5289. Fair Distribution of Cookies

My Submissions (/contest/weekly-contest-297/problems/fair-distribution-of-cookies/submissions/) Back to Contest (/contest/weekly-contest-297/)

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

You are given an integer array `cookies`, where `cookies[i]` denotes the number of cookies in the i^{th} bag. You are also given an integer `k` that denotes the number of children to distribute **all** the bags of cookies to. All the cookies in the same bag must go to the same child and cannot be split up.

The **unfairness** of a distribution is defined as the **maximum total** cookies obtained by a single child in the distribution.

Return *the **minimum** unfairness of all distributions*.

Example 1:

```
Input: cookies = [8,15,10,20,8], k = 2
Output: 31
Explanation: One optimal distribution is [8,15,8] and [10,20]
- The 1st child receives [8,15,8] which has a total of 8 + 15 + 8 = 31 cookies.
- The 2nd child receives [10,20] which has a total of 10 + 20 = 30 cookies.
The unfairness of the distribution is max(31,30) = 31.
It can be shown that there is no distribution with an unfairness less than 31.
```

Example 2:

```
Input: cookies = [6,1,3,2,2,4,1,2], k = 3
Output: 7
Explanation: One optimal distribution is [6,1], [3,2,2], and [4,1,2]
- The 1st child receives [6,1] which has a total of 6 + 1 = 7 cookies.
- The 2nd child receives [3,2,2] which has a total of 3 + 2 + 2 = 7 cookies.
- The 3rd child receives [4,1,2] which has a total of 4 + 1 + 2 = 7 cookies.
The unfairness of the distribution is max(7,7,7) = 7.
It can be shown that there is no distribution with an unfairness less than 7.
```

Constraints:

- `2 <= cookies.length <= 8`
- `1 <= cookies[i] <= 10^5`
- `2 <= k <= cookies.length`

JavaScript

```
const initializeGraphSet = (n) => { let g = []; for (let i = 0; i < n; i++) { g.push(new Set()); } return g; };

let a, n, k, res, sum;
const distributeCookies = (cookies, K) => {
    a = cookies, k = K, n = a.length, res = Number.MAX_SAFE_INTEGER;
    dfs(0, initializeGraphSet(n));
    return res;
};

const dfs = (pos, cur) => {
    if (pos == n) {
        let v = unfairness(cur);
        // pr(cur, debug(cur), unfairness(cur));
        res = Math.min(res, v);
        return;
    }
    for (let i = 0; i < k; i++) {
        cur[i].add(pos);
        dfs(pos + 1, cur);
        cur[i].delete(pos);
        if (cur[i].size == 0) break;
    }
};

// const debug = (g) => {
//     let d = [];
```

```
//     g.map(se => {
//         let t = [];
//         for (const idx of se) t.push(a[idx]);
//         d.push(t);
//     });
//     return d;
// };

const unfairness = (g) => Math.max(...g.map(se => sm(se)));
const sm = (se) => { let sum = 0; for (const idx of se) sum += a[idx]; return sum; };
```

Custom Testcase | Use Example Testcases

Run | Submit

Submission Result: Accepted (/submissions/detail/720250741/) | More Details > (/submissions/detail/720250741/)

Share your acceptance!