Day 39 / 100 :

Topic - Backtracking, Array

1 Problem statement: Permutations (Medium)

Given an array nums of distinct integers, return all the possible permutations. You can return the answer in any order.

Example 1:
Input: nums = [1,2,3]
Output: [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]

Example 2:
Input: nums = [0,1]
Output: [[0,1],[1,0]]

Example 3:
Input: nums = [1]
Output: [[1]]

Solutions :

Approach 1 - backtracking + Array

The idea is much like the insertion sort. We start with only the first number, and pick next number from the rest of array, insert it to the front or back position of the first number, and pick the third number, insert it to front or mid or back position of the [first, second] array, and so on, until no element left.
For example, given array [1,2,3].
First, we have [1] as array to be inserted, and [2,3] remain;
next number is 2, we insert it to the front or back of [1], which become [1,2] and [2,1], with [3] remain.
Then we pick number 3, insert it to [1,2] and [2,1],
Insert to front position, we get
[3,1,2],
[3,2,1]
Insert to mid-position, we get
[1,3,2],
[2,3,1],
Insert to back, we get
[1,2,3],
[2,1,3],

and no element left in remain array, finish.

```
class Solution {
public:
    vector<vector<int>> permute(vector<int>& nums) {
        vector<vector<int>>res;
        vector<int>vec;
        DFS(nums, 0, vec, res);
        return res;
    }

    void DFS(vector<int>& nums, int pos, vector<int>& vec, vector<vector<int>>&
res){
        if(pos == nums.size()){
            res.push_back(vec);
            return;
        }
        for(int i = 0; i <= vec.size(); i++){
            vec.insert(vec.begin() + i, nums[pos]);
            DFS(nums, pos + 1, vec, res);
            vec.erase(vec.begin() + i);
        }
        return;
    }
};
```

2 Problem statement: multiply 2 number represented by linked List (Medium)

Given two numbers represented by linked lists, write a function that returns the multiplication of these two linked lists.

Examples:

Input : 9->4->6
8->4
Output : 79464

Input : 3->2->1
1->2
Output : 3852

Solutions :
Approach 1 - Traversing

Traverse both lists and generate the required numbers to be multiplied and then return the multiplied values of the two numbers.
Algorithm to generate the number from linked list representation:

1) Initialize a variable to zero
2) Start traversing the linked list
3) Add the value of first node to this variable
4) From the second node, multiply the variable by 10
   and also take modulus of this value by 10^9+7
   and then add the value of the node to this
   variable.
5) Repeat step 4 until we reach the last node of the list.

```
long long multiplyTwoLists (Node* first, Node* second)
{
    long long N= 1000000007;
    long long num1 = 0, num2 = 0;
    while (first || second){

        if(first){
            num1 = ((num1)*10)%N + first->data;
            first = first->next;
        }

        if(second)
        {
            num2 = ((num2)*10)%N + second->data;
            second = second->next;
        }

    }
    return ((num1%N)*(num2%N))%N;
}
```