

# Assignment #8: 递归

Updated 1315 GMT+8 Oct 21, 2025

2025 fall, Compiled by 同学的姓名、院系

## 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：深刻感受到了递归的优雅，感觉代码基本和自己在纸上写的思路长得一模一样。

代码

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

代码运行截图 (至少包含有"Accepted")

#50603825提交状态

查看 提交 统计 提问

状态: Accepted

源代码

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

基本信息

#: 50603825  
 题目: M04147  
 提交人: 25n2500011474  
 内存: 3540kB  
 时间: 24ms  
 语言: Python3  
 提交时间: 2025-10-28 17:29:43

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

## M05585: 晶矿的个数

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

思路: 感觉这题就是经典的dfs题型, 上课的时候dfs没太理解, 课后学习了一下这题以及二叉树的前序遍历, 算是理解了一些, 其算法的实现方法确实就是递归。

代码

```
def nums(m):
    hang,lie=len(m),len(m[0])
    table=[[False for _ in range(lie)] for _ in range(hang)]
    nums_r=0
    nums_b=0

    def dfs(i,j,cry):
        if 0<=i<hang and 0<=j<lie and m[i][j]==cry and not table[i][j]:
            table[i][j]=True
            dfs(i+1,j,cry)
            dfs(i-1,j,cry)
            dfs(i,j-1,cry)
            dfs(i,j+1,cry)

    for i in range(hang):
        for j in range(lie):
            if m[i][j]=='r' and not table[i][j]:
                dfs(i,j,'r')
                nums_r+=1
            if m[i][j]=='b' and not table[i][j]:
                dfs(i,j,'b')
                nums_b+=1
    return nums_r,nums_b

k=int(input())
for _ in range(k):
    n=int(input())
    m=[[ ] for _ in range(n)]
```

```

for i in range(n):
    district=input()
    for k in district:
        m[i].append(k)
print(*nums(m))

```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```

def nums(m):
    hang, lie = len(m), len(m[0])
    table = [[False for _ in range(lie)] for _ in range(hang)]
    nums_r = 0
    nums_b = 0

    def dfs(i, j, cry):
        if 0 <= i < hang and 0 <= j < lie and m[i][j] == cry and not table[i][j]:
            table[i][j] = True
            dfs(i+1, j, cry)
            dfs(i-1, j, cry)
            dfs(i, j-1, cry)
            dfs(i, j+1, cry)

    for i in range(hang):
        for j in range(lie):
            if m[i][j] == 'r' and not table[i][j]:
                dfs(i, j, 'r')
                nums_r += 1
            if m[i][j] == 'b' and not table[i][j]:
                dfs(i, j, 'b')
                nums_b += 1
    return nums_r, nums_b

k = int(input())
for _ in range(k):
    n = int(input())
    m = [[] for _ in range(n)]
    for i in range(n):
        district = input()
        for k in district:
            m[i].append(k)
    print(*nums(m))

```

基本信息

#: 50618992  
 题目: M05585  
 提交人: 25n2500011474  
 内存: 3712kB  
 时间: 22ms  
 语言: Python3  
 提交时间: 2025-10-29 16:52:29

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

## M02786: Pell数列

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

思路: 这道递归感觉还是比较直接简单的, 题目已经告诉了你递归的方式, 只需要在程序里执行就行, 值得注意的是, 可以提前在计算的时候取模防止结果太大。

代码

```

def pell(k):
    if k == 1:
        return 1
    elif k == 2:
        return 2

```

```

    a1, a2 = 1, 2
    for i in range(3, k + 1):
        a1, a2 = a2, (2 * a2 + a1) % 32767

    return a2

n = int(input())
for _ in range(n):
    k = int(input())
    print(pell(k))

```

代码运行截图 (至少包含有"Accepted")

#50627659提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```

def pell(k):
    if k == 1:
        return 1
    elif k == 2:
        return 2

    a1, a2 = 1, 2
    for i in range(3, k + 1):
        a1, a2 = a2, (2 * a2 + a1) % 32767

    return a2

n = int(input())
for _ in range(n):
    k = int(input())
    print(pell(k))

