

Assignment #6: 回溯、树、双向链表和哈希表

Updated 1526 GMT+8 Mar 22, 2025

2025 spring, Compiled by 胡新璞, 工学院

1. 题目

LC46.全排列

backtracking, <https://leetcode.cn/problems/permutations/>

思路:

代码:

```
class Solution(object):
    def permute(self, nums):
        def perm(lst,k,ans):
            if k == len(lst):
                ans.append(lst.copy())
                return
            for i in range(k,len(lst)):
                lst[k],lst[i] = lst[i],lst[k]
                perm(lst,k + 1,ans)
                lst[k],lst[i] = lst[i],lst[k]
        ans = []
        perm(nums,0,ans)
        return ans
```

代码运行截图 （至少包含有"Accepted"）

通过 26 / 26 个通过的测试用例
提交于 2025.04.01 15:00

小红书特刊计划
掌握小红书面试高频考点

执行用时分布
0 ms | 击败 100.00%
复杂度分析

消耗内存分布
17.60 MB | 击败 82.55%

```
1 class Solution(object):
2     def permute(self, nums):
3         def perm(lst,k,ans):
4             if k == len(lst):
5                 ans.append(lst.copy())
6                 return
7             for i in range(k,len(lst)):
8                 lst[k],lst[i] = lst[i],lst[k]
9                 perm(lst,k + 1,ans)
10                lst[k],lst[i] = lst[i],lst[k]
11        ans = []
12        perm(nums,0,ans)
13        return ans
```

已存储 行 15.

测试用例
Case 1 Case 2 Case 3 +
nums =

LC79: 单词搜索

backtracking, <https://leetcode.cn/problems/word-search/>

思路: dfs

代码:

class Solution:

```
def exist(self, board: List[List[str]], word: str) -> bool:
    def dfs(a, b, cnt):
        directions = [[1, 0], [0, 1], [-1, 0], [0, -1]]
        if a < 0 or b < 0 or a >= len(board) or b >= len(board[0]):
            return False
        if board[a][b] != word[cnt]:
            return False
        if cnt == len(word) - 1:
            return True
        tmp = board[a][b]
        board[a][b] = ""
        for _ in range(len(directions)):
            nx = a + directions[_][0]
            ny = b + directions[_][1]
            if dfs(nx, ny, cnt + 1):
                board[a][b] = tmp
                return True
        board[a][b] = tmp
        return False

    for i in range(len(board)):
        for j in range(len(board[0])):
            if dfs(i, j, 0):
                return True
    return False
```

代码运行截图 （至少包含有"Accepted"）

通过 87 / 87 个通过的测试用例

提交于 2025.04.01 16:03

官方题解

写题解

🕒 执行用时分布

5430 ms | 击败 17.33%

📈 复杂度分析

📊 消耗内存分布

17.87 MB | 击败 6.34%

LC94.二叉树的中序遍历

dfs, <https://leetcode.cn/problems/binary-tree-inorder-traversal/>

思路:

代码:

class Solution:

```
def inorderTraversal(self, root: Optional[TreeNode]) -> List[int]:
```

```
    def dfs(root):
```

```
        if not root:
```

```
            return
```

```
        dfs(root.left)
```

```
        ans.append(root.val)
```

```
        dfs(root.right)
```

```
    ans = []
```


```
    dfs(root)
```

```
    return ans
```

代码运行截图 （至少包含有"Accepted"）

通过 71 / 71 个通过的测试用例

 ? 。 提交于 2025.04.01 16:49

 官方题解

 写题解

🕒 执行用时分布



0 ms | 击败 100.00% 🏆

💎 复杂度分析

🧠 消耗内存分布

17.48 MB | 击败 53.15% 🏆

LC102.二叉树的层序遍历

bfs, <https://leetcode.cn/problems/binary-tree-level-order-traversal/>

思路：

代码：

```
from collections import deque
```

```
class Solution:
```

```
    def levelOrder(self, root: Optional[TreeNode]) -> List[List[int]]:
```

```
        if not root:
```

```
            return []
```

```
        ans = []
```

```
        q = deque([root])
```

```
        while q:
```

```
            tmp = []
```

```
            for _ in range(len(q)):
```

```
                node = q.popleft()
```

```
                tmp.append(node.val)
```

```
                if node.left:
```

```
                    q.append(node.left)
```

```
                if node.right:
```

```
                    q.append(node.right)
```

```
            ans.append(tmp)
```

```
        return ans
```

代码运行截图 （至少包含有"Accepted"）

通过 35 / 35 个通过的测试用例
提交于 2025.04.01 19:13

官方题解 写题解

执行用时分布
0 ms 击败 100.00%

消耗内存分布
18.04 MB 击败 95.00%

复杂度分析

```
1 from collections import deque
2 class Solution:
3     def levelOrder(self, root: Optional[TreeNode]) -> List[List[int]]:
4         if not root:
5             return []
6         ans = []
7         q = deque([root])
8         while q:
9             tmp = []
10            for _ in range(len(q)):
11                node = q.popleft()
```

已存储 行 20, 列 1

测试用例

LC131.分割回文串

dp, backtracking, <https://leetcode.cn/problems/palindrome-partitioning/>

思路:

代码:

class Solution:

```
def partition(self, s: str) -> List[List[str]]:
```

```
    def dfs(i):
```

```
        if i == len(s):
```

```
            lst.append(ans.copy())
```

```
            return
```

```
        for j in range(i, len(s)):
```

```
            if dp[i][j]:
```

```
                ans.append(s[i:j + 1])
```

```
                dfs(j + 1)
```

```
                ans.pop()
```

```
dp = [[True] * len(s) for _ in range(len(s))]
```

```
for i in range(len(s)-1, -1, -1):
```

```
    for j in range(i + 1, len(s)):
```

```
        dp[i][j] = (s[i] == s[j]) and dp[i + 1][j - 1]
```

```
lst = []
```

```
ans = []
```


```
dfs(0)
```

```
return lst
```

代码运行截图（至少包含有"Accepted"）

通过 32 / 32 个通过的测试用例

 ? 。 提交于 2025.04.01 21:55

 官方题解

 写题解

🕒 执行用时分布

ⓘ

39 ms | 击败 95.64% 🍀

💎 复杂度分析

💼 消耗内存分布

33.95 MB | 击败 22.32%

LC146.LRU 缓存

hash table, doubly-linked list, <https://leetcode.cn/problems/lru-cache/>

思路：

代码：

代码运行截图 （至少包含有"Accepted"）

2. 学习总结和收获

如果发现作业题目相对简单，有否寻找额外的练习题目，如“数算 2025spring 每日选做”、LeetCode、Codeforces、洛谷等网站上的题目。

要期中考试了，来不及了，这周作业主要复习了一下回溯，bfs dfs 的常规写法。这个期中周好像比我预想得来得早啊。。。。。较多地依赖了 deepseek 的思路导引和查错以及leetcode 关于树的写法，感觉期中考试完要开始恶补。