SCALER

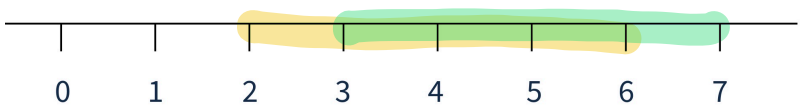
Interview Problems

TABLE OF CONTENTS

Merge Overlapping Intervals

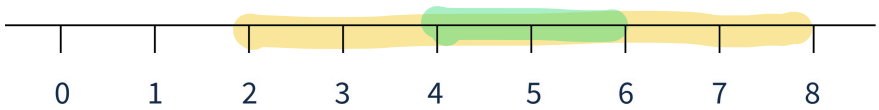
I_1 I_2

(2, 6) (3, 7) →



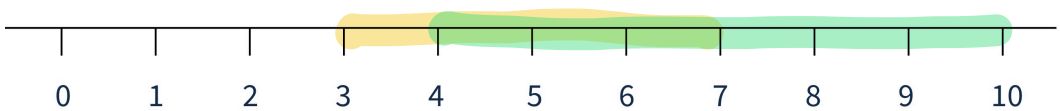
2 - 7

(2, 8) (4, 6) →



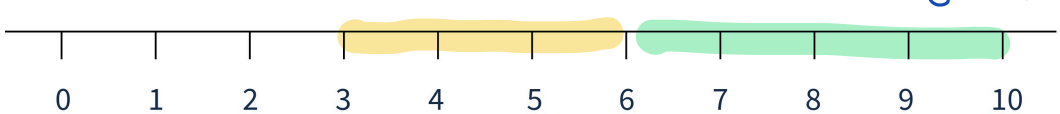
2 - 8

(3, 7) (4, 10) →



3 - 10

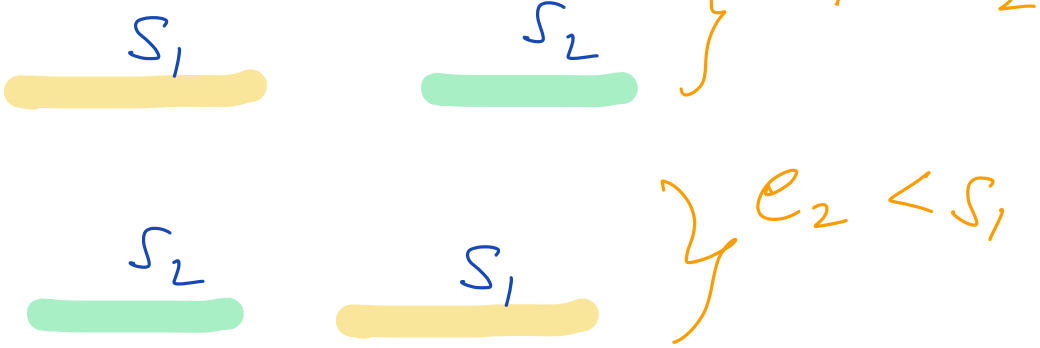
(3, 6) (6, 10) →



3 - 10

(2, 5) (8, 10) → no overlap

(5, 8) (1, 3) → no overlap



< / > Code

s_1, e_1 s_2, e_2

How to check no overlap

```
if ( e1 < s2 || e2 < s1 ) {
    no overlap
}
else {
    Smerged = min( s1, s2 )
    emerged = max( e1, e2 )
}
```

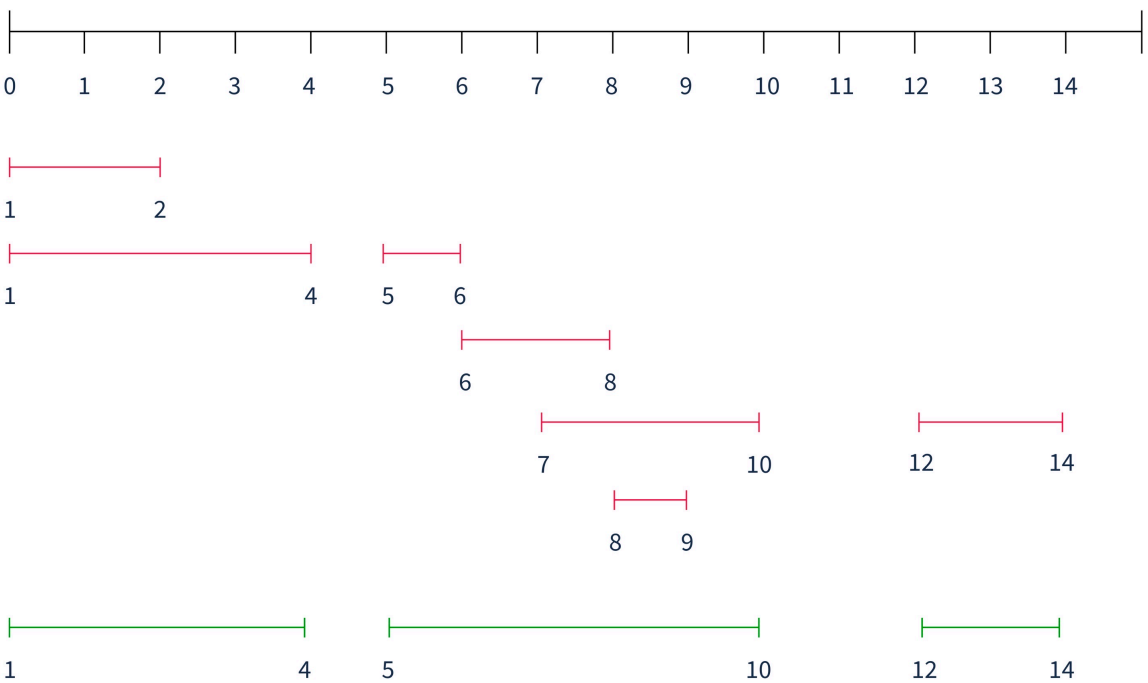
< *Question* > : Given a sorted list of overlapping intervals, sorted based on start-time, Merge all overlapping intervals and return the sorted list of non-overlapping intervals.

