#LeetCode 46

https://leetcode.com/problems/permutations/description/

Yifeng Zeng

#Description

46. Permutations

#Idea Report

Each time we need to select a number out from input add to a list, and recursively do the same thing until the list has all the element from the input. But how can we know which element is already selected or not? We can have an array of same length of the input, if we select one element already, we set that index to 1, so when we pick a number, we only picks ones from the index of 0s. And when the list has the same size of the input, meaning we selected all the elements from the input, we can just add this list to the final result.

Code

```
public class Solution {

    public List<List<Integer>> permute(int[] nums) {
        List<List<Integer>> res = new ArrayList<>();
        int[] visited = new int[nums.length];
        dfsHelper(res, new ArrayList<>(), nums, visited);
        return res;
    }

    private void dfsHelper(List<List<Integer>> res,
                           List<Integer> permutation,
                           int[] nums,
                           int[] visited) {
        if (permutation.size() == nums.length) {
            res.add(new ArrayList<>(permutation));
        }
        for (int i = 0; i < nums.length; i++) {
            if (visited[i] == 0) {
                permutation.add(nums[i]);
                visited[i] = 1;
                dfsHelper(res, permutation, nums, visited);
                permutation.remove(permutation.size() - 1);
                visited[i] = 0;
            }
        }
    }
}
```

#Summary

- This is the standard DFS template, need to be very familiar with it.
- Deep copy.