package com.iimtiaz.day\_14;  
  
import java.util.ArrayList;  
import java.util.Arrays;  
import java.util.List;  
  
public class Main {  
 public static void main(String[] args) {  
 int[] nums = {1, 2, 3};  
 System.*out*.println(new Solution().subsets(nums));  
 System.*out*.println(new Solution\_2().subsets(nums));  
  
 }  
}  
  
*/\*\*  
 Time Complexity: O(n \* 2^n)  
 Nested Loops: The outer loop iterates n times, and for each iteration, the inner loop iterates over all existing  
 subsets in result.  
 Subset Size Growth: The number of subsets doubles in each iteration, leading to a total of 2^n subsets.  
 Copying and Appending: Creating new lists and appending elements within the loops take linear time proportional  
 to the subset size.  
 Space Complexity: O(n \* 2^n)  
 Result List: The result list holds up to 2^n subsets, each potentially containing up to n elements, resulting in  
 a space usage of n \* 2^n in the worst case.  
 Temporary List: The temp list also holds up to 2^n subsets in each iteration, contributing to space complexity.  
 \*/*class Solution {  
 public List<List<Integer>> subsets(int[] nums) {  
 List<List<Integer>> result = new ArrayList<>();  
 result.add(new ArrayList<Integer>());  
 for (int i = 0; i < nums.length; i++) {  
 List<List<Integer>> temp = new ArrayList<>();  
 for (List<Integer> el : result) {  
 List<Integer> newList = new ArrayList<>(el);  
 newList.add(nums[i]);  
 temp.add(newList);  
 }  
 result.addAll(temp);  
 }  
 return result;  
 }  
}  
  
class Solution\_2 {  
 public List<List<Integer>> subsets(int[] nums) {  
 List<List<Integer>> list = new ArrayList<>();  
 Arrays.*sort*(nums);  
 backtrack(list, new ArrayList<>(), nums, 0);  
 return list;  
 }  
 private void backtrack(List<List<Integer>> list , List<Integer> tempList, int [] nums, int start){  
 list.add(new ArrayList<>(tempList));  
 for(int i = start; i < nums.length; i++){  
 tempList.add(nums[i]);  
 backtrack(list, tempList, nums, i + 1);  
 tempList.remove(tempList.size() - 1);  
 }  
 }  
}  
  
  
// https://leetcode.com/problems/subsets/