

LintCode领扣题解 (/problem) / 子集 · Subsets

子集 · Subsets

中文

Coupang (/problem/?tags=coupang)

Bloomberg (/problem/?tags=bloomberg)

脸书 (/problem/?tags=facebook)

亚马逊 (/problem/?tags=amazon)

优步 (/problem/?tags=uber)

递归 (/problem/?tags=recursion)

描述

给定一个含不同整数的集合，返回其所有的子集。

❗ 子集中的元素排列必须是非降序的，解集必须不包含重复的子集。

样例

样例 1:

```
输入: [0]
输出:
[
  [],
  [0]
]
```

样例 2:

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

挑战

你可以同时用递归与非递归的方式解决么？

在线评测地址: <https://www.lintcode.com/problem/subsets/> (<https://www.lintcode.com/problem/subsets/>)

收起题目描述 ^

语言类型

ALL (42)

python (21)

cpp (9)

java (8)

javascript (3)

golang (1)

上传题解



令狐冲

更新于 8/4/2020, 9:52:51 PM

使用组合类搜索的专用深度优先搜索算法。一层一层的决策每个数要不要放到最后的集合里。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    /**
     * @param nums: A set of numbers
     * @return: A list of lists
     */
    public List<List<Integer>> subsets(int[] nums) {
        List<List<Integer>> results = new ArrayList<>();
        Arrays.sort(nums);
        dfs(nums, 0, new ArrayList<Integer>(), results);
        return results;
    }

    // 1. 递归的定义
    // 以 subset 开头的, 配上 nums 以 index 开始的数所有组合放到 results 里
    private void dfs(int[] nums,
                    int index,
                    List<Integer> subset,
                    List<List<Integer>> results) {
        // 3. 递归的出口
        if (index == nums.length) {
            results.add(new ArrayList<Integer>(subset));
            return;
        }

        // 2. 递归的拆解
        // (如何进入下一层)

        // 选了 nums[index]
        subset.add(nums[index]);
        dfs(nums, index + 1, subset, results);

        // 不选 nums[index]
        subset.remove(subset.size() - 1);
        dfs(nums, index + 1, subset, results);
    }
}
```

👍 获赞 39

💬 14 条评论

你的口袋题库

2000+ 算法真题、国内外名企题库免费开放



九章算法APP



令狐冲

更新于 6/9/2020, 7:03:45 AM

使用比较通用的深度优先搜索方法

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
// 递归: 实现方式, 一种实现DFS算法的一种方式
class Solution {
    /**
     * @param S: A set of numbers.
     * @return: A list of lists. All valid subsets.
     */
    public List<List<Integer>> subsets(int[] nums) {
        List<List<Integer>> results = new ArrayList<>();

        if (nums == null) {
            return results;
        }

        if (nums.length == 0) {
            results.add(new ArrayList<Integer>());
            return results;
        }

        Arrays.sort(nums);
        helper(new ArrayList<Integer>(), nums, 0, results);
        return results;
    }

    // 递归三要素
    // 1. 递归的定义: 在 Nums 中找到所有以 subset 开头的的集合, 并放到 results
    private void helper(ArrayList<Integer> subset,
                        int[] nums,
                        int startIndex,
                        List<List<Integer>> results) {
        // 2. 递归的拆解
        // deep copy
        // results.add(subset);
        results.add(new ArrayList<Integer>(subset));

        for (int i = startIndex; i < nums.length; i++) {
            // [1] -> [1,2]
            subset.add(nums[i]);
            // 寻找所有以 [1,2] 开头的集合, 并扔到 results
            helper(subset, nums, i + 1, results);
            // [1,2] -> [1] 回溯
            subset.remove(subset.size() - 1);
        }

        // 3. 递归的出口
        // return;
    }
}
```

👍 获赞 33

💬 5 条评论



令狐冲

更新于 12/31/2020, 7:08:56 PM

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution:
    """
    @param nums: A set of numbers
    @return: A list of lists
    """
    def subsets(self, nums):
        nums = sorted(nums)
        combinations = []
        self.dfs(nums, 0, [], combinations)
        return combinations

    def dfs(self, nums, index, combination, combinations):
        combinations.append(list(combination))

        for i in range(index, len(nums)):
            combination.append(nums[i])
            self.dfs(nums, i + 1, combination, combinations)
            combination.pop()
```

👍 获赞 16

💬 12 条评论



令狐冲

更新于 6/9/2020, 7:03:45 AM

使用宽度优先搜索算法的做法 (BFS) 一层一层的找到所有的子集:

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
[]
[1] [2] [3]
[1, 2] [1, 3] [2, 3]
[1, 2, 3]
```

