759. Employee Free Time

DescriptionHintsSubmissionsDiscussSolution

- Difficulty:Hard
- Total Accepted:1.5K
- Total Submissions:3K
- Contributor:1337c0d3r
- Subscribe to see which companies asked this question.

Related Topics Greedy

We are given a list `schedule` of employees, which represents the working time for each employee.

Each employee has a list of non-overlapping `Intervals`, and these intervals are in sorted order.

Return the list of finite intervals representing **common, positive-length free time** for *all* employees, also in sorted order.

Example 1:

```
Input: schedule = [[[1,2],[5,6]],[[1,3]],[[4,10]]]
Output: [[3,4]]
Explanation:
There are a total of three employees, and all common
free time intervals would be [-inf, 1], [3, 4], [10, inf].
We discard any intervals that contain inf as they aren't finite.
```

Example 2:

```
Input: schedule = [[[1,3],[6,7]],[[2,4]],[[2,5],[9,12]]]
Output: [[5,6],[7,9]]
```

(Even though we are representing `Intervals` in the form `[x, y]`, the objects inside are `Intervals`, not lists or arrays. For example, `schedule[0][0].start = 1`, `schedule[0][0].end = 2`, and `schedule[0][0][0]` is not defined.)

Also, we wouldn't include intervals like [5, 5] in our answer, as they have zero length.

Note:

1. `schedule` and `schedule[i]` are lists with lengths in range `[1, 50]`.
2. `0 <= schedule[i].start < schedule[i].end <= 10^8`.

The idea is to flat the schedule into flat vector and sort them based on the start, then find the free steps, easy and have fun

```
struct Interval {
      int start;
```

```
	int end;
	Interval() : start(0), end(0) {}
	Interval(int s, int e) : start(s), end(e) {}
};


bool compare_tmpl(Interval &a, Interval &b)
{
	return a.start<b.start;
}

vector<Interval> employeeFreeTime(vector<vector<Interval>>& schedule) {
	//flat
	vector<Interval> schedules;
	for(int i=0;i<(int)schedule.size();++i)
	{
		for(auto elem:schedule[i])
		{
			schedules.push_back(elem);
		}
	}
	sort(schedules.begin(),schedules.end(),compare_tmpl);
	vector<Interval> res;
	int hi = schedules[0].end;
	for(int i=0;i<(int)schedules.size();i++)
	{
		Interval tmper = schedules[i];
		if(tmper.start>hi)
		{
			Interval c(hi,tmper.start);
			res.push_back(c);
			hi = tmper.end;
		}else if (tmper.end>hi)
		{
			hi = tmper.end;
		}
	}
	return res;
}
```