# 078. Subsets

## 078 Subsets

- **Backtracking**
- **Bit Manipulation**

### Description

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

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

For example,
If ***nums*** = `[1,2,3]` , a solution is:

```
[
  [3],
  [1],
  [2],
  [1,2,3],
  [1,3],
  [2,3],
  [1,2],
  []
]
```

### 1. Thought line

(1) When vector& nums.empty(), result should be [ [] ].

### 2. Backtracking

```
class Solution {
private:
    void backtrackingPowerSet(vector<vector<int>>& result, vector<int>& temp, int st, vector<int>& nums){
        // put push action here for corner case_(1)
        result.push_back(temp);
        if (st>nums.size()-1) return;
        for (int i = st; !nums.empty() && i<=nums.size()-1; ++i){
            temp.push_back(nums[i]);
            backtrackingPowerSet(result, temp, i+1, nums);
            temp.pop_back();
        }
    }
public:
    vector<vector<int>> subsets(vector<int>& nums) {
        vector<vector<int>> result;
        vector<int> temp;
        if (nums.empty()) return result;
        backtrackingPowerSet(result, temp, 0, nums);
        return result;
    }
};
```

### 3. Bit Manipulation