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    /**
     * @param nums: A set of numbers
     * @return: A list of lists
     */
    public List<List<Integer>> subsets(int[] nums) {
        if (nums == null) {
            return new ArrayList<>();
        }

        List<List<Integer>> queue = new ArrayList<>();
        int index = 0;

        Arrays.sort(nums);
        queue.add(new LinkedList<Integer>());
        while (index < queue.size()) {
            List<Integer> subset = queue.get(index++);
            for (int i = 0; i < nums.length; i++) {
                if (subset.size() != 0 && subset.get(subset.size() - 1) >= nums[i]) {
                    continue;
                }
                List<Integer> newSubset = new ArrayList<>(subset);
                newSubset.add(nums[i]);
                queue.add(newSubset);
            }
        }

        return queue;
    }
}
```

👍 获赞 16

💬 4 条评论



九章管理员

更新于 12/17/2020, 2:35:53 AM

非递归版本, 利用二进制的方式逐个枚举 subsets。

```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design，面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
// Non Recursion
class Solution {
    /**
     * @param S: A set of numbers.
     * @return: A list of lists. All valid subsets.
     */
    public List<List<Integer>> subsets(int[] nums) {
        List<List<Integer>> result = new ArrayList<List<Integer>>();
        int n = nums.length;
        Arrays.sort(nums);

        // 1 << n is 2^n
        // each subset equals to an binary integer between 0 .. 2^n - 1
        // 0 -> 000 -> []
        // 1 -> 001 -> [1]
        // 2 -> 010 -> [2]
        // ..
        // 7 -> 111 -> [1,2,3]
        for (int i = 0; i < (1 << n); i++) {
            List<Integer> subset = new ArrayList<Integer>();
            for (int j = 0; j < n; j++) {
                // check whether the jth digit in i's binary representation is 1
                if ((i & (1 << j)) != 0) {
                    subset.add(nums[j]);
                }
            }
            result.add(subset);
        }
        return result;
    }
}
```

👍 获赞 12

💬 4 条评论



九章-小原

更新于 12/28/2020, 11:22:44 PM

解题思路

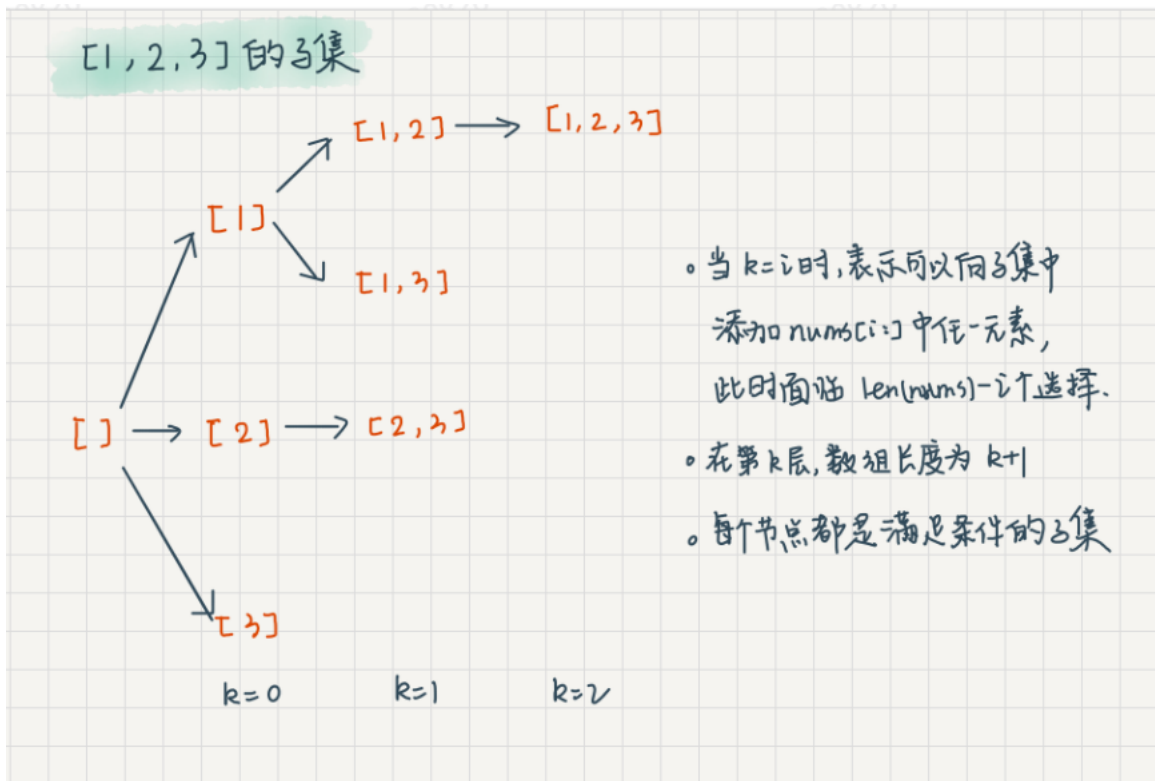
- 这道题我们需要使用dfs+回溯的方法来进行求解。

算法

- 将数组进行升序排列。
- 定义一个递归方法 dfs，参数有：当前子集 subset，当前子集长度 k，返回结果 res。将当前子集添加到 res 中。从 k 到 len(nums)-1 遍历索引 i。将 nums[i] 添加到当前子集 subset。进行下一层的递归搜索，继续向子集中添加元素，这时 k 要加一。* 从 subset 中删除 nums[i] 进行回溯。

举例分析

- 假设 nums = [1, 2, 3]，递归树如图所示。树每深一层，子集的长度就加一。每个节点都是满足条件的子集，需要记录到结果 res 中。



复杂度分析

- 时间复杂度: $O(n * 2^n)$, 其中 n 为 $nums$ 的长度。生成所有子集, 并复制到输出集合中。
- 空间复杂度: $O(n * 2^n)$, 其中 n 为 $nums$ 的长度。存储所有子集, 共 n 个元素, 每个元素都有可能存在或者不存在。

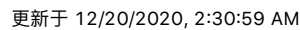
代码

python

java

c++

👍 获赞 4 💬 5 条评论





