1. Subsets

Medium

Given a set of **distinct** integers, *nums*, return all possible subsets (the power set).

**Note:** The solution set must not contain duplicate subsets.

**Example:**

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

class Solution {  
public:  
 vector<vector<int>> subsets(vector<int>& nums) {  
 vector<vector<int>>ans;  
 vector<int>tmp;  
 for(int k = 0; k <= nums.size(); ++k){  
 bool flag[nums.size()] = {false};  
 tmp.clear();  
 dfs(ans, tmp, nums, 0, k, flag);  
 }  
 return ans;  
 }  
 void dfs(vector<vector<int>>&ans, vector<int>&tmp, vector<int>& nums, int curIndex, int k, bool flag[]){  
 if(tmp.size() == k){  
 ans.push\_back(tmp);  
 return;  
 }  
 for(int i = curIndex; i < nums.size(); ++i){  
 if(flag[i] == true)continue;  
 tmp.push\_back(nums[i]);  
 flag[i] = true;  
 dfs(ans, tmp, nums, i + 1, k, flag);  
 tmp.pop\_back();  
 flag[i] = false;  
 }  
 }  
};