Problem 436: Find Right Interval

Problem Information

Difficulty: Medium
Acceptance Rate: 54.86%
Paid Only: No
Tags: Array, Binary Search, Sorting

Problem Description

You are given an array of `intervals`, where `intervals[i] = [starti, endi]` and each `starti` is **unique**.

The **right interval** for an interval `i` is an interval `j` such that `startj >= endi` and `startj` is **minimized**. Note that `i` may equal `j`.

Return _an array of**right interval** indices for each interval `i`_. If no **right interval** exists for interval `i`, then put `-1` at index `i`.

Example 1:

Input: intervals = [[1,2]] **Output:** [-1] **Explanation:** There is only one interval in the collection, so it outputs -1.

Example 2:

Input: intervals = [[3,4],[2,3],[1,2]] **Output:** [-1,0,1] **Explanation:** There is no right interval for [3,4]. The right interval for [2,3] is [3,4] since start0 = 3 is the smallest start that is >= end1 = 3. The right interval for [1,2] is [2,3] since start1 = 2 is the smallest start that is >= end2 = 2.

Example 3:

Input: intervals = [[1,4],[2,3],[3,4]] **Output:** [-1,2,-1] **Explanation:** There is no right interval for [1,4] and [3,4]. The right interval for [2,3] is [3,4] since start2 = 3 is the smallest start that is >= end1 = 3.

Constraints:

* `1 <= intervals.length <= 2 * 104` * `intervals[i].length == 2` * `-106 <= starti <= endi <= 106` * The start point of each interval is **unique**.

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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