Merging Intervals

First, we sort the list and insert the first value, even if it is overlapping. Then, we start iterating over each interval.

1. If the current interval ENDS, before the next interval STARTS, it is not overlapping. In
   that case, we push this interval to the merged list.
2. If the next interval starts BEFORE the current interval ends, it is overlapping. We update the ending time of the current interval to whichever is the maximum among the current interval and the next interval's end time.

```
#include <bits/stdc++.h>
/*

    intervals[i][0] = start point of i'th interval
    intervals[i][1] = finish point of i'th interval

*/

vector<vector<int>> mergeIntervals(vector<vector<int>> &intervals)
{
    // Write your code here.
    vector<vector<int>> ans;
    sort(intervals.begin(),intervals.end());

    for(auto interval:intervals)
    {
        if(ans.empty()||ans.back()[1]<interval[0])
        {

            ans.push_back(interval);
        }
        else
        {
            ans.back()[1]=max(ans.back()[1],interval[1]);
        }
    }
    return ans;
}
```