{[1,2], [1,4], [5,6], [6,8]} . $1 \le N \le 10^5$

N = 7

Intervals

1	2
1	4
5	6
6	8
7	10
8	9
12	14



1,4 5,10 12,14

Idea -1

Take the 0th interval as the reference initially. Now start iterating. If overlap, update the reference to the merged interval.
If no overlap, put the reference in answer list & update ref to new interval.

Example :

Intervals[N] → [(0, 2) , (1, 4) , (5, 6) , (6, 8) , (7, 10) , (8, 9) , (12, 14)]

< / > Code

```
Input is 2D array arr
cur_start = arr[0][0]
cur_end = arr[0][1]          TC: O(N)
ans = []
for(i: 1 → n-1){
    if( arr[i][0] > cur_end ){
        //no overlap
        ans.insert{ cur_start, cur_end }
        cur_start = arr[i][0]
        cur_end = arr[i][1]
    }
    else{
        cur_end = max(cur_end, arr[i][1]
    }
}
ans.insert{ cur_start, cur_end }
```

< *Question* > : Given N non-overlapping intervals sorted based on start-time. Given a new interval. Merge this with existing intervals, if possible and return final non-overlapping intervals.

N = 8 J = (10,22)

[(1 , 3) , (4 , 7) , (10 , 14) , (16 , 19) , (21 , 24) , (27 , 30) , (32 , 35) , (38 , 45)]

(10 , 22)

Resultant Intervals → 1,3 4,7 10,24 27,30 32,35 38,45

N = 5 12,22 I = (12,22)

Intervals( ) → [(1 , 5) , (8 , 10) , (11 , 14) , (15 , 20) , (21 , 24)]

I → (12 , 22)

Resultant Intervals → 1,5 8,10 11,14 15,20 21,24 27,30

1,5 8,10 11,24 27,30 32,44

~~12~~ 11, ~~22~~ 24

SCALER

input is 2D array all & 1 interval (I)

{11,22}

</> Code

```
ans = []
for ( i : 0 → n-1) {
    if ( all[i][1] < I[0])
        ans.insert(all[i])
    else if ( all[i][0] > I[1]) {
        ans.insert(I)

        for(j : i → n-1) {
            ans.insert(all[j])
        }
        break
    }
    else {
        I[0] = min(I[0], all[i][0])
        I[1] = max(I[1], all[i][1])
    }
}
ans.insert(I)
```

1, 2, 3, 4

< Question > : Find the first missing natural number. N = length of array

Example 1: arr [5] : [3 -1 1 2 7] ⇒ 4

Example 2: arr [7] : [9 2 6 4 -8 1 3] ⇒ 5

Example 3 : arr [6] : [1 0 -5 -6 4 2] ⇒ 3

Example 4 : arr [6] : [1 2 5 6 4 3] ⇒ 7

Example 5 : arr [4] : [1 2 3 4] ⇒ 5

BF Idea check for 1, 2, 3, 4, 5 .... till you find the answer. TC: $O(n^2)$

min possible $\Rightarrow$ 1
max possible $\Rightarrow$ N+1

Idea -2

0	1	2	3	4	5
8	1	4	2	6	3

num $\Rightarrow$ idx

$1 \Rightarrow 0$

$2 \Rightarrow 1$

$3 \Rightarrow 2$

$x \Rightarrow x-1$

Idea -3 (all given numbers are +ve)

0	1	2	3	4	5
-8	-1	-4	-2	6	-3

```
for( i : 0 → n-1 ){
    elem = abs( arr[i] )
    idx = elem - 1
    if ( idx >= 0 && idx < n ){
        arr[idx] *= -1
    }
}
```

Idea -4 Keep the elements at their correct position.

Example : arr [ ] → [4 2 7 6 9 1 5 3]

0 1 2 3 4 5 6 7

$-ve \Rightarrow n+2$

arr [ ] → [3 9 -2 5 -1 1]

0 1 2 3 4 5

$n = 6$

0 1 2 3 4 5

-3 9 -8 5 -8 1

ans = 2

0 1 2 3

-1 -2 -3 -4

< / > Code

```
for (i: 0 → n-1) {
    if (A[i] ≤ 0) {
        A[i] = N+2
    }
}

for (i: 0 → n-1) {
    elem = abs(arr[i])
    idx = elem - 1
    if (idx >= 0 && idx < n) {
        arr[idx] *= -1
    }
}

for (i: 0 → n-1) {
    if (A[i] > 0)
        return i+1
}

return N+1
```

0	1	2	3
-3	-1	-2	-4

0	1	2	3
21	22	23	24