1751. Maximum Number of Events That Can Be Attended II

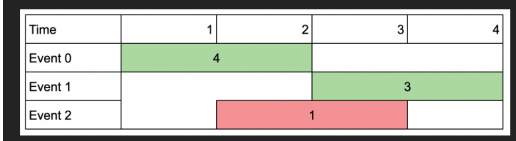
Hard | Topics | Companies | Hint

You are given an array of `events` where `events[i] = [startDay_i, endDay_i, value_i]`. The `i^th` event starts at `startDay_i` and ends at `endDay_i`, and if you attend this event, you will receive a value of `value_i`. You are also given an integer `k` which represents the maximum number of events you can attend.

You can only attend one event at a time. If you choose to attend an event, you must attend the **entire** event. Note that the end day is **inclusive**: that is, you cannot attend two events where one of them starts and the other ends on the same day.

Return *the **maximum sum** of values that you can receive by attending events.*

Example 1:




```
Input: events = [[1,2,4],[3,4,3],[2,3,1]], k = 2
Output: 7
Explanation: Choose the green events, 0 and 1 (0-indexed) for a total value of
4 + 3 = 7.
```

- prioritize value or attendence ? Doesn't matter

- dp for sure. → if dp, then can't greedy

dp :

option 1 : skip the current event

2 : join, find next non-overlaping event .

```python
class Solution:
    def maxValue(self, events: List[List[int]], k: int) -> int:
        events.sort()
        n = len(events)
        dp = [[-1] * n for _ in range(k+1)]
        starts = [s for s, e, v in events ]  — for bisect function

        def dfs(curr, k):
            if k == 0 or curr == n:
                return 0

            if dp[k][curr] != -1:
                return dp[k][curr]

            next_ind = bisect.bisect(starts, events[curr][1])
            dp[k][curr] = max(dfs(curr+1, k), dfs(next_ind, k-1) + events[curr][2])
            return dp[k][curr]

        return dfs(0, k)
```

skip attend