

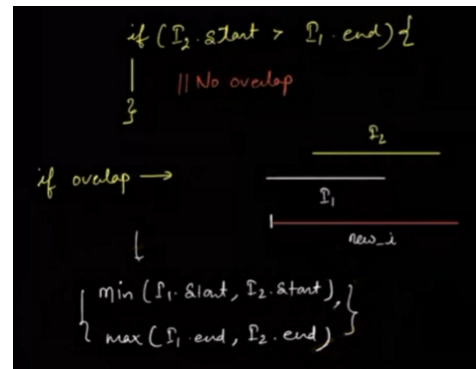
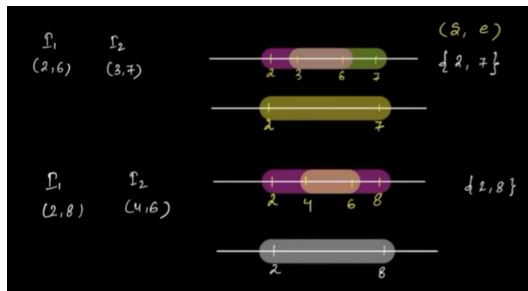
Agenda

16 June 2024 13:29

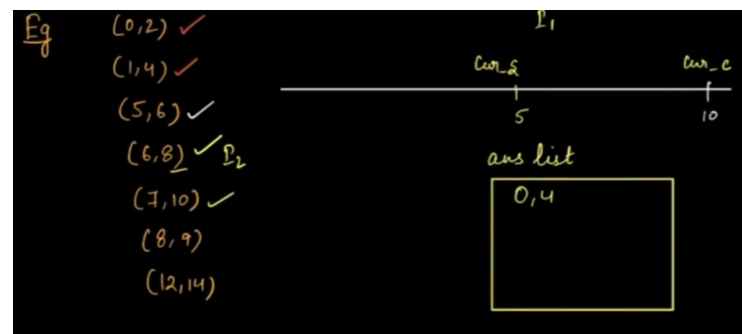
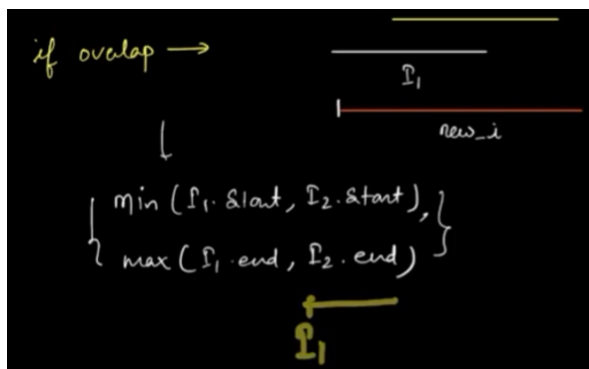
1. Merge Overlapping Intervals
2. Insert a new interval in a list of sorted non-overlapping intervals.
3. First missing positive integer in an array.

1. Merge Overlapping Intervals

16 June 2024 13:32



Q1) Given a sorted (based on start time) list of overlapping intervals. Merge All overlapping intervals and return sorted list.



```

List<Interval> ans;
int cur_s = A[0].start;
int cur_e = A[0].end;

for (int i = 1; i < N; i++) {
    if (A[i].start <= cur_e) {
        // overlap
        cur_e = max(cur_e, A[i].end);
    } else {
        Interval new_i = new Interval(cur_s, cur_e);
        ans.push(new_i);
        cur_s = A[i].start;
        cur_e = A[i].end;
    }
}

Interval new_i = new Interval(cur_s, cur_e);
ans.push(new_i);
return ans;
    
```

```

class Interval {
    int start;
    int end;
    Interval(int s, int e) {
        start = s;
        end = e;
    }
}
    
```

2D array

$A[N][2]$

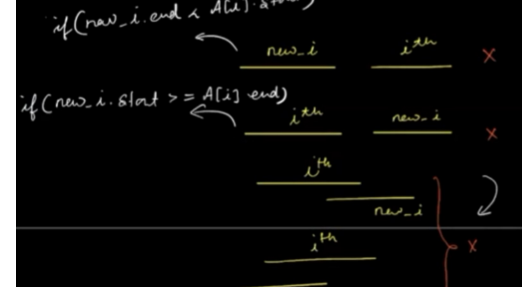
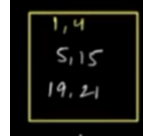
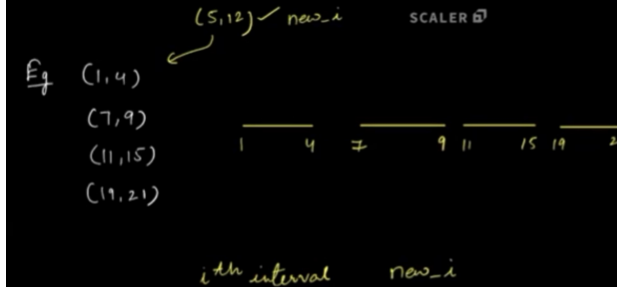
$A[i][0] = s, A[i][1] = e$

2. Sorted List of Non Overlapping Interval

26 June 2024 15:57

Question: Sorted List of Non Overlapping Interval sorted based on start time.

