731. My Calendar II

You are implementing a program to use as your calendar. We can add a new event if adding the event will not cause a **triple booking**.

A **triple booking** happens when three events have some non-empty intersection (i.e., some moment is common to all the three events.).

The event can be represented as a pair of integers `start` and `end` that represents a booking on the half-open interval `[start, end)`, the range of real numbers `x` such that `start <= x < end`.

Implement the `MyCalendarTwo` class:

- `MyCalendarTwo()` Initializes the calendar object.
- `boolean book(int start, int end)` Returns `true` if the event can be added to the calendar successfully without causing a **triple booking**. Otherwise, return `false` and do not add the event to the calendar.

Example 1:

```
Input
["MyCalendarTwo", "book", "book", "book", "book", "book", "book"]
[[], [10, 20], [50, 60], [10, 40], [5, 15], [5, 10], [25, 55]]
Output
[null, true, true, true, false, true, true]

Explanation
MyCalendarTwo myCalendarTwo = new MyCalendarTwo();
myCalendarTwo.book(10, 20); // return True, The event can be booked.
myCalendarTwo.book(50, 60); // return True, The event can be booked.
myCalendarTwo.book(10, 40); // return True, The event can be double booked.
myCalendarTwo.book(5, 15);  // return False, The event cannot be booked, because it
would result in a triple booking.
myCalendarTwo.book(5, 10); // return True, The event can be booked, as it does not
use time 10 which is already double booked.
myCalendarTwo.book(25, 55); // return True, The event can be booked, as the time in
[25, 40) will be double booked with the third event, the time [40, 50) will be
single booked, and the time [50, 55) will be double booked with the second event.
```

Constraints:

- `0 <= start < end <= 10^9`
- At most `1000` calls will be made to `book`.

731. My Calendar II

```
/*
    Active Interval counting
    Time complexity: O(nm)
    Space complexity: O(m)
    n: total number of events
    m: number of starting points and and points
*/

class MyCalendarTwo {
    private:
        std::map<int,int> dp;
    public:
        MyCalendarTwo() {

        }

        bool book(int start, int end) {
            dp[start]++;
            dp[end]--;
            int s = 0;
            for (auto v: dp){
                s += v.second;
                if (s >= 3) {
                    dp[start]--;
                    dp[end]++;
                    return false;
                }
            }
            return true;
        }
};
```