**DSA Learnings**

* **In place algorithm**

an in-place algorithm is an algorithm which uses **no auxiliary data structure**. However, a small amount of extra storage space is allowed for auxiliary variables. The input is usually overwritten by the output as the algorithm executes.

Output = Modified input data structure

* **Q. Next\_Permutation**

My idea (AC) –

For all next permutations, whether char or numbers, it follows like this:

A, {all combinations of rest in lexicographic order} **then**

B, { “”}….

At the change point from A to B , Last sequence in A is the one where :

A, {rest all in descending order}

That’s the logic, if rest all are in descending order it marks the end of A series.

Travese from back and find the increasing order, the place where it violates, is the place which is at **end of its occurrence** in next terms.

4, 8, **7**, {9, 3, 2, 1}

Now 7 will end. In the next sequence, place of 7 will be taken by next greater number from {9, 3, 2, 1} i.e. **9**

4,8,**9,** {7, 3, 2, 1} X

In 9’s sequence at start it should be sorted.

4, 8, 9, {**1,2,3,7}**

Flow: **Traverse from back – Find the violating point – sort from violating point + 1 till end – replace the violating point by its NEXT greater number/char.**