链滴

Leetcode 每日一题：1692. 按键持续时间最长的键

作者：Hildaquan

原文链接：https://ld246.com/article/1641708213021

来源网站：链滴

自己的思路

使用 哈希法 ， 将字符映射到数组的索引中，然后在遍历的时候记录最大值。

最后通过两个循环找到合适下标

第一个循环找到出现次数最大的值

第二个循环根据这个最大值找到下标。由于要找到按照字符排序大小输出了答案，所以需要逆序查找

时间复杂度 O(3N)，空间复杂度 O(1)

```
class Solution {
    public char slowestKey(int[] releaseTimes, String keysPressed) {
        int[] map = new int[26];
        Arrays.fill(map, 0);
        int index = keysPressed.charAt(0) - 'a';

        map[index] = releaseTimes[0];
        for (int i = 1; i < keysPressed.length(); i++) {
            index = keysPressed.charAt(i) - 'a';
            map[index] = Math.max(map[index], releaseTimes[i] - releaseTimes[i - 1]);
        }

        int max = 0;
        int maxIndex = 0;
        for (int i = 0; i < map.length; i++) {
            if (max < map[i]) {
                max = map[i];
            }
        }
        for (int i = map.length - 1; i >= 0; i--) {
            if (map[i] == max) {
                maxIndex = i;
                break;
            }
        }

        return (char)(maxIndex + 'a');

    }
}
```

哈希法 不一定非得用 Map 集合 ， 能使用数组的就使用数组，因为数组本来就是哈希法最基础的实现具。

我看到我自己实现的哈希算法，在整体速度上排第三。后面的都是使用 HashMap 来做的。

别人的思路

和自己的差不多，但是可以对我的算法进行优化。时间复杂度压缩到 O(N)，将我额外的两个循环的断逻辑放到了第一个循环上，实现效率上的提升。

```
class Solution {
    public char slowestKey(int[] releaseTimes, String keysPressed) {
        int time = releaseTimes[0];
        char res = keysPressed.charAt(0);
        for (int i = 1; i < keysPressed.length(); i++) {
            int cur = releaseTimes[i] - releaseTimes[i - 1];
            if (time < cur || (time == cur && res < keysPressed.charAt(i))) {
                time = cur;
                res = keysPressed.charAt(i);
            }
        }

        return res;
    }
}
```

这样看，优化过后的算法就是一个 贪心算法。不断取最大值，获取当前的局部最优，最后得到全局最
。