

# Assignment #8: 递归

Updated 1315 GMT+8 Oct 21, 2025

2025 fall, Complied by 郑文萱 基础医学院

## 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本人头像, 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

M04147汉诺塔问题(Tower of Hanoi)

dfs, <http://cs101.openjudge.cn/pctbook/M04147>

思路: 发现递归跟套娃一样, 做的时候不能想太多, 想多了就不会了, 把递推式搞清楚就行了。汉诺塔就是这样, 剩一个了直接移动, 不是一个, 就把除了那一个的n-1个先移动到中转站, 后移动到目标杆, 具体怎么移不管。

代码

```
def hanoi(n,source,medium,target):  
    if n == 1:  
        print(f"1:{source}->{target}")  
    else:  
        hanoi(n-1,source,target,medium)  
        print(f"{n}:{source}->{target}")  
        hanoi(n-1,medium,source,target)  
n,a,b,c=input().split()  
n=int(n)  
hanoi(n,a,b,c)
```

## 代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
def hanoi(n,source,medium,target):
    if n == 1:
        print(f"1:{source}->{target}")
    else:
        hanoi(n-1,source,target,medium)
        print(f"{n}:{source}->{target}")
        hanoi(n-1,medium,source,target)
n,a,b,c=input().split()
n=int(n)
hanoi(n,a,b,c)
```

## 基本信息

#: 50661056  
 题目: M04147  
 提交人: 25n2510305212(zx)  
 内存: 3512kB  
 时间: 18ms  
 语言: Python3  
 提交时间: 2025-11-01 20:34:45

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

## M05585: 矿石的个数

matrices, dfs similar, <http://cs101.openjudge.cn/pctbook/M05585>

思路: 这个有点难, 是Dfs, 第一次正经做这种, 向AI请教了一些。使用了上次作业里面储存四个方向以便检查的方法, 并把搜过的都改成'#', 感觉有了上周和上上周矩阵的铺垫, 思考解题方式快了很多

## 代码

```
def solve():
    k=int(input())
    for _ in range(k):
        n=int(input())
        map=[list(input().strip()) for _ in range(n)]
        rcount = 0
        bcount = 0
        for i in range(n):
            for j in range(n):
                if map[i][j] in ['r', 'b']:
                    mineral_type=map[i][j]
                    stack=[(i,j)]
                    map[i][j]='#'

                    while stack:
                        x,y=stack.pop()
                        for dx,dy in [(0,1),(0,-1),(1,0),(-1,0)]:
                            nx,ny=x+dx,y+dy
                            if 0<=nx< n and 0<=ny< n and map[nx][ny]==mineral_type:
                                stack.append((nx,ny))
                                map[nx][ny]='#'
                    if mineral_type=='r':
                        rcount+=1
                    else:
                        bcount+=1
        print(f"{rcount} {bcount}")
solve()
```

## 代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```

def solve():
    k=int(input())
    for _ in range(k):
        n=int(input())
        map=[list(input().strip()) for _ in range(n)]
        rcount = 0
        bcount = 0
        for i in range(n):
            for j in range(n):
                if map[i][j] in ['r', 'b']:
                    mineral_type=map[i][j]
                    stack=[(i,j)]
                    map[i][j]='#'

                    while stack:
                        x,y=stack.pop()
                        for dx,dy in [(0,1),(0,-1),(1,0),(-1,0)]:
                            nx,ny=x+dx,y+dy
                            if 0<=nx<=n and 0<=ny<=n and map[nx][ny]==mineral_type:
                                stack.append((nx,ny))
                                map[nx][ny]='#'
                    if mineral_type=='r':
                        rcount+=1
                    else:
                        bcount+=1
        print(f'{rcount} {bcount}')
solve()

```

基本信息

#: 50661478  
 题目: M05585  
 提交人: 25n2510305212(zx)  
 内存: 3664kB  
 时间: 19ms  
 语言: Python3  
 提交时间: 2025-11-01 21:04:47

## M02786: Pell数列

dfs, dp, <http://cs101.openjudge.cn/pctbook/M02786/>

思路: 哈哈哈这个简单, 这不就稍微变式的斐波那契数列吗, 我一开始是这么想的, 结果第一遍超时了! 才发现无脑写递归时间复杂度都 $O(n^2)$ 了, 改成动态规划用循环去做, 哈哈, 又超时啦! (其实wait很长时间的时候就已经感觉不对劲了) 改成每算一个数就及时取余之后很快速地通过了, 这个真的涨知识了, 以后一定记住及时取余, 不影响结果。

