

## 56. Merge Intervals

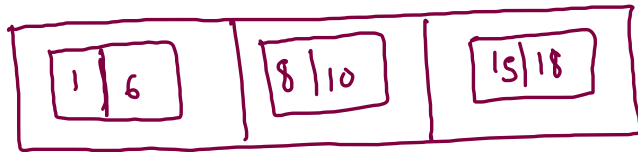
`[[1,3],[2,6],[8,10],[15,18]]`

1,3

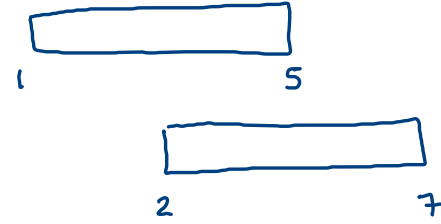
2,6

8,10

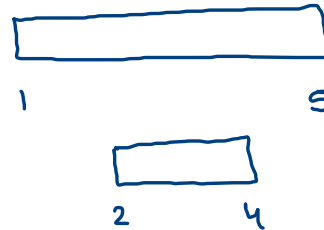
15,18



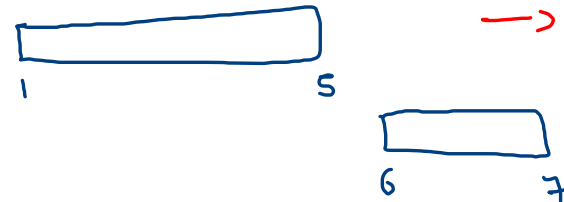
case 1



case 2



Case



→ no overlapping

```

public int[][] merge(int[][] intervals) {
    ArrayList<int[]> ans = new ArrayList<>();

    Arrays.sort(intervals, (a,b) -> {
        return a[0] - b[0];
    });

    ans.add(intervals[0]);

    int i = 1;

    while(i < intervals.length) {
        int li = ans.size()-1;
        if(intervals[i][0] <= ans.get(li)[1]) {
            ans.get(li)[1] = Math.max(ans.get(li)[1], intervals[i][1]);
        }
        else {
            ans.add(intervals[i]);
        }
        i++;
    }

    int[][] fres = new int[ans.size()][2];
    fres = ans.toArray(fres);

    return fres;
}

```

intervals

2d

1	3
2	6
8	10
15	18

i

1   6	8   10	15   18
-------	--------	---------

AL < [] >

li

2d

fres

1	6
8	10
15	18

## Minimum Platforms

```
arr[] = {0900, 0940, 0950, 1100, 1500, 1800}
dep[] = {0910, 1200, 1120, 1130, 1900, 2000}
```

Handwritten diagram showing the sequence of trains and their platform requirements:

$c = \cancel{0} \cancel{1} \cancel{0} \cancel{2} \cancel{2}$

Trains are represented by horizontal bars with their arrival and departure times written above and below them:

- Train 1: 0900 - 0910
- Train 2: 0940 - 1200
- Train 3: 0950 - 1120
- Train 4: 1100 - 1130
- Train 5: 1500 - 1900
- Train 6: 1800 - 2000

The diagram shows that at any given time, at most 3 trains are present, hence the minimum number of platforms required is 3.

Timeline of train arrivals and departures:

Time	Event
09:00	Train 1 arrives
09:10	Train 1 departs
09:40	Train 2 arrives
09:50	Train 3 arrives
11:00	Train 4 arrives
11:20	Train 3 departs
11:30	Train 4 departs
12:00	Train 2 departs
15:00	Train 5 arrives
18:00	Train 6 arrives
19:00	Train 5 departs
20:00	Train 6 departs

max no. of trains at any instant  
= min no. of platforms

int max = ~~0~~ ~~1~~ ~~2~~ 3

if (arr[i] ≤ dep[j]) {  
    c++; i++;

}  
else {

    c--; j++;

