

Merge overlapping Interval in C++

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <stack>
using namespace std;

// Structure to represent a pair of start and end times
struct Pair {
    int st;
    int et;

    Pair(int s, int e) {
        st = s;
        et = e;
    }
};

// Comparator function to sort pairs based on start
// time
bool comparePairs(const Pair& a, const Pair& b) {
    return a.st < b.st;
}

// Function to merge overlapping intervals and print
// in increasing order of start time
void mergeOverlappingIntervals(vector<Pair>&
intervals) {
    // Sort intervals based on start time
    sort(intervals.begin(), intervals.end(),
comparePairs);

    stack<Pair> st;
    st.push(intervals[0]);

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

        // If current interval overlaps with the top of the
        // stack, merge them
        if (intervals[i].st <= top.et) {
            top.et = max(top.et, intervals[i].et);
            st.pop();
            st.push(top);
        } else {
            st.push(intervals[i]);
        }
    }

    // Output the merged intervals in sorted order
    stack<Pair> result;
    while (!st.empty()) {
        result.push(st.top());
        st.pop();
    }

    while (!result.empty()) {
        Pair p = result.top();
        cout << p.st << " " << p.et << endl;
        result.pop();
    }
}
```

Input Intervals (Unsorted)

```
{22, 28}
{1, 8}
{25, 27}
{14, 19}
{27, 30}
{5, 12}
```

Step 1: Sort Intervals by Start Time

After sorting using comparePairs, the list becomes:

Index	Start	End
0	1	8
1	5	12
2	14	19
3	22	28
4	25	27
5	27	30

Step 2: Merge Overlapping Intervals using Stack

i	Current Interval	Top of Stack	Action	Stack Content
0	{1, 8}	-	Push first interval	[[1, 8]]
1	{5, 12}	{1, 8}	Overlaps, merge to {1, 12}	[[1, 12]]
2	{14, 19}	{1, 12}	No overlap, push	[[1, 12], {14, 19}]
3	{22, 28}	{14, 19}	No overlap, push	[[1, 12], {14, 19}, {22, 28}]
4	{25, 27}	{22, 28}	Overlaps, merge to {22, 28}	[[1, 12], {14, 19}, {22, 28}] (no change)
5	{27, 30}	{22, 28}	Overlaps, merge to {22, 30}	[[1, 12], {14, 19}, {22, 30}]


```

int main() {
    // Hardcoded input
    vector<Pair> intervals = {
        {22, 28},
        {1, 8},
        {25, 27},
        {14, 19},
        {27, 30},
        {5, 12}
    };

    // Calling the function to merge overlapping
    intervals
    mergeOverlappingIntervals(intervals);

    return 0;
}

```

 **Final Stack (top to bottom):**

```

{22, 30}
{14, 19}
{1, 12}

```

 **Step 3: Print Intervals in Sorted Order**

We reverse the stack to maintain start-time order:

```

1 12
14 19
22 30

```

 **Output:**

```

1 12
14 19
22 30

```

```

1 12
14 19
22 30

```