代码

```

def pell(m):
    if m==1:
        return 1
    elif m==2:
        return 2
    else:
        a,b=1,2
        for i in range(3,m+1):
            a,b=b,(2*b+a)%32767
        return b
k=int(input())
for _ in range(k):
    n=int(input())
    print(pell(n))

```

## 代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```

def pell(m):
    if m==1:
        return 1
    elif m==2:
        return 2
    else:
        a,b=1,2
        for i in range(3,m+1):
            a,b=b, (2*b+a)%32767
        return b
k=int(input())
for _ in range(k):
    n=int(input())
    print(pell(n))

```

## 基本信息

#: 50661799  
 题目: M02786  
 提交人: 25n2510305212(zx)  
 内存: 3600kB  
 时间: 225ms  
 语言: Python3  
 提交时间: 2025-11-01 21:28:21

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

## M46.全排列

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

思路: 这个也是dfs, 其实也是递归的应用, 递推思路是先选定一个数然后把剩下的全排列。交了半天交不上, 我服了, 他这代码给了函数, 我看都没看直接删了, 还纳闷为啥输入里还带num, 结果恢复了一下才发现必须按他的写, 被自己蠢笑了。看了最快的解法, 确实很优雅, 时间复杂度降低不少。

## 代码

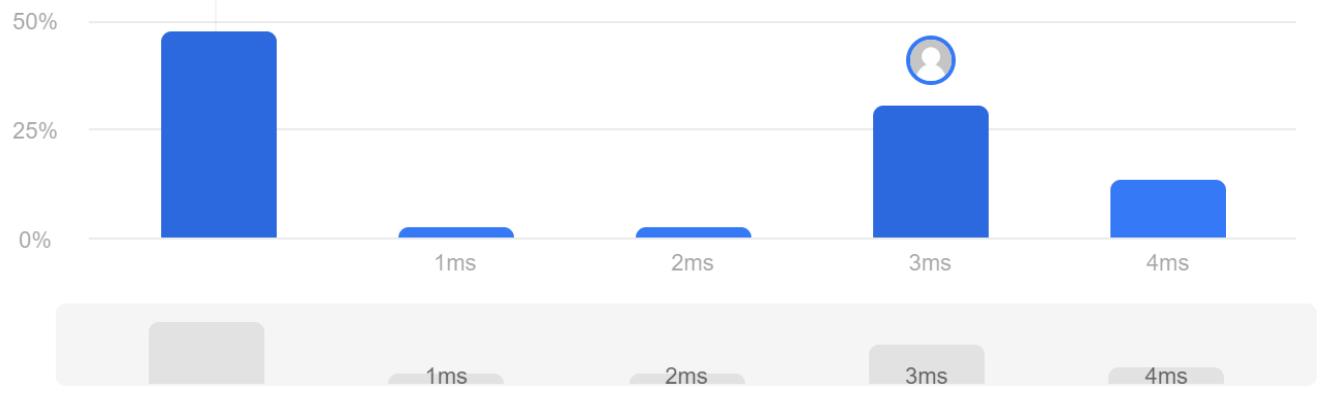
```

class Solution:
    def permute(self, nums: List[int]) -> List[List[int]]:
        def fenxi(dangqian):
            if len(dangqian) == len(nums):
                result.append(dangqian[:])
                return
            for num in nums:
                if num not in dangqian:
                    dangqian.append(num)
                    fenxi(dangqian)
                    dangqian.pop()
        result = []
        fenxi([])
        return result

```

## 代码运行截图 (至少包含有"Accepted")

47.93% 的用户使用了类似解法 Runtime: 0 ms



码 | Python3

```
class Solution:
    def permute(self, nums: List[int]) -> List[List[int]]:
        def fenxi(dangqian):
            if len(dangqian) == len(nums):
                result.append(dangqian[:])
                return
            for num in nums:
                if num not in dangqian:
                    dangqian.append(num)
                    fenxi(dangqian)
                    dangqian.pop()
        result = []
        fenxi([])
        return result
```

T02754: 八皇后

dfs and similar, <http://cs101.openjudge.cn/pctbook/T02754>

思路:

代码

## 代码运行截图 (至少包含有"Accepted")

T01958 Strange Towers of Hanoi

<http://cs101.openjudge.cn/practice/01958/>

思路:

代码

代码运行截图 (至少包含有"Accepted")

## 2. 学习总结和收获

如果作业题目简单，有否额外练习题目，比如：OJ“计概2025fall每日选做”、CF、LeetCode、洛谷等网站题目。前四个题很好的理解了递归还有深搜，后两题明显难好多啊，都不太有思路（下周还有期中考有点焦头烂额啊抱歉），还有这两个都是经典（难）题所以以后肯定会做到的。