Insert a new interval such that the final list is sorted and non overlapping



```

for(int i=0; i < N; i++) {
    if(A[i].end < new_i.start) {
        ans.push(A[i]);
    }
    else if(new_i.end < A[i].start) {
        ans.push(new_i);
        while(i < N) {
            ans.push(A[i]);
            i++;
        }
        return ans;
    }
    else {
        new_i.start = min(new_i.start, A[i].start);
        new_i.end = max(new_i.end, A[i].end);
    }
}
ans.push(new_i);
return ans;

```

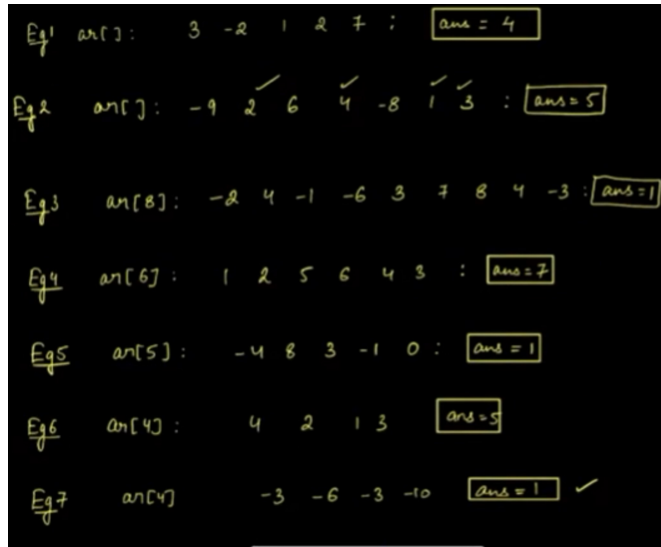
TC: $O(N)$
SC: $O(1)$

- Iterate from 0 to end.
- Check for non overlapping condition, if true add it to answer.
- Else If found the insert position add the interval in answer consecutively add all other left out intervals, as they are already non overlapping.
- Else overlapping perform **merging of overlapping intervals**.

3. Missing Integer

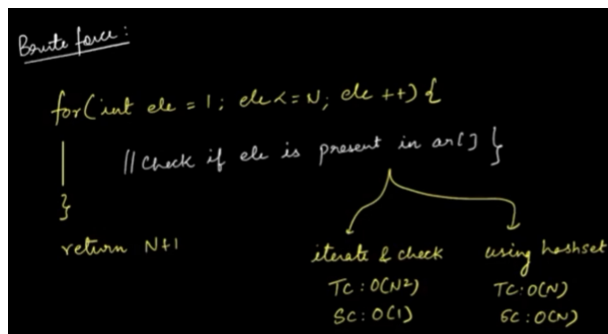
26 June 2024 16:31

Question: Given N array elements, find first missing positive integer.



Brute Force Approach:

- Start setting missing value as 1, iterate from 1 to N, and check for the missing values, every time found increment missing value else return it as the answer.



Optimised Approach:

Optimised 1:

```

bool ch[N]
for(int i=0; i<N; i++){
    int ele = ar[i]
    if (ele >= 1 && ele <= N) {
        ch[ele-1] = 1
    }
}

for(int i=0; i<N; i++){
    if (ch[i] == 0) {
        return i+1
    }
}

return N+1

```

Optimised 2:

Ans range: [1, 6]

ar[5]: 5 4 1 -2 3

↓

Replace ≤ 0 with +7

	0	1	2	3	4
ar[i]	5	4	1	-2	3
Replace	-5	4	-1	-7	-3
with +7	✓				

ans = 2

ele = abs(ar[i])

- Iterate and If element is less than = 0, update it as N+2
- Again iterate and if element is present at index-1 mark it as negative
- Again iterate and get first unmarked. Return it.

Step 1: Iterate & replace ≤ 0 with N+2

```

for(int i=0; i<N; i++){
    if (ar[i] <= 0) {
        ar[i] = N+2
    }
}

```

Step 2: Mark presence

```

for(int i=0; i<N; i++){
    int ele = abs(ar[i]);
    if (ele >= 1 && ele <= N) {
        ar[ele-1] = -1 * abs(ar[ele-1])
    }
}

```

Step 3: Get first unmarked

```

for(int i=0; i<N; i++){
    if (ar[i] > 0) {
        return i+1
    }
}

return N+1

```

TC: $O(N)$
SC: $O(1)$