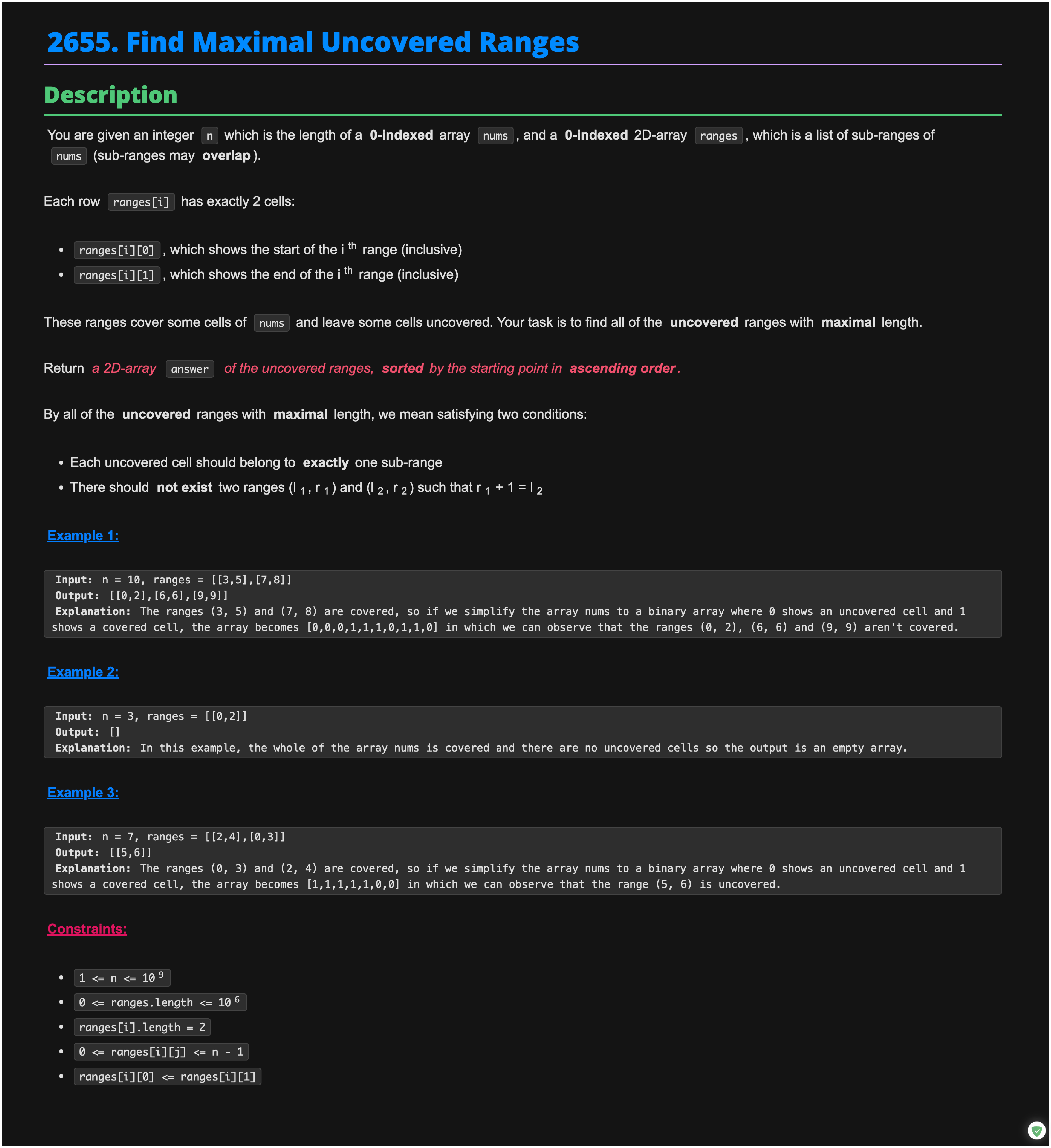

2655. Find Maximal Uncovered Ranges

Description

You are given an integer `n` which is the length of a **0-indexed** array `nums`, and a **0-indexed** 2D-array `ranges`, which is a list of sub-ranges of `nums` (sub-ranges may **overlap**).

Each row `ranges[i]` has exactly 2 cells:

- `ranges[i][0]`, which shows the start of the i^{th} range (inclusive)
- `ranges[i][1]`, which shows the end of the i^{th} range (inclusive)

These ranges cover some cells of `nums` and leave some cells uncovered. Your task is to find all of the **uncovered** ranges with **maximal** length.

Return *a 2D-array* `answer` *of the uncovered ranges,* ***sorted*** *by the starting point in* ***ascending order****.*

By all of the **uncovered** ranges with **maximal** length, we mean satisfying two conditions:

- Each uncovered cell should belong to **exactly** one sub-range
- There should **not exist** two ranges (l_1, r_1) and (l_2, r_2) such that $r_1 + 1 = l_2$

Example 1:

```
Input: n = 10, ranges = [[3,5],[7,8]]
Output: [[0,2],[6,6],[9,9]]
Explanation: The ranges (3, 5) and (7, 8) are covered, so if we simplify the array nums to a binary array where 0 shows an uncovered cell and 1 shows a covered cell, the array becomes [0,0,0,1,1,1,0,1,1,0] in which we can observe that the ranges (0, 2), (6, 6) and (9, 9) aren't covered.
```

Example 2:

```
Input: n = 3, ranges = [[0,2]]
Output: []
Explanation: In this example, the whole of the array nums is covered and there are no uncovered cells so the output is an empty array.
```

Example 3:

```
Input: n = 7, ranges = [[2,4],[0,3]]
Output: [[5,6]]
Explanation: The ranges (0, 3) and (2, 4) are covered, so if we simplify the array nums to a binary array where 0 shows an uncovered cell and 1 shows a covered cell, the array becomes [1,1,1,1,1,0,0] in which we can observe that the range (5, 6) is uncovered.
```

Constraints:

- `1 <= n <= 10`^{9}
- `0 <= ranges.length <= 10`^{6}
- `ranges[i].length = 2`
- `0 <= ranges[i][j] <= n - 1`
- `ranges[i][0] <= ranges[i][1]`