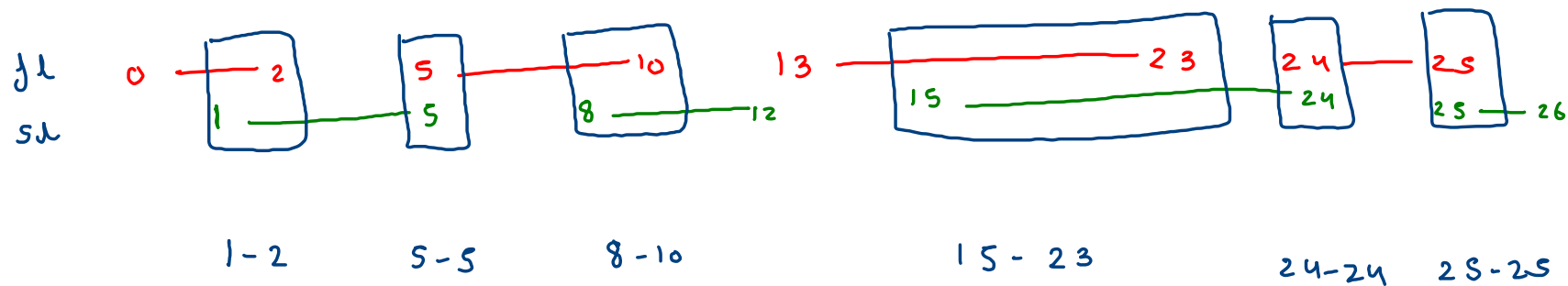
}

max = Math.max(max, c);

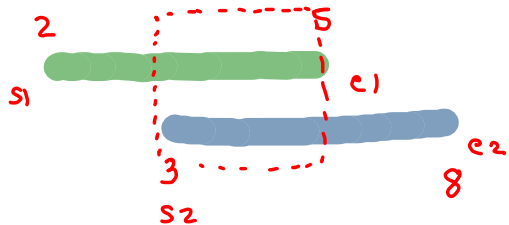
## 986. Interval List Intersections

Input: firstList = `[[0,2],[5,10],[13,23],[24,25]]`,

secondList = `[[1,5],[8,12],[15,24],[25,26]]`



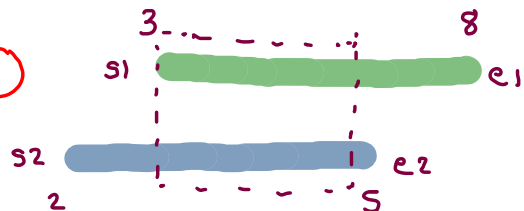
①



$$sp = \max(s_1, s_2)$$

$$ep = \min(e_1, e_2)$$

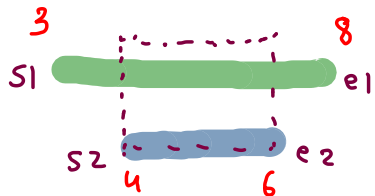
②



$$sp = 3$$

$$ep = 5$$

③



$$sp = 4$$

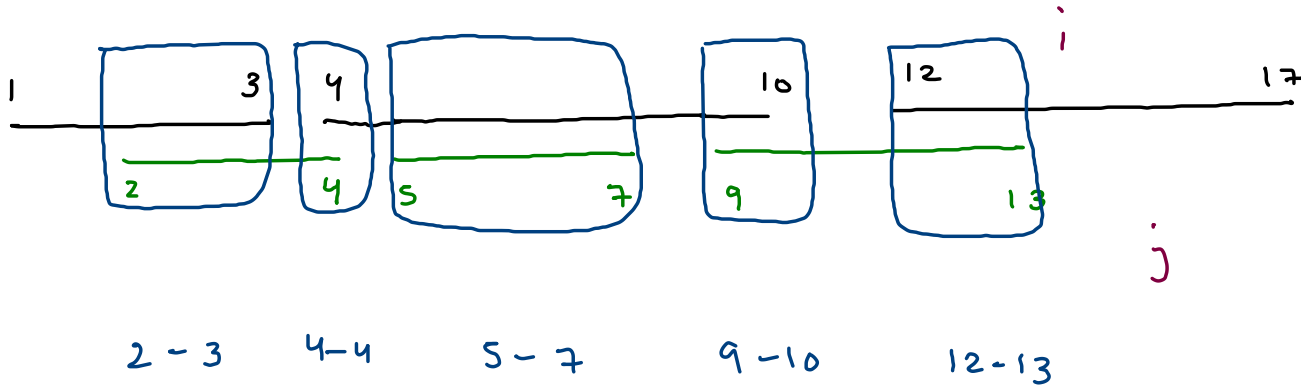
$$ep = 6$$

list\_1 : [[1,3] [5,10] [12,17]]

list\_2 : [[2,4] [5,7] [9,13]]

$$sp = \max(s_1, s_2)$$

$$ep = \min(e_1, e_2)$$



2,3

4,4

5,7

9,10

12,13