

Day-2



Q1 Problem Name: Rotate Image by 90 degree.

1	2	3
4	5	6
7	8	9

 \Rightarrow

7	4	1
8	5	2
9	6	3

// Brute force:

// Take another matrix (dummy) and then take the first row of matrix and put it in the last col^m, second row \rightarrow 2nd col^m, and so on....

// TC: $O(N * N)$

// SC: $O(N * N)$

Remarks



// Second approach (Optimized approach)

// TC: $O(N^2)$

// SC: $O(1)$

```
for (int i = 0; i < n; i++) {
```

```
    for (int j = 0; j < i; j++) {
```

```
        swap(matrix[i][j], matrix[j][i])
```

```
    }
```

```
}
```

```
for (int i = 0; i < n; i++)
```

```
{
```

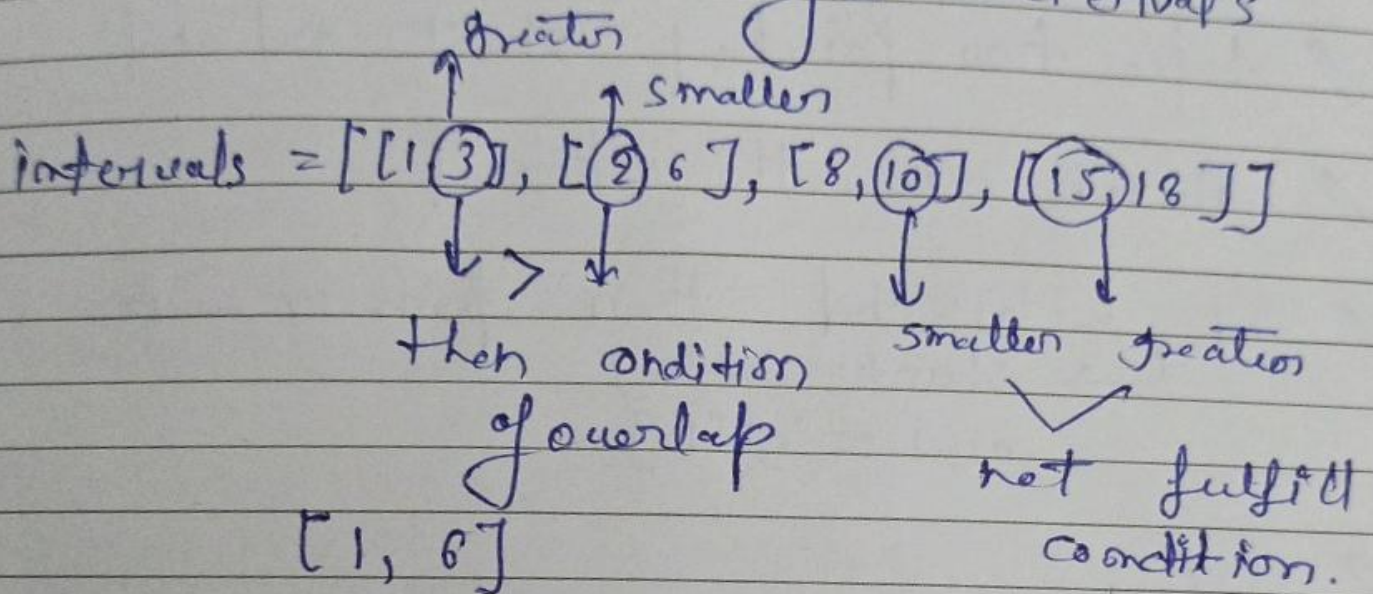
```
    reverse(matrix[i].begin(),  
            matrix[i].end());
```

```
}
```

Remarks

Ques 2

Problem Name : Merge Intervals



Brute force : $Tc (O(N \log N + O(N^2)))$
 $Sc : O(N)$

first check array is sorted or not

Now linearly iterate over the array and check for all next intervals whether they are overlapping or not.

Remarks

Take a new data structure and insert the overlapped interval.



optimal approach

TC:
 $O(N \log N + O(N))$
SG: $O(N)$

sort intervals

take two pointers start and end

iterate over the interval

if $int[0] > end$ then push in vector
 $\rightarrow start = int[0]$
 $end = int[1]$

else

$end = \max(end, int[1])$

push in resultant vector.

Remarks

Q3 Problem Name:

Merge Sorted Arrays



nums1 = [1, 2, 3, 0, 0, 0]

nums2 = [2, 5, 6]

n = 3, m = 3

// Brute force

// We can use a new array of size (n+m) and put all elements of arr1 and arr2 and sort it.

// Tc: $O(N \log N) + O(N) + O(N)$

// Sc: $O(N)$

// optional approach : Tc: $O(N \log N)$

// create a vector Sc: $O(1)$

Remarks

// By comparing nums1 and nums2
push the nums1 and nums2 acc to
condition in num vector



Problem Name : find duplicate in array of $N+1$ integers.

nums = [1, 3, 4, 2, 2]

output : 2

// using std \leftarrow approach (1)

// using Binary Search \leftarrow approach (2)

// using linked list cycle detection approach (3)

approach 1 : $TC \Rightarrow O(N \log N + N)$
 $SC \Rightarrow O(1)$

approach 2 : $TC \Rightarrow O(N)$
 $SC \Rightarrow O(1)$

approach 3 : $TC \Rightarrow O(N)$
 $SC \Rightarrow O(1)$

Remarks