

Intervals Intersection (medium)

We'll cover the following



- Problem Statement
- Try it yourself
- Solution
- Code
 - Time complexity
 - Space complexity

Problem Statement

Given two lists of intervals, find the **intersection of these two lists**. Each list consists of **disjoint intervals sorted on their start time**.

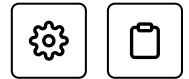
Example 1:

```
Input: arr1=[[1, 3], [5, 6], [7, 9]], arr2=[[2, 3], [5, 7]]  
Output: [2, 3], [5, 6], [7, 7]  
Explanation: The output list contains the common intervals between  
the two lists.
```

Example 2:

```
Input: arr1=[[1, 3], [5, 7], [9, 12]], arr2=[[5, 10]]  
Output: [5, 7], [9, 10]  
Explanation: The output list contains the common intervals between  
the two lists.
```

Try it yourself



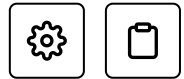
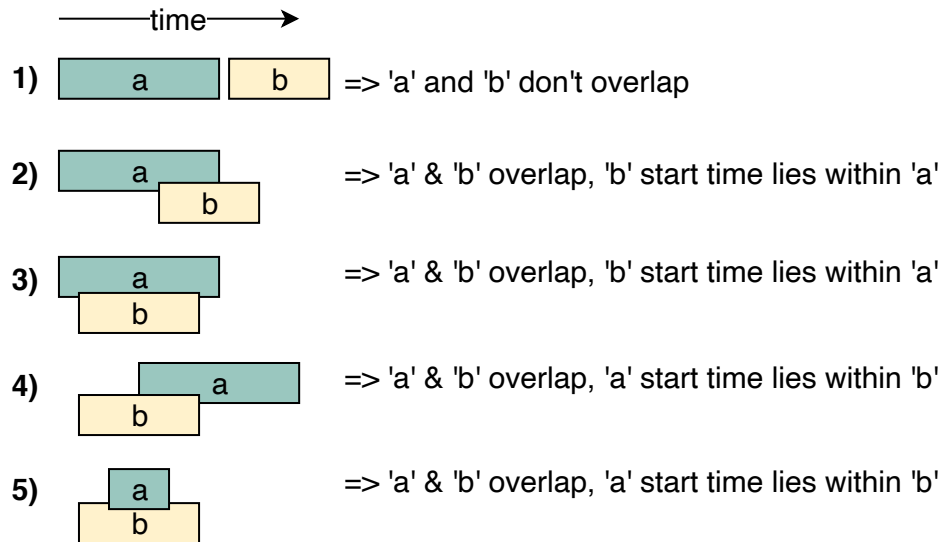
Try solving this question here:

Java	Python3	JS	C++
<pre>1 def merge(intervals_a, intervals_b): 2 result = [] 3 # TODO: Write your code here 4 return result 5 6 7 def main(): 8 print("Intervals Intersection: " + str(merge(9 print("Intervals Intersection: " + str(merge(10 11 12 main() 13</pre>			
<div> </div>			

Solution

This problem follows the Merge Intervals

(<https://www.educative.io/collection/page/5668639101419520/5671464854355968/5652017242439680/>) pattern. As we have discussed under Insert Interval (<https://www.educative.io/collection/page/5668639101419520/5671464854355968/5718314357620736/>), there are five overlapping possibilities between two intervals 'a' and 'b'. A close observation will tell us that whenever the two intervals overlap, one of the interval's start time lies within the other interval. This rule can help us identify if any two intervals overlap or not.



Now, if we have found that the two intervals overlap, how can we find the overlapped part?

Again from the above diagram, the overlapping interval will be equal to:

```
start = max(a.start, b.start)
end = min(a.end, b.end)
```

That is, the highest start time and the lowest end time will be the overlapping interval.

So our algorithm will be to iterate through both the lists together to see if any two intervals overlap. If two intervals overlap, we will insert the overlapped part into a result list and move on to the next interval which is finishing early.

Code

Here is what our algorithm will look like:

Java	Python3	C++	JS
<pre>1 def merge(intervals_a, intervals_b): 2 result = [] 3 i, j, start, end = 0, 0, 0, 1</pre>			



```
4
5 while i < len(intervals_a) and j < len(intervals_b):
6     # check if intervals overlap and intervals_a overlaps intervals_b
7     a_overlaps_b = intervals_a[i][start] >= intervals_b[j][start] and
8                     intervals_a[i][end] <= intervals_b[j][end]
9
10    # check if intervals overlap and intervals_b overlaps intervals_a
11    b_overlaps_a = intervals_b[j][start] >= intervals_a[i][start] and
12                  intervals_b[j][end] <= intervals_a[i][end]
13
14    # store the intersection part
15    if (a_overlaps_b or b_overlaps_a):
16        result.append([max(intervals_a[i][start], intervals_b[j][start]),
17                        min(intervals_a[i][end], intervals_b[j][end])])
18
19    # move next from the interval which is finished
20    if intervals_a[i][end] < intervals_b[j][end]:
21        i += 1
22    else:
23        j += 1
24
25    return result
26
27
28 def main():
```



Time complexity

As we are iterating through both the lists once, the time complexity of the above algorithm is $O(N + M)$, where 'N' and 'M' are the total number of intervals in the input arrays respectively.

Space complexity

Ignoring the space needed for the result list, the algorithm runs in constant space $O(1)$.

Insert Interval (medium)

Conflicting Appointments (medium)



Mark as Completed



Report
an Issue



Ask a Question

(https://discuss.educative.io/tag/intervals-intersection-medium__pattern-merge-intervals__grokking-the-coding-interview-patterns-for-coding-questions)