

# Assignment #6: 回溯、树、双向链表和哈希表

Updated 1526 GMT+8 Mar 22, 2025

2025 spring, Compiled by 王梓航、物理学院

## 说明:

### 1. 解题与记录:

对于每一个题目, 请提供其解题思路(可选), 并附上使用Python或C++编写的源代码(确保已在OpenJudge, Codeforces, LeetCode等平台上获得Accepted)。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。(推荐使用Typora <https://typoraio.cn> 进行编辑, 当然你也可以选择Word。)无论题目是否已通过, 请标明每个题目大致花费的时间。

2. **提交安排:** 提交时, 请首先上传PDF格式的文件, 并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的头像, 提交的文件为PDF格式, 并且“作业评论”区包含上传的.md或.doc附件。

3. **延迟提交:** 如果你预计无法在截止日期前提交作业, 请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。

请按照上述指导认真准备和提交作业, 以保证顺利完成课程要求。

## 1. 题目

### LC46.全排列

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

思路: 回溯处理即可

代码:

```
class Solution:
    from collections import deque
    def permute(self, nums: List[int]) -> List[List[int]]:
        a = []
        c = []
        def f(b):
            n = len(b)
            if n==0 and a:
                c.append(a[:])
            for _ in range(n):
                a.append(b.popleft())
                f(b)
                b.append(a.pop())
        b = deque(nums)
        f(b)
```

```
return c
```

代码运行截图 (至少包含有"Accepted")

	所有状态 ▾	所有语言 ▾	执行用时	消耗内存	备注
2	通过 2025.03.08	Python3	🕒 0 ms	💾 17.7 MB	

## LC79: 单词搜索

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


思路: dfs, 回溯处理

代码:

```
class Solution:
    def exist(self, board: List[List[str]], word: str) -> bool:
        a = list(word)
        n = len(a)
        l, m = len(board), len(board[0])
        def f(i, x, y):
            if i == n:
                return True
            for dx, dy in [(-1, 0), (1, 0), (0, 1), (0, -1)]:
                px, py = x + dx, y + dy
                if 0 <= px < l and 0 <= py < m and board[px][py] == a[i]:
                    board[px][py] = ''
                    if f(i + 1, px, py):
                        return True
                    board[px][py] = a[i]
            return False
        for p in range(l):
            for q in range(m):
                if board[p][q] == a[0]:
                    board[p][q] = ''
                    if f(1, p, q):
                        return True
                    board[p][q] = a[0]
        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

 Hungry NorthcuttqqE 提交于 2025.03.27 15:40

## LC94.二叉树的中序遍历

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

思路: 正常写递归即可

代码:

```
class Solution:
    def inorderTraversal(self, root: Optional[TreeNode]) -> List[int]:
        def f(root):
            if not root:
                return []
            left = f(root.left)
            right = f(root.right)
            return left+[root.val]+right
        return f(root)
```

代码运行截图 (至少包含有"Accepted")

1 通过 Python3 0 ms 17.7 MB  
2025.02.17

## LC102.二叉树的层序遍历

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

思路:

代码:

```
class Solution:
    def levelOrder(self, root: Optional[TreeNode]) -> List[List[int]]:
        a = []
        def f(n,x):
            if not x:
                return
            if len(a)<n:
```


```

        a.append([x.val])
    else:
        a[n-1].append(x.val)
    f(n+1,x.left)
    f(n+1,x.right)
f(1,root)
return a

```

代码运行截图 (至少包含有"Accepted")

 35 / 35 个通过的测试用例

 Hungry NorthcuttqqE 提交于 2025.03.27 15:53

## 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]]:
        n = len(list(s))
        c = []
        res = []
        def f(ans,j):
            if j==n-1:
                res.append(ans[:])
                return
            temp = ''
            for q in range(j+1,n):
                index = s[q]
                temp+=index
                if temp==temp[::-1]:
                    ans.append(temp)
                    f(ans,q)
                    ans.pop()
            f([],-1)
        return res

```

代码运行截图 (至少包含有"Accepted")

通过 32 / 32 个通过的测试用例

 Hungry NorthcuttqqE 提交于 2025.03.27 16:20

## LC146.LRU缓存

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

思路：没想到用链表，所以用deque做的

代码：

```
class LRUCache:
    from collections import defaultdict, deque


    def __init__(self, capacity: int):
        self.cap=capacity
        self.dic = defaultdict(lambda:-1)
        self.num = 0
        self.other = deque([])
        self.new = {}

    def get(self, key: int) -> int:
        if self.dic[key]!=-1:
            self.other.append(key)
            self.new[key]+=1
        return self.dic[key]

    def put(self, key: int, value: int) -> None:
        if self.dic[key]==-1:
            self.num+=1
            self.dic[key]=value
            self.new.setdefault(key,0)
            self.other.append(key)
            self.new[key]+=1
        while self.num>self.cap:
            b = self.other.popleft()
            self.new[b]-=1
            if self.new[b]==0:
                self.dic[b]=-1
                self.num-=1
```

代码运行截图 (至少包含有"Accepted")

通过 23 / 23 个通过的测试用例

 Hungry NorthcuttqqE 提交于 2025.03.27 16:58

## 2. 学习总结和收获

---

每日选做在跟进，最近要准备期中，投入时间会少点，但是还是尽量多做点题。