Problem 1229: Meeting Scheduler

Problem Information

Difficulty: Medium
Acceptance Rate: 55.20%
Paid Only: Yes
Tags: Array, Two Pointers, Sorting

Problem Description

Given the availability time slots arrays `slots1` and `slots2` of two people and a meeting duration `duration`, return the **earliest time slot** that works for both of them and is of duration `duration`.

If there is no common time slot that satisfies the requirements, return an **empty array**.

The format of a time slot is an array of two elements `[start, end]` representing an inclusive time range from `start` to `end`.

It is guaranteed that no two availability slots of the same person intersect with each other. That is, for any two time slots `[start1, end1]` and `[start2, end2]` of the same person, either `start1 > end2` or `start2 > end1`.

Example 1:

Input: slots1 = [[10,50],[60,120],[140,210]], slots2 = [[0,15],[60,70]], duration = 8
Output: [60,68]

Example 2:

Input: slots1 = [[10,50],[60,120],[140,210]], slots2 = [[0,15],[60,70]], duration = 12
Output: []

Constraints:

* `1 <= slots1.length, slots2.length <= 104` * `slots1[i].length, slots2[i].length == 2` * `slots1[i][0] < slots1[i][1]` * `slots2[i][0] < slots2[i][1]` * `0 <= slots1[i][j], slots2[i][j] <= 109` * `1 <= duration <= 106`

Code Snippets

C++:

```
class Solution {
public:
vector<int> minAvailableDuration(vector<vector<int>>& slots1,
vector<vector<int>>& slots2, int duration) {

}
};
```

Java:

```
class Solution {
public List<Integer> minAvailableDuration(int[][] slots1, int[][] slots2, int
duration) {

}
}
```

Python3:

```
class Solution:
def minAvailableDuration(self, slots1: List[List[int]], slots2:
List[List[int]], duration: int) -> List[int]:
```