```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution:

    def search(self, nums, S, index):
        if index == len(nums):
            self.results.append(list(S))
            return

        S.append(nums[index])
        self.search(nums, S, index + 1)
        S.pop()
        self.search(nums, S, index + 1)

    def subsets(self, nums):
        self.results = []
        self.search(sorted(nums), [], 0)
        return self.results
```

 获赞 4 6 条评论

令狐冲

更新于 8/16/2020, 8:32:54 PM

第二种 BFS 算法



invitation/shi



```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    /**
     * @param nums: A set of numbers
     * @return: A list of lists
     */
    public List<List<Integer>> subsets(int[] nums) {
        if (nums == null) {
            return new ArrayList<>();
        }

        List<List<Integer>> queue = new ArrayList<>();
        queue.add(new LinkedList<Integer>());
        Arrays.sort(nums);

        for (int num : nums) {
            int size = queue.size();
            for (int i = 0; i < size; i++) {
                List<Integer> subset = new ArrayList<>(queue.get(i));
                subset.add(num);
                queue.add(subset);
            }
        }

        return queue;
    }
}
```

👍 获赞 2 💬 添加评论



令狐冲

更新于 7/15/2020, 2:23:31 AM

DFS 算法 增加一个数的时候, 用 `S + [nums[index]]` 往下传递, 这样虽然写起来简单, 但是效率会低一些, 因为每次都会创建一个新的数组。

```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
class Solution:

    def search(self, nums, S, index):
        if index == len(nums):
            self.results.append(S)
            return

        self.search(nums, S + [nums[index]], index + 1)
        self.search(nums, S, index + 1)

    def subsets(self, nums):
        self.results = []
        self.search(sorted(nums), [], 0)
        return self.results
```

👍 获赞 0

💬 2 条评论



令狐冲

更新于 6/9/2020, 7:04:28 AM

Given a set of distinct integers, S, return all possible subsets.

Note: Elements in a subset must be in non-descending order. The solution set must not contain duplicate subsets.

For example, If S = 1,2,3 (), a solution is:

[3 (), 1 (), 2 (), 1,2,3 (), 1,3 (), 2,3 (), 1,2 (), []]<div>
</div><div>详细题解，以及Bit位算法，请至九章微博：http://weibo.com/3948019741/BEDeKeB0v</div>

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
private:
    void helper(vector<vector<int> > &results,
                vector<int> &subset,
                vector<int> &nums,
                int start) {
        results.push_back(subset);

        for (int i = start; i < nums.size(); i++) {
            subset.push_back(nums[i]);
            helper(results, subset, nums, i + 1);
            subset.pop_back();
        }
    }

public:
    vector<vector<int> > subsets(vector<int> &nums) {
        vector<vector<int> > results;
        vector<int> subset;

        sort(nums.begin(), nums.end());
        helper(results, subset, nums, 0);

        return results;
    }
};

