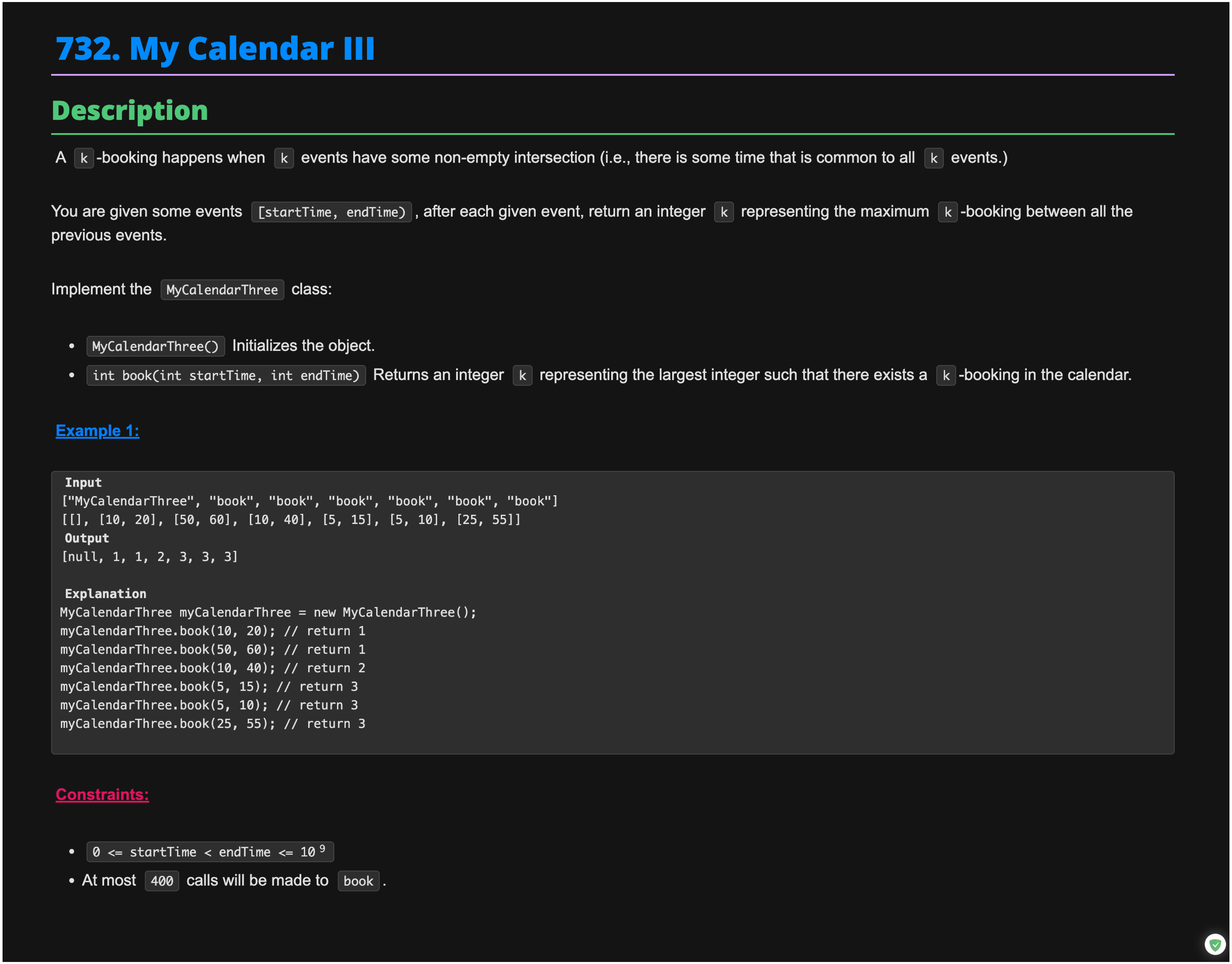

732. My Calendar III

Description

A `k`-booking happens when `k` events have some non-empty intersection (i.e., there is some time that is common to all `k` events.)

You are given some events `[startTime, endTime)`, after each given event, return an integer `k` representing the maximum `k`-booking between all the previous events.

Implement the `MyCalendarThree` class:

- `MyCalendarThree()` Initializes the object.
- `int book(int startTime, int endTime)` Returns an integer `k` representing the largest integer such that there exists a `k`-booking in the calendar.

Example 1:

```
Input
["MyCalendarThree", "book", "book", "book", "book", "book", "book"]
[[], [10, 20], [50, 60], [10, 40], [5, 15], [5, 10], [25, 55]]
Output
[null, 1, 1, 2, 3, 3, 3]

Explanation
MyCalendarThree myCalendarThree = new MyCalendarThree();
myCalendarThree.book(10, 20); // return 1
myCalendarThree.book(50, 60); // return 1
myCalendarThree.book(10, 40); // return 2
myCalendarThree.book(5, 15); // return 3
myCalendarThree.book(5, 10); // return 3
myCalendarThree.book(25, 55); // return 3
```

Constraints:

- 0 <= startTime < endTime <= 10^9
- At most `400` calls will be made to `book`.