56. Merge Intervals

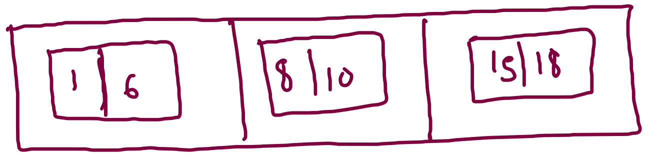
```
[[1,3],[2,6],[8,10],[15,18]]
```

1, 3

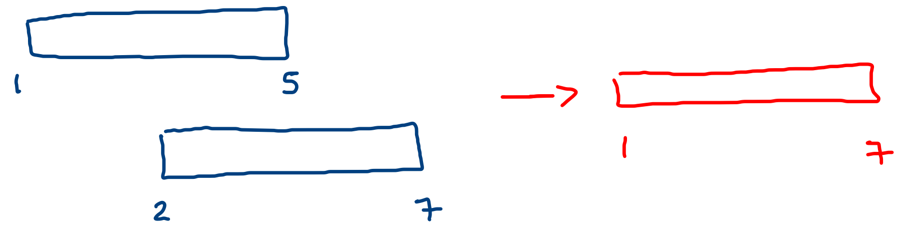
2, 6

8, 10

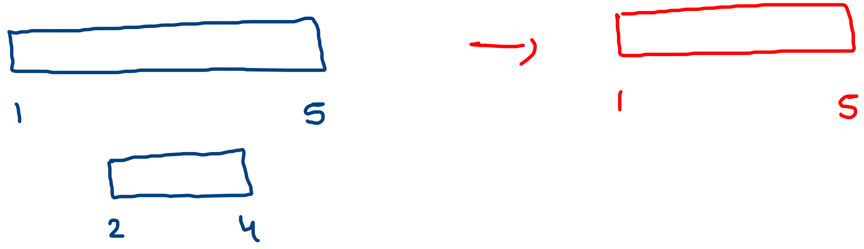
15, 18



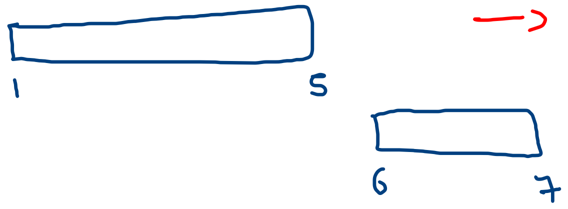
case 1



case 2

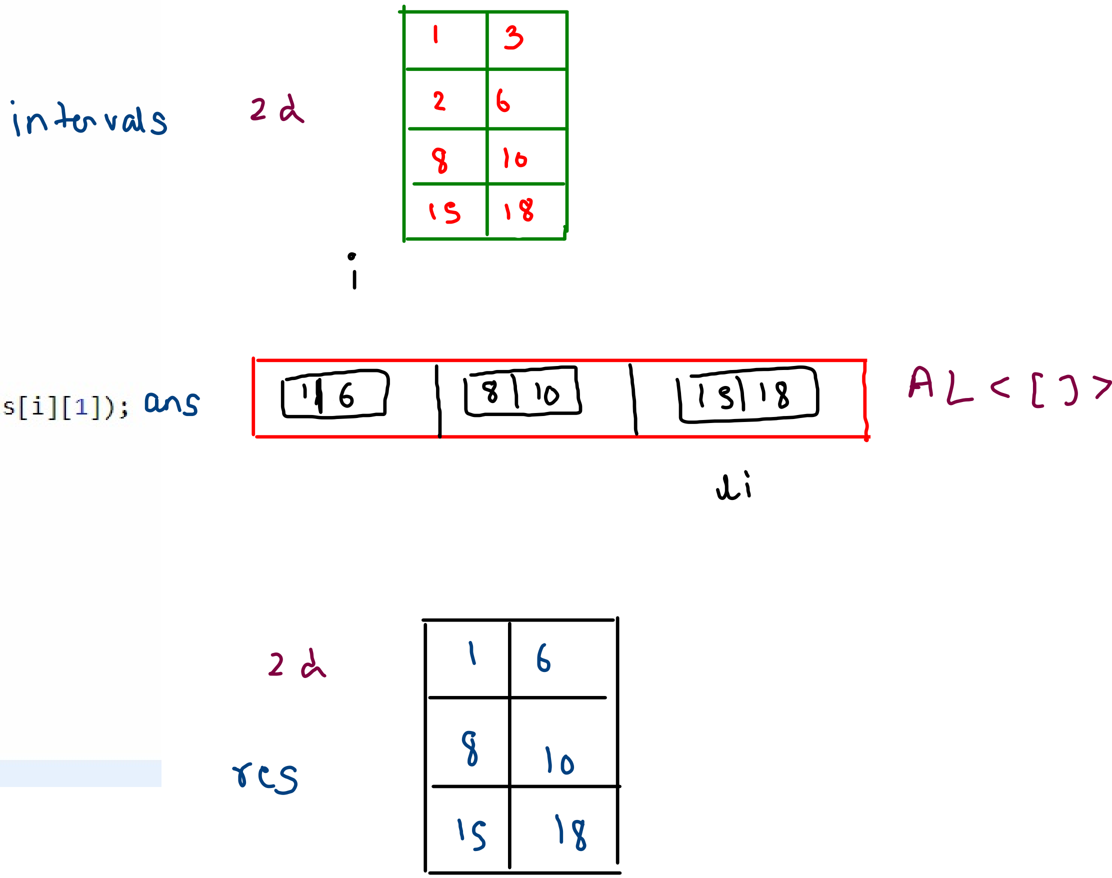


Case



→ no overlapping

```java
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;

}
```



