Merge Overlapping Intervals

Given an array of intervals, merge all the overlapping intervals and return an array of non-overlapping intervals

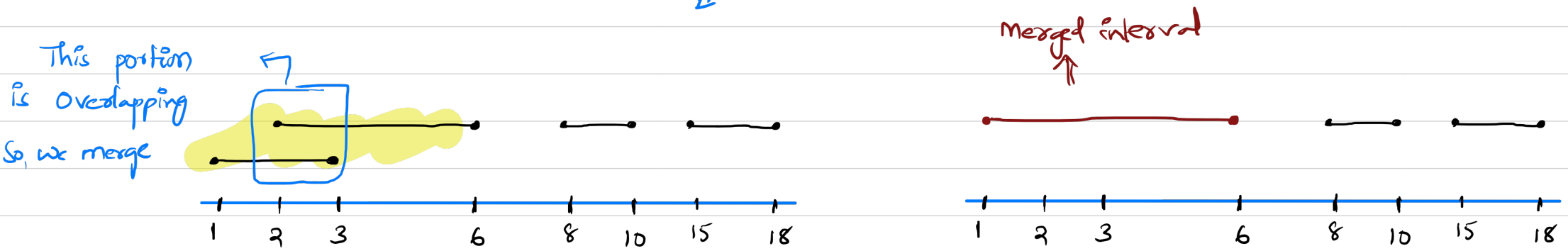
Ex:- $[[1,3],[2,6],[8,10],[15,18]]$

$\Downarrow$

output : $[[1,6],[8,10],[15,18]]$

First Sort array intervals

if you mark the given intervals



* We have to make sure that, in final ans, there won't be any Overlapping intervals

Intution

* Iterate from 0 → n

  ↳ as initially, Our ans list is empty, we add first element into it

* Observations

```
if(currentInterval[i][0] < lastInsertarray[1])
{ lastInsertedarray[1] = max(currentInterval[i][1], lastInsertedarray[1])
```

Case - I

If the current interval can be merged with the last inserted interval of the answer list :

Ex:- [[0,2] [1,4], [8,10]]

Comparator 1

ans list → update this

[0 2] comparator 2

Comparator 1 ⇒ Current Interval
comparator 2 → last Inserted Interval

1 is in range of this list
So update ans list[x][1] with max of Both Comparators

Updation → last inserted interval's end

maximum (current interval's end , last inserted interval's end)

Case - ii

If the current interval cannot be merged with last inserted interval of answer list

→ we insert the current interval in the answer list
and update last inserted interval as current interval

if(currentInterval[i][0] > lastInsertedIntervalarray[1])

To Sort 2D array (arr[][])

① Arrays.sort(arr, (a,b) → Integer.compare(a[0], b[0]))

② Arrays.sort(arr, new Comparator<Int[]>() {

```
public int compare(int a[], int b[])
{
  return a[0] - b[0];
}
});
```

Tc :- O(nlogn) + O(n)

Sc : O(n)

Dry Run

arr[][] ⇒ [[1,3],[2,6],[8,9],[10,13],[11,12],[14,15]]

i	currentIntervalStart	lastIntervalEnd	Check Merge	currentIntervalEnd	Update lastIntervalEnd
1	[2,6] → 2	[1,3] → 3	2<3 ✓	6	[1,6]
2	[8,9] → 8	[1,6] → 6	8>6 X	X	[1,9]
3	[10,13] → 10	[8,9] → 9	10>9 X	X	[10,13]
4	[11,12] → 11	[10,13] → 13	11<13 ✓	12	[10,13]
5	[14,15] → 14	[10,13] → 13	14>13 X	X	[14,15]