Question: <https://leetcode.com/problems/merge-intervals/>

* In order to merge all these intervals, we need to sort them in increasing order.
* So when can we merge?

If we observe carefully we will notice that we can merge only if the previous end point is greater than or equal to the present start point.

* So if merge is possible then update the end point; else add previously start and previously end to the resultant list and set new start and end.
* Once done with traversing add the left behind start and end.

Code:  
class Solution {

public int[][] merge(int[][] intervals) {

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

int i=1,start=intervals[0][0],end=intervals[0][1];

List<int[]> lt = new ArrayList<int[]>();

while(i<intervals.length){

int s=intervals[i][0], e=intervals[i][1];

if(s<=end){

end=Math.max(end, e);

}else{

lt.add(new int[]{start, end});

start=s;

end=e;

}

i++;

}

lt.add(new int[]{start, end});

int dp[][] = new int[lt.size()][2];

for(int k=0; k<lt.size(); k++){

dp[k]=lt.get(k);

}

return dp;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>