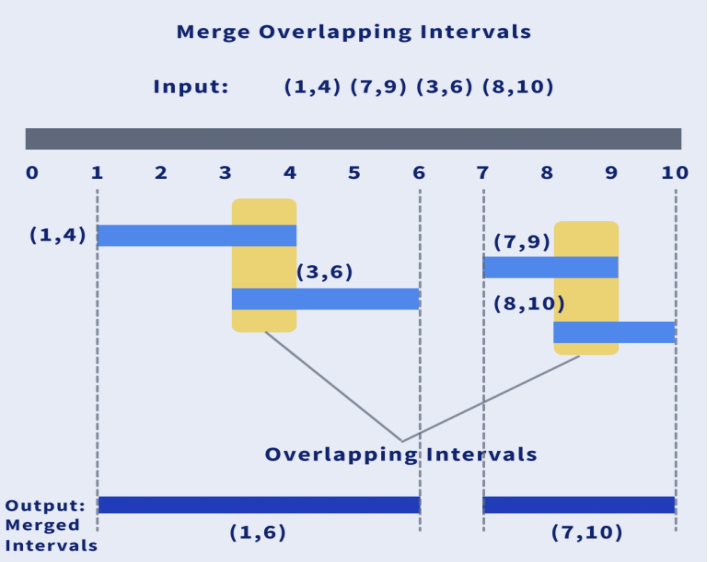
Merge Intervals

Difficulty: *Medium*

Question:

Given an array of intervals where intervals[i] = [$start_i$, end_i], merge all overlapping intervals, and return an array of the non-overlapping intervals that cover all the intervals in the input.



Example 1:

Input: intervals = [[1,3],[2,6],[8,10],[15,18]]

Output: [[1,6],[8,10],[15,18]]

Explanation: Since intervals [1,3] and [2,6] overlap, merge them into [1,6].

Example 2:

Input: intervals = [[1,4],[4,5]]

Output: [[1,5]]

Explanation: Intervals [1,4] and [4,5] are considered overlapping.

(Code given below! Try on your own before viewing the code)

Leetcode Question Link :

https://leetcode.com/problems/merge-intervals/

Solution : (C++)

```
#include<bits/stdc++.h>
using namespace std;
int main(){

        vector<vector<int>> intervals =
{{1,3},{2,6},{8,10},{15,18}};

        // Sort by starting time
        sort(intervals.begin(), intervals.end(), [](vector<int> a,
vector<int> b){
            return a[0] < b[0];
        });

        vector<vector<int>> ans;
        vector<int> temp = intervals[0];

        for(int i=1; i<intervals.size(); ++i){

            vector<int> pair = intervals[i];
            if(temp[1] >= pair[0]){
                temp[1] = max(temp[1], pair[1]);
            }
            else{
                ans.push_back(temp);
                temp = intervals[i];
            }
        }

        ans.push_back(temp);

        for(auto elem : ans){
            cout<<elem[0]<<" "<<elem[1]<<endl;
        }
}
```