Minimum Platforms

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

2 ~~1~~ ~~0~~

c = ~~0~~ ~~1~~ ~~0~~ ~~1~~ ~~1~~ ~~2~~

~~2~~ ~~3~~

j

9	9 : 40	9 : 50	11 : 00	15 : 00	18 : 00
9:10	11 : 20	11 : 30	12 : 00	19 : 00	20 : 00

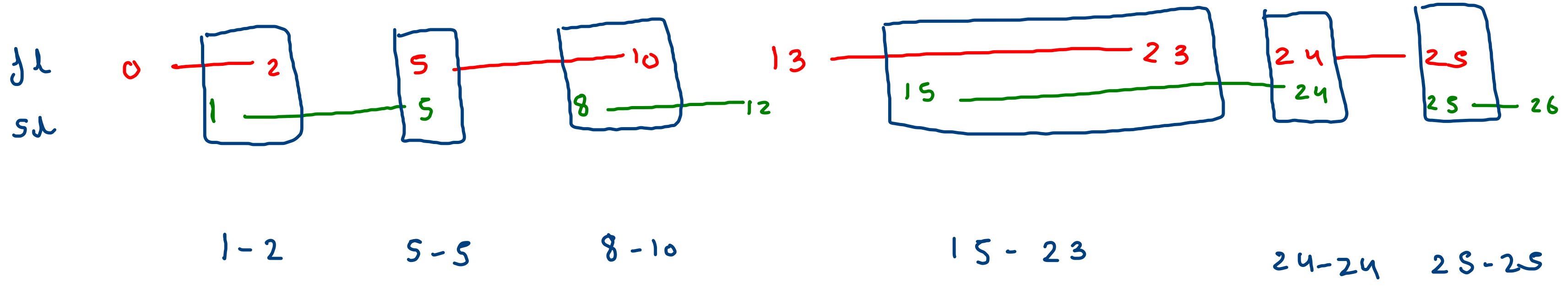
```
if ( arr[i] ≤ dep[j]) {
        c++; i++;
}
else {
        c--; j++;
}
max = Math.max(max, c);
```

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

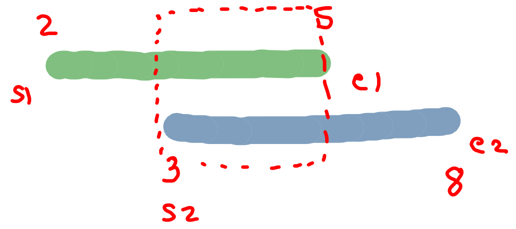
int max = ~~0~~ ~~1~~ ~~2~~ 3

986. Interval List Intersections

```
Input: firstList = [[0,2],[5,10],[13,23],[24,25]],
secondList = [[1,5],[8,12],[15,24],[25,26]]
```



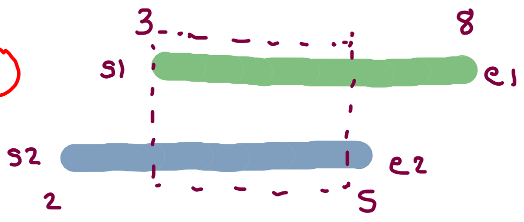
①



$sp = \max(s1, s2)$

$ep = \min(e1, e2)$

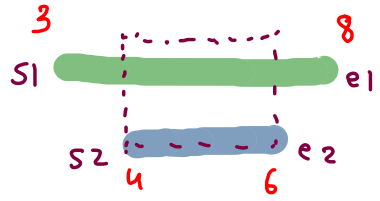
②



$sp = 3$

$ep = 5$

③



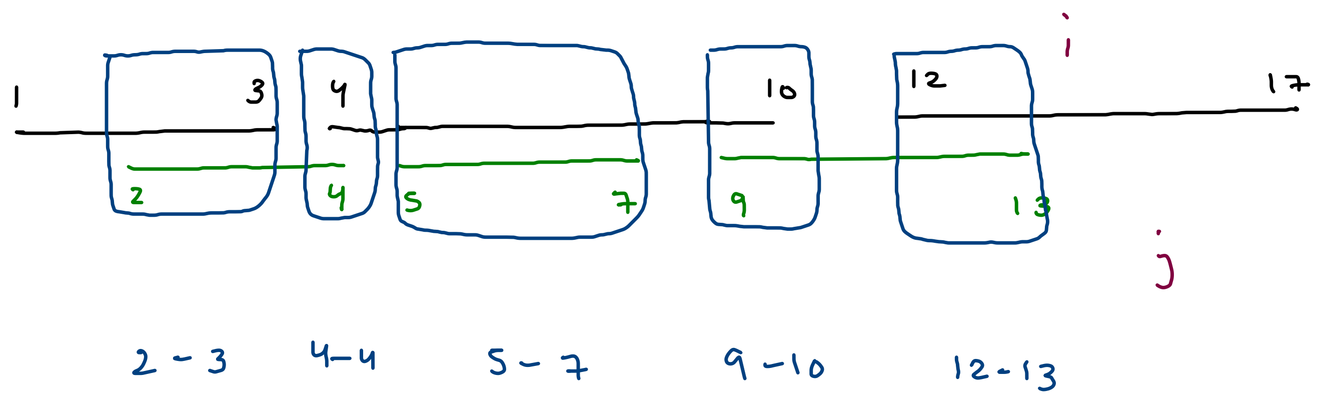
$sp = 4$

$ep = 6$

list_1 : [[1,3] [5,10] [12,17]]
list_2 : [[2,4] [5,7] [9,13]]

sp = max (s1, s2)

ep = min (e1, e2)



2, 3

4, 4

5, 7

9, 10

12, 13