2933. High-Access Employees

My Submissions (/contest/weekly-contest-371/problems/high-access-employees/submissions/) Back to Contest (/contest/weekly-contest-371/)

You are given a 2D **0-indexed** array of strings, `access_times`, with size `n`. For each `i` where `0 <= i <= n - 1`, `access_times[i][0]` represents the name of an employee, and `access_times[i][1]` represents the access time of that employee. All entries in `access_times` are within the same day.

The access time is represented as **four digits** using a **24-hour** time format, for example, `"0800"` or `"2250"`.

An employee is said to be **high-access** if he has accessed the system **three or more** times within a **one-hour period**.

Times with exactly one hour of difference are **not** considered part of the same one-hour period. For example, `"0815"` and `"0915"` are not part of the same one-hour period.

User Accepted:	6140
User Tried:	7653
Total Accepted:	6356
Total Submissions:	19012
Difficulty:	Medium

Access times at the start and end of the day are **not** counted within the same one-hour period. For example, `"0005"` and `"2350"` are not part of the same one-hour period.

Return *a list that contains the names of **high-access** employees with any order you want.*

Example 1:

```
Input: access_times = [["a","0549"],["b","0457"],["a","0532"],["a","0621"],["b","0540"]]
Output: ["a"]
Explanation: "a" has three access times in the one-hour period of [05:32, 06:31] which are 05:32, 05:49, and 06:21.
But "b" does not have more than two access times at all.
So the answer is ["a"].
```

Example 2:

```
Input: access_times = [["d","0002"],["c","0808"],["c","0829"],["e","0215"],["d","1508"],["d","1444"],["d","1410"],["
Output: ["c","d"]
Explanation: "c" has three access times in the one-hour period of [08:08, 09:07] which are 08:08, 08:09, and 08:29.
"d" has also three access times in the one-hour period of [14:10, 15:09] which are 14:10, 14:44, and 15:08.
However, "e" has just one access time, so it can not be in the answer and the final answer is ["c","d"].
```

Example 3:

```
Input: access_times = [["cd","1025"],["ab","1025"],["cd","1046"],["cd","1055"],["ab","1124"],["ab","1120"]]
Output: ["ab","cd"]
Explanation: "ab" has three access times in the one-hour period of [10:25, 11:24] which are 10:25, 11:20, and 11:24.
"cd" has also three access times in the one-hour period of [10:25, 11:24] which are 10:25, 10:46, and 10:55.
So the answer is ["ab","cd"].
```

Constraints:

- `1 <= access_times.length <= 100`
- `access_times[i].length == 2`
- `1 <= access_times[i][0].length <= 10`
- `access_times[i][0]` consists only of English small letters.
- `access_times[i][1].length == 4`
- `access_times[i][1]` is in 24-hour time format.
- `access_times[i][1]` consists only of `'0'` to `'9'`.

Discuss (https://leetcode.com/problems/high-access-employees/discuss)

JavaScript

```javascript
const findHighAccessEmployees = (g) => {
    let m = new Map(), res = [];
    for (const [c, t] of g) {
        if (!m.has(c)) m.set(c, []);
        m.get(c).push(t.slice(0, 2) + ":" + t.slice(2))
    }
    for (const [, a] of m) a.sort();
    for (const [c, a] of m) {
        for (let i = 0; i + 2 < a.length; i++) {
            let minutesDiff = calMinutes(a[i], a[i + 2]);
            if (minutesDiff < 60) {
                res.push(c);
                break;
            }
        }
    }
    return res;
};

const calMinutes = (s, t) => { // HH:MM
    let [hs, ms] = s.split(":").map(Number), [ht, mt] = t.split(":").map(Number);
    let ds = new Date(2023, 1, 1, hs, ms, 0), dt = new Date(2023, 1, 1, ht, mt, 0);
    let seconds = Math.abs(ds.getTime() - dt.getTime()) / 1000, miutes = seconds / 60;
    return miutes;
};
```

Custom Testcase | Use Example Testcases

Run | Submit

Submission Result: Accepted (/submissions/detail/1097102431/) | More Details > (/submissions/detail/1097102431/)

Share your acceptance!