```
class Solution {
    public List<List<Integer>> subsets(int[] nums) {
        List<List<Integer>> res = new ArrayList<>();
        dfs(res, new ArrayList<>(), nums, 0);
        return res;
    }

    public void dfs(List<List<Integer>> res, ArrayList<Integer> tmp, int[] nums, int idx) {
        res.add(new ArrayList<>(tmp));
        for (int i = idx; i < nums.length; i++) {
            tmp.add(nums[i]);
            dfs(res, tmp, nums, i+1);
            tmp.remove(tmp.size()-1);
        }
    }
}
```

```
class Solution {
    public List<List<Integer>> subsets(int[] nums) {
        List<List<Integer>> result = new ArrayList<>();
        if (nums == null) {
            return result;
        }
        DFS(nums, 0, new ArrayList<>(), result);
        return result;
    }

    private void DFS(int[] nums, int level, List<Integer>
curr, List<List<Integer>> result) {
        if (level == nums.length) {
            result.add(new ArrayList<>(curr));
            return;
        }
        curr.add(nums[level]);
        DFS(nums, level + 1, curr, result);
        curr.remove(curr.size()-1);
        DFS(nums, level + 1, curr, result);
    }
```

```
}
```