Merge Sort

Merge 2 Sorted Array

Suppose ⟹ Extra space given already

A = [1,2,3,0,0,0] B = [2,5,6]

m = 3 (only 3 valid) n = 3

Merge 2 sorted form would be like

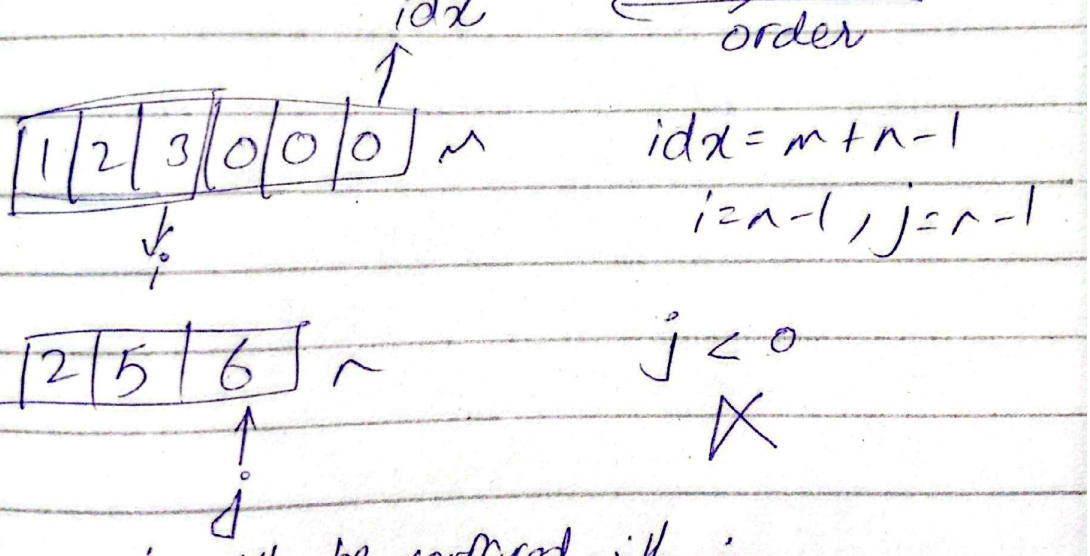
A = [1,2,3,2,5,6] but in sorted as well

A = [1,2,2,3,5,6] — (sorted)

A = [1,2,3] B = [2,5,6]

C = [1,2,2,3,5,6]

We will see it in the decreasing/reverse order

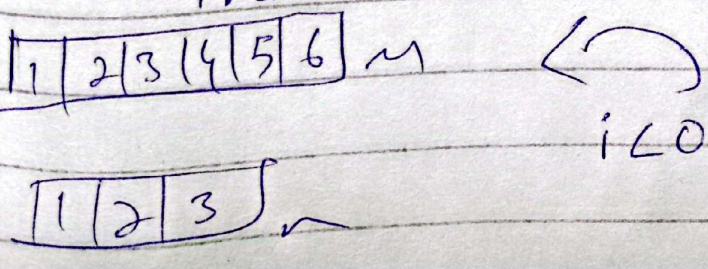


→ j will be compared with i

greater will go to idx position (idx--, j--)

→ Then j will be compared with i

Then i with idx —

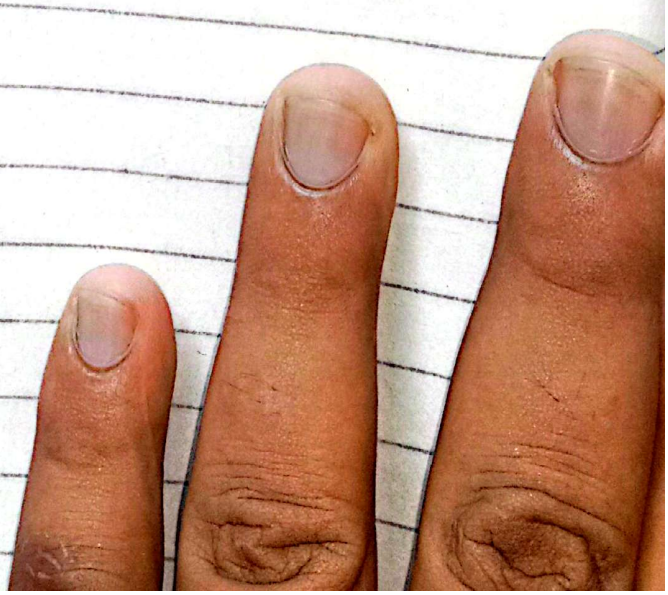


Pseudocode :-

```
idx = m+n-1 , i = m-1 , j = n-1
while( i>=0 && j>=0){
    if(A[i] >= A[j]){
        A[idx] = A[i]
        idx--
        i--
    }
    else
        A[idx] = B[j]
        idx--
        j--
}
while (j >= 0){
    A[idx] = B[j];
    idx--
    j--
}
```

T.C
O(m+n)

Additional loops if A gets empty but j/B remain full



Next Permutation

A = [1,2,3] Return lexicographically next.

1,2,3	3,1,2
1,3,2	3,2,1
2,1,3	
2,3,1	

↓ = greater

6 Possible permutations

Next permutation must be in-place of existing array.

→ Brute Force Approach

① finding all the permutations

② find lex next permutation of our array.

→ Optimal Approach O(1) → T.C

O(n) → S.C

Common observation

Pivot = A[i] < A[i+1]

1 2 3 5 4

1 2 3 6 5 4 (already largest)

2 5 4 3

In decreasing order 1 2 4 6 5 3 (to be smallest)

1 2 4 3 5 6 (have to reverse)

1 2 4 3 5

1 3 2 4 5

Pivot is the decreased element if we start to come backward.

① Find the pivot element.

② Find the right most element > pivot

③ swap(RME, Pivot)

④ Reverse (pivot+1) to n-1

1 2 5 4 3 (largest)

p

1 3 5 4 2 (have to make smallest) ← reverse them

p

1 3 2 4 5 (Answer)

Pseudocode

5,4,3,2,1 ← decreasing piv = -1 → Reverse it

```
piv = -1
for(i=n-2 ; i>=0 ; i--){
    if (A[i] < A[i+1]){
        piv = i
        break;
    }
}
if (piv == -1)
    reverse array
    return;                    // Edge case
reverse(A.begin(), A.end())

for(i=n-1, i>pivot; i--){
    A[i] > A[pivot]
        swap(A[i], A[pivot])
        break;
    }
}
i = piv+1
j = n-1
while(i<=j){
    swap(A[i], A[j])
    i++
    j--
}
```