子集

给你一个整数数组 nums ， 数组中的元素 **互不相同** 。返回该数组所有可能的子集（幂集）。

解集 **不能** 包含重复的子集。你可以按 **任意顺序** 返回解集。

示例 1：

输入： nums = [1,2,3]

输出： [[],[1],[2],[1,2],[3],[1,3],[2,3],[1,2,3]]

示例 2：

输入： nums = [0]

输出： [[],[0]]

解题思路：

数组中的每一个数字都有选和不选两种状态，我们可以用0和1表示，0表示不选，1表示选择。如果数组的长度是n，那么子集的数量就是2^n。比如数组长度是3，就有8种可能，分别是

[0，0，0]

[0，0，1]

[0，1，0]

[0，1，1]

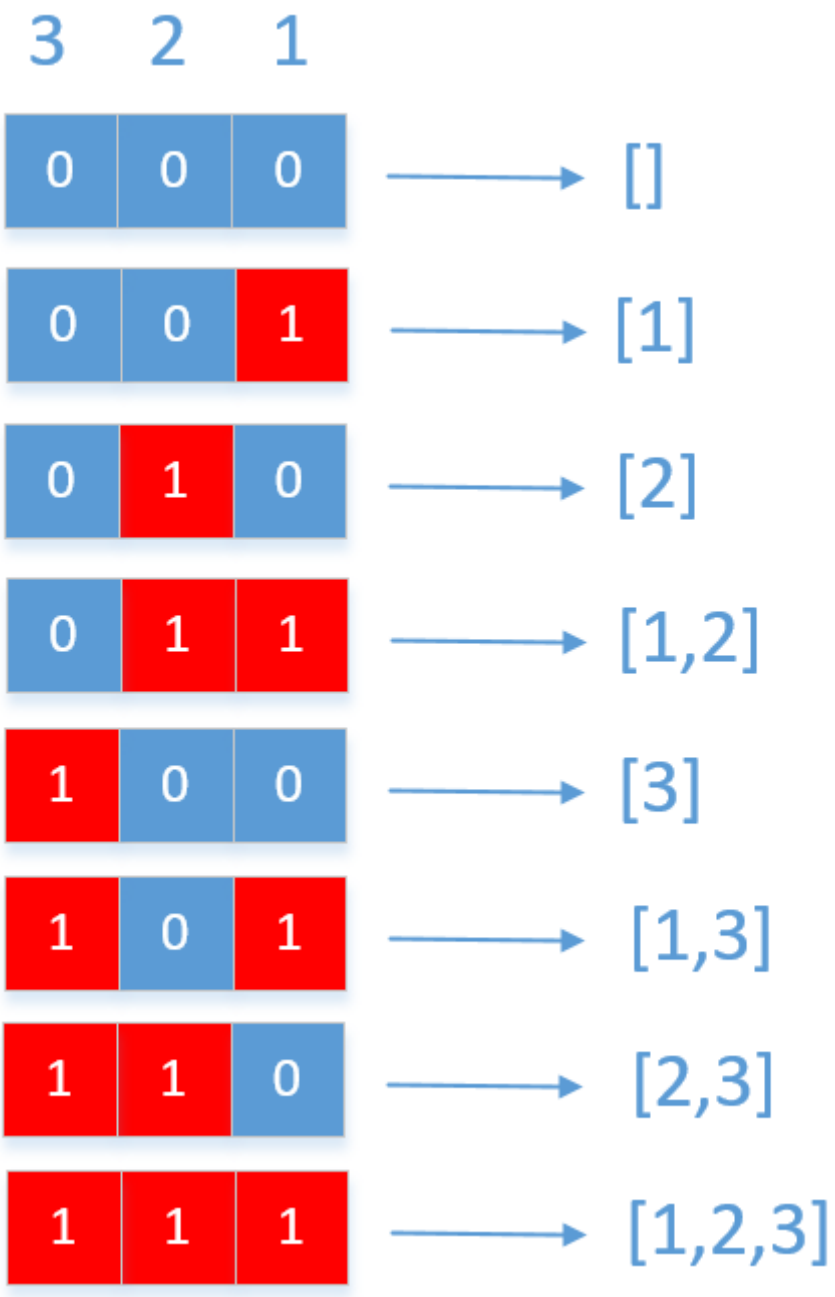
[1，0，0]

[1，0，1]

[1，1，0]

[1，1，1]

这里参照示例画个图来看下




```
class Solution {
public:
    vector<vector<int>> subsets(vector<int>& nums) {
        int n=1<<nums.size();//根据数据位数求出所有二进制序列  2^n
        int m=nums.size();//数据位数
        vector<vector<int>> res;//二维容器存储
        //依次遍历所有二进制序列  找到 1 对应位数字并存储在结果中
        for(int i=0;i<n;i++)
        {
            vector<int>tmp;
            for(int j=0;j<m;j++)
            {
```

```
                if((i>>j)&1==1)
                {
                    tmp.push_back(nums[j]);//找到二进制序列中对应为 1 位置的数字存储在 tmp 临时一维容器中
                }
            }
            res.push_back(tmp);//每一此二进制序列遍历结束后将 tmp 临时容器存储在最终结果中
        }
        return res;
    }
};
```