

# 090. Subsets II

## 090 Subsets II

- Backtracking + Array

### Description

Given a collection of integers that might contain duplicates, *nums*, return all possible subsets (the power set).

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

For example,

If *nums* = [1,2,2], a solution is:

```
[
  [2],
  [1],
  [1,2,2],
  [2,2],
  [1,2],
  []
]
```

### 1. Thought line

### 2. Backtracking + Array

```
1 class Solution {
2 private:
3     void backtrackingSubsets(vector<int>& nums, int st, vector<vector<int>>& result, vector<int>& temp){
4         if (st>nums.size()-1) return;
5         for (int i = st; i<=nums.size()-1; ++i){
6             temp.push_back(nums[i]);
7             result.push_back(temp);
8             backtrackingSubsets(nums, i+1, result, temp);
9             temp.pop_back();
10            while(i+1<=nums.size()-1 && nums[i+1] == nums[i])
11                ++i;
12        }
13    }
14 }
15 public:
16 vector<vector<int>>> subsetsWithDup(vector<int>& nums) {
17     vector<vector<int>>> result = {{}};
18     vector<int> temp;
19     sort(nums.begin(), nums.end());
20     backtrackingSubsets(nums, 0, result, temp);
21     return result;
22 }
23 };
```