```

👍 获赞 0

💬 1 条评论

**Leon**

更新于 10/22/2020, 11:21:07 PM

提供4种python做法: 1. 使用DFS组合的思路, 对于每一位, 考虑是否使用, 是append再DFS, 否pop再DFS 2. 使用DFS排列的思路, 对于每一位, 从这一位往后进行DFS, DFS结束后要pop (回溯) 3. 使用BFS排列的思路, queue种的每一次subset再向后进行选择, 但是因为没有记录index, 所以用 subset-1 () < numsi ()来找index 4. 使用BFS组合的思路, 借助位运算, 来标记每一位是否被使用

总结: 1. DFS是需要回溯的 2. python的deep copy 如果是一重list 可用用: () 来进行, 再多就不行了 3. 排列类算法可以继续用在subsets II 中

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution:
    """
    @param nums: A set of numbers
    @return: A list of lists
    """
    def subsets(self, nums):
        # write your code here
        if nums==None:
            return []
        if len(nums) == 0:
            return [[]]
        res = []
        subset = []
        nums.sort()
        self.helper4(res,nums)
        return res

    def helper1(self,subset,res,index,nums):
        if index == len(nums):
            res.append(subset[:])
            return
        subset.append(nums[index])
        self.helper1(subset,res,index+1,nums)
        subset.pop(-1)
        self.helper1(subset,res,index+1,nums)

    def helper2(self,subset,res,index,nums):
        res.append(subset[:])
        for i in range(index,len(nums)):
            subset.append(nums[i])
            self.helper2(subset,res,i+1,nums)
            subset.pop(-1)

    def helper3(self,res,nums):
        q = []
        q.append([])
        while q:
            subset = q.pop()[:]
            res.append(subset)
            for i in range(len(nums)):
                if not subset or subset[-1] < nums[i]:
                    newSubset = subset[:]
                    newSubset.append(nums[i])
                    q.append(newSubset)

        return res

    def helper4(self,res,nums):
        n = len(nums)
        for i in range(1<=n):
            subset = []
            for j in range(i):
                if i & 1 <= j:
                    subset.append(nums[j])
            res.append(subset)
        return res
```

👍 获赞 22

💬 2 条评论

**九章用户PJGZHC**

更新于 6/9/2020, 7:03:47 AM

注意 `res.append(copy.deepcopy(subSet))` 如果这里不使用 `deep copy`, 就会导致 `res` 里的每一项都指向了 `subSet` 的 `reference`

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
import copy
class Solution:
    """
    @param nums: A set of numbers
    @return: A list of lists
    """
    def subsets(self, nums):
        # write your code here
        if nums is None:
            return []

        result = []
        nums.sort()
        self.dfs(nums, 0, [], result)
        return result

    def dfs(self, nums, start, subSet, res):
        res.append(copy.deepcopy(subSet))

        for i in xrange(start, len(nums)):
            subSet.append(nums[i]) #这里其实是将nums[i]的reference存入了subSet里, 如果第18行不使用deepcopy,
                                  #将会直接将nums[i]的reference再一次存储进res里
            self.dfs(nums, i + 1, subSet, res)
            subSet.pop()
```

👍 获赞 4

💬 2 条评论

[加载更多题解](#)

进阶课程

视频+互动	直播+互动	直播+互动	互动课
<div>九章算法班 2021 版</div> <div>8周时间精通 57 个核心高频考点，9招击破 FLAG、BATJ 算法面试。22....</div>	<div>系统架构设计 System Design 2021 版</div> <div>成为百万架构师必上。30 课时带你快速掌握18大系统架构设计知识点与面...</div>	<div>九章算法面试高频题冲刺班</div> <div>每期更新 15% 题目，考前押题，一举拿下FLAG & BATJ Offer</div>	<div>面向对象设计 OOD</div> <div>应届生及亚马逊面试必考，IT求职必备基础</div>