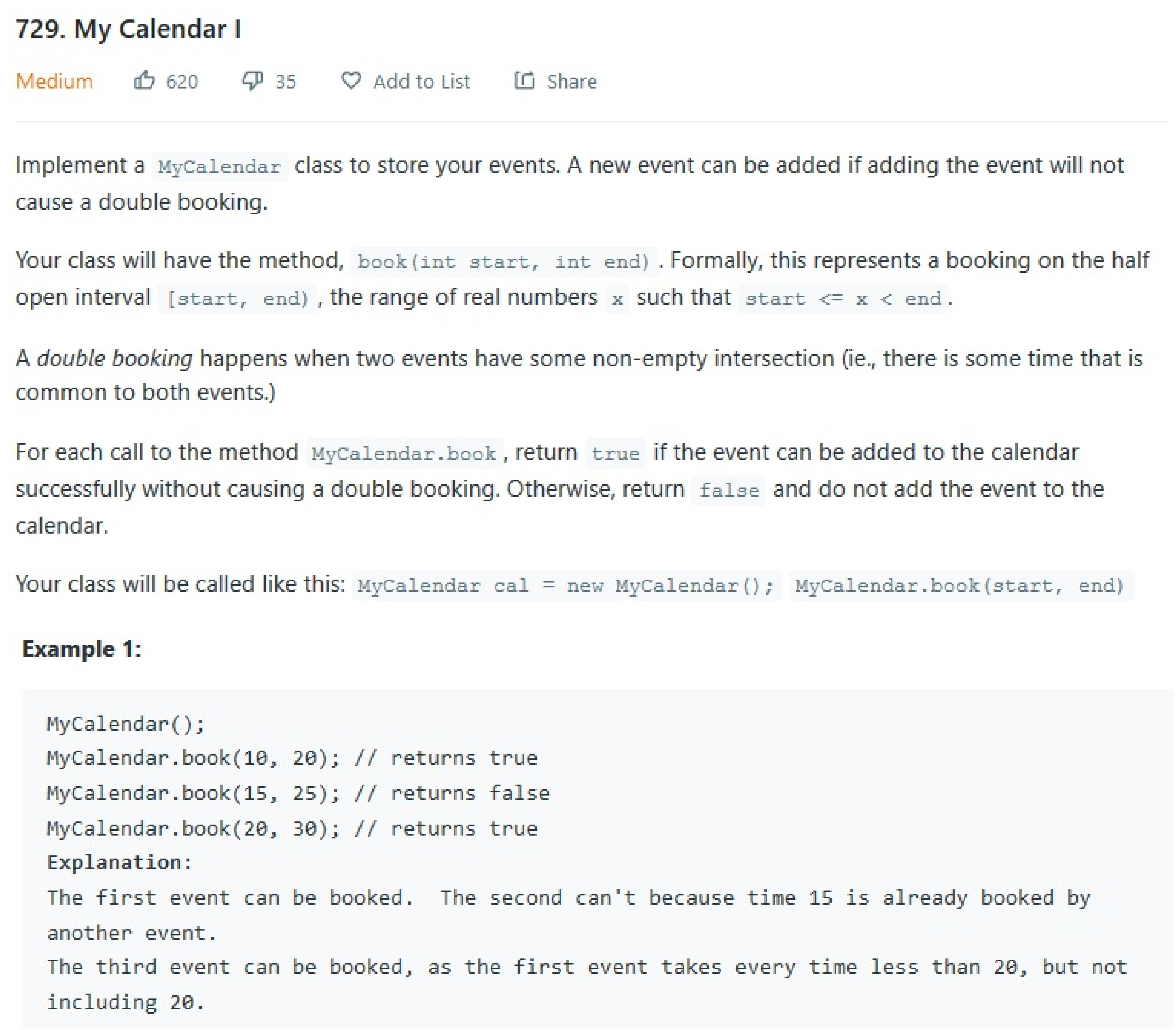


729. My Calendar I

Medium 620 35 Add to List Share

Implement a `MyCalendar` class to store your events. A new event can be added if adding the event will not cause a double booking.