```

基本信息

#: 50627659  
 题目: M02786  
 提交人: 25n2500011474  
 内存: 3616kB  
 时间: 228ms  
 语言: Python3  
 提交时间: 2025-10-30 10:28:54

©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：这题的递归思路是：对于N排列，先选出一个数排在第一个，然后执行N-1排列，依次类推。不过这个涉及的列表有点多，感觉有点绕。同时通过这题也大概对回溯有了一点概念。

代码

```

class Solution:
    def permute(self, nums: List[int]) -> List[List[int]]:
        nums_list=[]
        nums_bool=[False]*len(nums)

        def permutation(A):
            if len(nums)==len(A):
                nums_list.append(A.copy())
                return
            for i in range(len(nums)):
                if not nums_bool[i]:

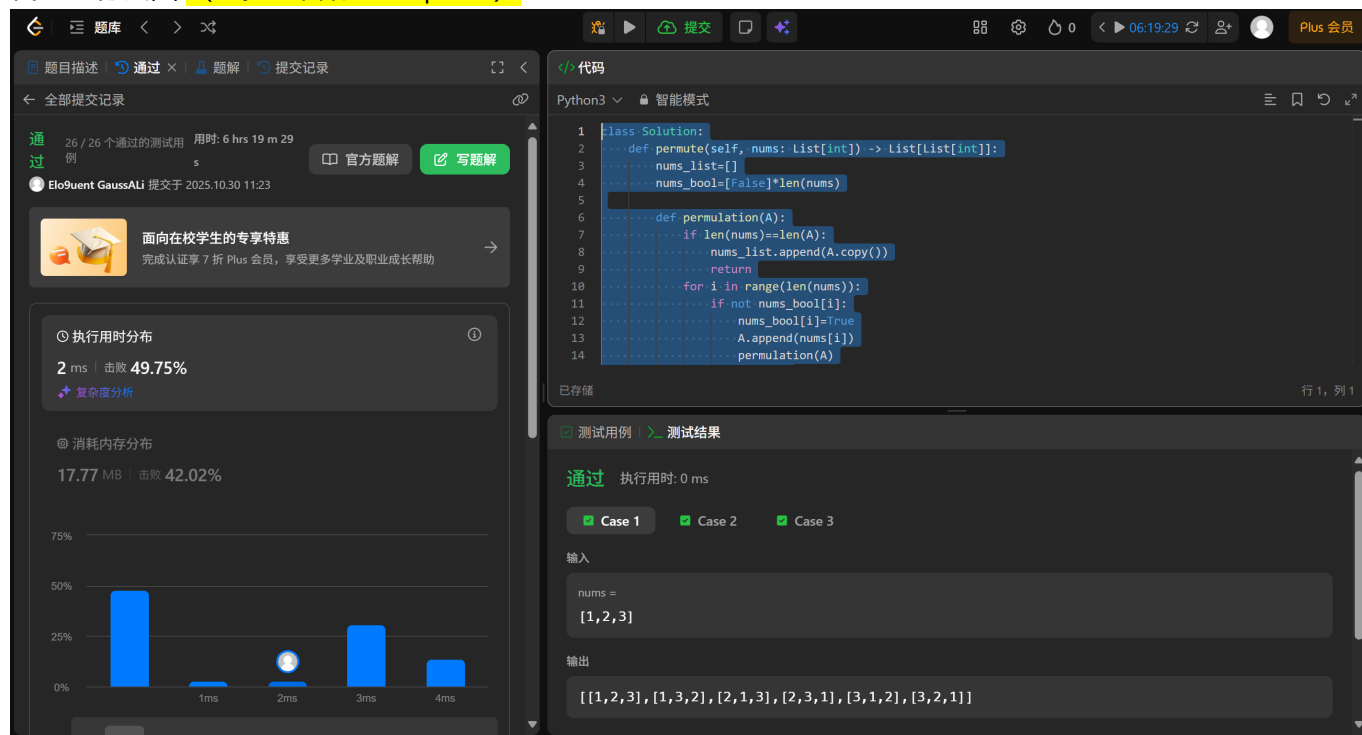
```

```

        nums_bool[i]=True
        A.append(nums[i])
        permutation(A)
        A.pop()
        nums_bool[i]=False
    permutation([])
    return nums_list

