Problem 2406: Divide Intervals Into Minimum Number of Groups

Problem Information

Difficulty: Medium
Acceptance Rate: 63.57%
Paid Only: No
Tags: Array, Two Pointers, Greedy, Sorting, Heap (Priority Queue), Prefix Sum

Problem Description

You are given a 2D integer array `intervals` where `intervals[i] = [lefti, righti]` represents the **inclusive** interval `[lefti, righti]`.

You have to divide the intervals into one or more **groups** such that each interval is in **exactly** one group, and no two intervals that are in the same group **intersect** each other.

Return _the**minimum** number of groups you need to make_.

Two intervals **intersect** if there is at least one common number between them. For example, the intervals `[1, 5]` and `[5, 8]` intersect.

Example 1:

Input: intervals = [[5,10],[6,8],[1,5],[2,3],[1,10]] **Output:** 3 **Explanation:** We can divide the intervals into the following groups: - Group 1: [1, 5], [6, 8]. - Group 2: [2, 3], [5, 10]. - Group 3: [1, 10]. It can be proven that it is not possible to divide the intervals into fewer than 3 groups.

Example 2:

Input: intervals = [[1,3],[5,6],[8,10],[11,13]] **Output:** 1 **Explanation:** None of the intervals overlap, so we can put all of them in one group.

Constraints:

* `1 <= intervals.length <= 105` * `intervals[i].length == 2` * `1 <= lefti <= righti <= 106`

Code Snippets

C++:

```
class Solution {
public:
int minGroups(vector<vector<int>>& intervals) {

}
};
```

Java:

```
class Solution {
public int minGroups(int[][] intervals) {

}
}
```

Python3:

```
class Solution:
def minGroups(self, intervals: List[List[int]]) -> int:
```