2349. Design a Number Container System

My Submissions (/contest/biweekly-contest-83/problems/design-a-number-container-system/submissions/) Back to Contest (/contest/biweekly-contest-83/)

User Accepted:	6731
User Tried:	8685
Total Accepted:	6952
Total Submissions:	18142
Difficulty:	Medium

Design a number container system that can do the following:

- **Insert** or **Replace** a number at the given index in the system.
- **Return** the smallest index for the given number in the system.

Implement the `NumberContainers` class:

- `NumberContainers()` Initializes the number container system.
- `void change(int index, int number)` Fills the container at `index` with the `number`. If there is already a number at that `index`, replace it.
- `int find(int number)` Returns the smallest index for the given `number`, or `-1` if there is no index that is filled by `number` in the system.

Example 1:

```
Input
["NumberContainers", "find", "change", "change", "change", "change", "find", "change", "find"]
[[], [10], [2, 10], [1, 10], [3, 10], [5, 10], [10], [1, 20], [10]]
Output
[null, -1, null, null, null, null, 1, null, 2]

Explanation
NumberContainers nc = new NumberContainers();
nc.find(10); // There is no index that is filled with number 10. Therefore, we return -1.
nc.change(2, 10); // Your container at index 2 will be filled with number 10.
nc.change(1, 10); // Your container at index 1 will be filled with number 10.
nc.change(3, 10); // Your container at index 3 will be filled with number 10.
nc.change(5, 10); // Your container at index 5 will be filled with number 10.
nc.find(10); // Number 10 is at the indices 1, 2, 3, and 5. Since the smallest index that is filled with 10 is 1, we return 1.
nc.change(1, 20); // Your container at index 1 will be filled with number 20. Note that index 1 was filled with 10 and then rep
nc.find(10); // Number 10 is at the indices 2, 3, and 5. The smallest index that is filled with 10 is 2. Therefore, we return 2
```

Constraints:

- `1 <= index, number <= 10^9`
- At most `10^5` calls will be made **in total** to `change` and `find`.

Discuss (https://leetcode.com/problems/design-a-number-container-system/discuss)

JavaScript

```
function Bisect() {
    return { insort_right, insort_left, bisect_left, bisect_right }
    function insort_right(a, x, lo = 0, hi = null) {
        lo = bisect_right(a, x, lo, hi);
        a.splice(lo, 0, x);
    }
    function bisect_right(a, x, lo = 0, hi = null) { // > upper_bound
        if (lo < 0) throw new Error('lo must be non-negative');
        if (hi == null) hi = a.length;
        while (lo < hi) {
            let mid = parseInt((lo + hi) / 2);
            a[mid] > x ? hi = mid : lo = mid + 1;
        }
        return lo;
    }
    function insort_left(a, x, lo = 0, hi = null) {
        lo = bisect_left(a, x, lo, hi);
        a.splice(lo, 0, x);
    }
```

```
    function bisect_left(a, x, lo = 0, hi = null) { // >= lower_bound
        if (lo < 0) throw new Error('lo must be non-negative');
        if (hi == null) hi = a.length;
        while (lo < hi) {
            let mid = parseInt((lo + hi) / 2);
            a[mid] < x ? lo = mid + 1 : hi = mid;
        }
        return lo;
    }
}

function TreeSet(elements) {
    let ts = [], se = new Set(), bisect = new Bisect();
    initialize();
    return { add, first, last, poll, pollLast, floor, ceiling, lower, higher, remove, contains, size, clear, show };
    function initialize() {
        if (elements) {
            for (const e of elements) {
                if (!se.has(e)) {
                    bisect.insort_right(ts, e);
                    se.add(e);
                }
            }
        }
    }
    function add(e) {
        if (!se.has(e)) {
            bisect.insort_right(ts, e);
            se.add(e);
        }
    }
    function first() {
        return ts[0];
    }
    function last() {
        return ts[ts.length - 1];
    }
    function poll() {
        let res = ts[0];
        ts.splice(0, 1);
        se.delete(res);
        return res;
    }
    function pollLast() {
        let res = ts.pop();
        se.delete(res);
        return res;
    }
    function ceiling(e) { // >= lower_bound
        let idx = bisect.bisect_right(ts, e);
        let res = ts[idx - 1] == e ? e : ts[bisect.bisect_right(ts, e)];
        return res == undefined ? null : res;
    }
    function higher(e) { // > upper_bound
        let idx = bisect.bisect_right(ts, e);
        let res = ts[idx] > e ? ts[idx] : ts[bisect.bisect_right(ts, e) + 1];
        return res == undefined ? null : res;
    }
    function floor(e) { // <=
        let idx = bisect.bisect_left(ts, e);
        let res = ts[idx] == e ? e : ts[bisect.bisect_left(ts, e) - 1];
        return res == undefined ? null : res;
    }
    function lower(e) { // <
        let idx = bisect.bisect_left(ts, e);
        let res = ts[idx] < e ? ts[idx] : ts[bisect.bisect_left(ts, e) - 1];
        return res == undefined ? null : res;
    }
    function remove(e) {
        let idx = bisect.bisect_left(ts, e);
        if (ts[idx] == e) ts.splice(idx, 1);
        se.delete(e);
    }
    function contains(e) {
        return se.has(e);
    }
    function size() {
```

```
97          return ts.length;
98      }
99      function clear() {
100         ts = [];
101         se.clear();
102     }
103     function show() {
104         return ts;
105     }
106 }
107
108 function NumberContainers() {
109     let im = new Map(), vm = new Map();
110     return { change, find }
111     function change(index, number) {
112         if (!vm.has(number)) vm.set(number, new TreeSet());
113         vm.get(number).add(index);
114         if (im.has(index)) {
115             let oldNumber = im.get(index);
116             if (oldNumber != number) vm.get(oldNumber).remove(index);
117         }
118         im.set(index, number);
119     }
120     function find(number) {
121         if (!vm.has(number) || vm.get(number).size() == 0) return -1;
122         return vm.get(number).first();
123     }
124 }
```

Custom Testcase | Use Example Testcases

Run | Submit

Submission Result: Accepted (/submissions/detail/758698402/) | More Details > (/submissions/detail/758698402/)

Share your acceptance!