```

代码运行截图 (至少包含有"Accepted")



T02754: 八皇后

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

思路：思路仍然是递归，先写出一个判断这一位置能不能放置的函数，利用这一函数，对第j行（写的时候i, j 经常搞混）进行判断并过渡到下一行。

代码

```

def check(j, i, map):
    for m in range(j):
        if map[m] == i or abs(map[m] - i) == abs(m - j):
            return False
    return True

n=8
Ans = []
map=[-1]*n
def queen(j, n, map):
    if j == n:
        Ans.append(map[:])
        return
    for i in range(n):

```

```

        if check(j, i, map):
            map[j] = i
            queen(j + 1, n, map)
            map[j] = -1
n=int(input())
queen(0,8,map)

Ans1=[]
for i in Ans:
    a=0
    for j in range(8):
        a+=(i[j]+1)*(10**(8-j-1))
    Ans1.append(a)
Ans1.sort()

for _ in range(n):
    i=int(input())
    print(Ans1[i-1])

```

代码运行截图 (至少包含有"Accepted")

#50637992提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```

def check(j, i, map):
    for m in range(j):
        if map[m] == i or abs(map[m] - i) == abs(m - j):
            return False
    return True

n=8
Ans = []
map=[-1]*n
def queen(j, n, map):
    if j == n:
        Ans.append(map[:])
        return
    for i in range(n):
        if check(j, i, map):
            map[j] = i
            queen(j + 1, n, map)
            map[j] = -1

n=int(input())
queen(0,8,map)

Ans1=[]
for i in Ans:
    a=0
    for j in range(8):
        a+=(i[j]+1)*(10**(8-j-1))
    Ans1.append(a)
Ans1.sort()

for _ in range(n):
    i=int(input())
    print(Ans1[i-1])

```

基本信息

#: 50637992  
 题目: T02754  
 提交人: 25n2500011474  
 内存: 3628kB  
 时间: 35ms  
 语言: Python3  
 提交时间: 2025-10-31 10:11:42

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

思路：会了三塔问题之后，感觉四塔只是规模大一点，同时需要添加一个求最小值的部分即可。

代码

```
def three_tower(n):
    if n==1:
        return 1
    if n==0:
        return 0
    else:
        times=0
        times+=three_tower(n-1)
        times+=1
        times+=three_tower(n-1)
        return times

def four_tower(n):
    if n==1:
        return 1
    if n==0:
        return 0
    min_times = 0
    for i in range(1,n+1):
        times=0
        times+=four_tower(n-i)
        times+=three_tower(i)
        times+=four_tower(n-i)
        if times<min_times or min_times==0:
            min_times=times
    return min_times

for i in range(1,13):
    print(four_tower(i))
```

代码运行截图 (至少包含有"Accepted")

#50638722提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```
def three_tower(n):
    if n==1:
        return 1
    if n==0:
        return 0
    else:
        times=0
        times+=three_tower(n-1)
        times+=1
        times+=three_tower(n-1)
        return times

def four_tower(n):
    if n==1:
        return 1
    if n==0:
        return 0
    min_times = 0
    for i in range(1,n+1):
        times=0
        times+=four_tower(n-i)
        times+=three_tower(i)
        times+=four_tower(n-i)
        if times<min_times or min_times==0:
            min_times=times
    return min_times

for i in range(1,13):
    print(four_tower(i))
```

基本信息

#: 50638722  
题目: 01958  
提交人: 25n2500011474  
内存: 3544kB  
时间: 217ms  
语言: Python3  
提交时间: 2025-10-31 11:36:21

2. 学习总结和收获

这周在忙期中考，分给计概的时间不多，只是总结了一下之前的内容，并做了一部分递归的习题