5546. Slowest Key

My Submissions (/contest/weekly-contest-212/problems/slowest-key/submissions/) Back to Contest (/contest/weekly-contest-212/)

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

A newly designed keypad was tested, where a tester pressed a sequence of `n` keys, one at a time.

You are given a string `keysPressed` of length `n`, where `keysPressed[i]` was the `i`th key pressed in the testing sequence, and a sorted list `releaseTimes`, where `releaseTimes[i]` was the time the `i`th key was released. Both arrays are **0-indexed**. The `0`th key was pressed at the time `0`, and every subsequent key was pressed at the **exact** time the previous key was released.

The tester wants to know the key of the keypress that had the **longest duration**. The `i`th keypress had a **duration** of `releaseTimes[i] - releaseTimes[i - 1]`, and the `0`th keypress had a duration of `releaseTimes[0]`.

Note that the same key could have been pressed multiple times during the test, and these multiple presses of the same key **may not** have had the same **duration**.

*Return the key of the keypress that had the **longest duration**. If there are multiple such keypresses, return the lexicographically largest key of the keypresses.*

Example 1:

```
Input: releaseTimes = [9,29,49,50], keysPressed = "cbcd"
Output: "c"
Explanation: The keypresses were as follows:
Keypress for 'c' had a duration of 9 (pressed at time 0 and released at time 9).
Keypress for 'b' had a duration of 29 - 9 = 20 (pressed at time 9 right after the release of the previous character and re
Keypress for 'c' had a duration of 49 - 29 = 20 (pressed at time 29 right after the release of the previous character and
Keypress for 'd' had a duration of 50 - 49 = 1 (pressed at time 49 right after the release of the previous character and r
The longest of these was the keypress for 'b' and the second keypress for 'c', both with duration 20.
'c' is lexicographically larger than 'b', so the answer is 'c'.
```

Example 2:

```
Input: releaseTimes = [12,23,36,46,62], keysPressed = "spuda"
Output: "a"
Explanation: The keypresses were as follows:
Keypress for 's' had a duration of 12.
Keypress for 'p' had a duration of 23 - 12 = 11.
Keypress for 'u' had a duration of 36 - 23 = 13.
Keypress for 'd' had a duration of 46 - 36 = 10.
Keypress for 'a' had a duration of 62 - 46 = 16.
The longest of these was the keypress for 'a' with duration 16.
```

Constraints:

- `releaseTimes.length == n`
- `keysPressed.length == n`
- `2 <= n <= 1000`
- `0 <= releaseTimes[i] <=` 10^9
- `releaseTimes[i] < releaseTimes[i+1]`
- `keysPressed` contains only lowercase English letters.

JavaScript

```
/**
 * @param {number[]} releaseTimes
 * @param {string} keysPressed
 * @return {character}
 */
const slowestKey = (releaseTimes, keysPressed) => {
    let n = releaseTimes.length;
    let map = new Map();
    for (let i = 0; i < n; i++) {
        let tmp = 0;
        if (i == 0) {
            tmp = releaseTimes[0];
        } else {
            tmp = releaseTimes[i] - releaseTimes[i - 1];
        }
        if (map.has(keysPressed[i])) {
            let max = Math.max(tmp, map.get(keysPressed[i]));
            map.set(keysPressed[i], max);
        } else {
            map.set(keysPressed[i], tmp);
        }
    }
    let resMap = sortMap(map);
    return resMap.entries().next().value[0];
};


const sortMap = (map) => {
    return new Map([...map].sort((a, b) => {
        if (a[1] == b[1]) return b[0].localeCompare(a[0]);
        return b[1] - a[1];
    }));
};
```

☐ **Custom Testcase** Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/412805956/) More Details (/submissions/detail/412805956/)

Share your acceptance!