Problem Challenge 3

We'll cover the following

Employee Free Time (hard)

For 'K' employees, we are given a list of intervals representing the working hours of each employee. Our goal is to find out if there is a **free interval that is common to all employees**. You can assume that each list of employee working hours is sorted on the start time.

Example 1:

```
Input: Employee Working Hours=[[[1,3], [5,6]], [[2,3], [6,8]]]
Output: [3,5]
Explanation: Both the employees are free between [3,5].
```

Example 2:

```
Input: Employee Working Hours=[[[1,3], [9,12]], [[2,4]], [[6,8]]]
Output: [4,6], [8,9]
Explanation: All employees are free between [4,6] and [8,9].
```

Example 3:

```
Input: Employee Working Hours=[[[1,3]], [[2,4]], [[3,5], [7,9]]]
Output: [5,7]
Explanation: All employees are free between [5,7].
```

Try it yourself

Try solving this question here:

Java | Python3 | JS | C++

```
import java.util.*;

class Interval {
  int start;
  int end;

  public Interval(int start, int end) {
    this.start = start;
    this.end = end;
  }
};

class EmployeeFreeTime {

  public static List<Interval> findEmployeeFreeTime(List<List<Interval>> schedule) {
    List<Interval> result = new ArrayList<>();
    // TODO: Write your code here
    return result;
  }

  public static void main(String[] args) {

    List<List<Interval>> input = new ArrayList<>();
    input.add(new ArrayList<Interval>(Arrays.asList(new Interval(1, 3), new Interval(5, 6))));
    input.add(new ArrayList<Interval>(Arrays.asList(new Interval(2, 3), new Interval(6, 8))));
    List<Interval> result = EmployeeFreeTime.findEmployeeFreeTime(input);
    System.out.print("Free intervals: ");
    for (Interval interval : result)
```

Run | Save | Reset

Back

Solution Review: Problem Challenge 2

Next

Solution Review: Problem Challenge 3

Mark as Completed