Your class will have the method, `book(int start, int end)`. Formally, this represents a booking on the half open interval `[start, end)`, the range of real numbers `x` such that `start <= x < end`.

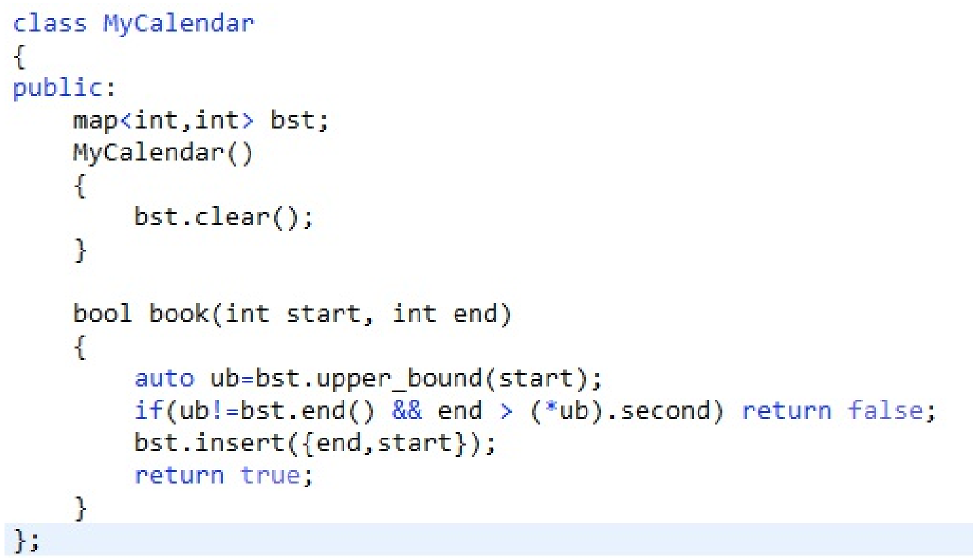
A *double booking* happens when two events have some non-empty intersection (ie., there is some time that is common to both events.)

For each call to the method `MyCalendar.book`, return `true` if the event can be added to the calendar successfully without causing a double booking. Otherwise, return `false` and do not add the event to the calendar.

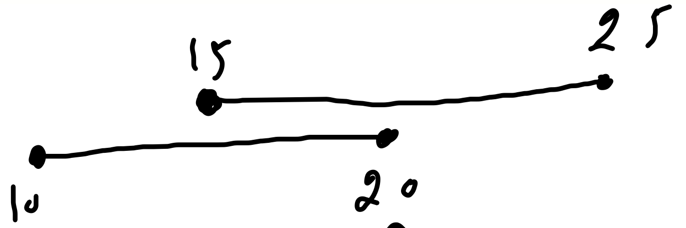
Your class will be called like this: `MyCalendar cal = new MyCalendar();` `MyCalendar.book(start, end)`

Example 1:

```
MyCalendar();
MyCalendar.book(10, 20); // returns true
MyCalendar.book(15, 25); // returns false
MyCalendar.book(20, 30); // returns true
Explanation:
The first event can be booked.  The second can't because time 15 is already booked by
another event.
The third event can be booked, as the first event takes every time less than 20, but not
including 20.
```



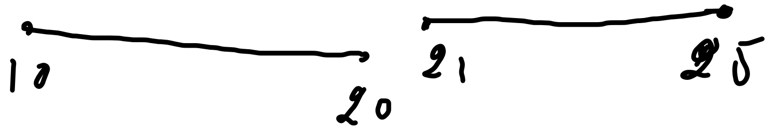
```
class MyCalendar
{
public:
    map<int,int> bst;
    MyCalendar()
    {
        bst.clear();
    }

    bool book(int start, int end)
    {
        auto ub=bst.upper_bound(start);
        if(ub!=bst.end() && end > (*ub).second) return false;
        bst.insert({end,start});
        return true;
    }
};
```



↑

→ iska upper bound 25

→ agar mera end upper bound ke start se bada hai. it means overlap hai.



-> isme upper bound 25 hai but end chhota hai to no overlap.