing the **ranks** of some mechanics. ranks_i is the rank of the i^{th} mechanic. A mechanic with a rank `r` can repair n cars in $r * n^2$ minutes.

You are also given an integer `cars` representing the total number of cars waiting in the garage to be repaired.

Return *the **minimum** time taken to repair all the cars.*

Note: All the mechanics can repair the cars simultaneously.

Example 1:

```
Input: ranks = [4,2,3,1], cars = 10
Output: 16
Explanation:
- The first mechanic will repair two cars. The time required is 4 * 2 * 2 = 16 minutes.
- The second mechanic will repair two cars. The time required is 2 * 2 * 2 = 8 minutes.
- The third mechanic will repair two cars. The time required is 3 * 2 * 2 = 12 minutes.
- The fourth mechanic will repair four cars. The time required is 1 * 4 * 4 = 16 minutes.
It can be proved that the cars cannot be repaired in less than 16 minutes.
```

Example 2:

```
Input: ranks = [5,1,8], cars = 6
Output: 16
Explanation:
- The first mechanic will repair one car. The time required is 5 * 1 * 1 = 5 minutes.
- The second mechanic will repair four cars. The time required is 1 * 4 * 4 = 16 minutes.
- The third mechanic will repair one car. The time required is 8 * 1 * 1 = 8 minutes.
It can be proved that the cars cannot be repaired in less than 16 minutes.
```

Constraints:

- $1 <= \text{ranks.length} <= 10^5$
- $1 <= \text{ranks[i]} <= 100$
- $1 <= \text{cars} <= 10^6$

JavaScript

```
let a, k;
const repairCars = (A, K) => {
    a = A, k = K;
    return BinarySearch(0, Number.MAX_SAFE_INTEGER);
};

const BinarySearch = (low, high) => {
    while (low <= high) {
        let mid = low + parseInt((high - low) / 2);
        if (possible(mid)) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }
    return low;
};

const possible = (v) => {
    let t = 0;
    for (const x of a) t += parseInt(parseInt(v / x) ** 0.5);
    return t < k;
};
```

☐ **Custom Testcase** Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/917650581/) More Details (/submissions/detail/917650581/)

Share your acceptance!