

## T25353:排队

这题是上周的，做了很久很久，最终还是看了题解（唉，第一次作业题没有独立完成）。最开始 1h 内就写出了思路正确的代码（最坏时间复杂度  $n^2$ ），但超时了。尝试了一些优化，都不行，且我发现无论怎么优化，这种思路的最坏时间复杂度都是  $n^2$ 。于是我误认为应该用其他思路解决，且最坏时间复杂度应小于  $n^2$ （从此处开始已经走偏了，哈哈），但我又注意到简单排序的复杂度至少是  $n \cdot \ln n$ ，所以说可行的思路大体上只有 3 种：1. 改编快速排序的代码，2. 改编归并排序的代码，3. 一次遍历+分治 sort。1 看上去就很扯，2 和 3 如果要记录很多值，那么最坏时间复杂度就不小于  $n^2$  了，这样剩余的思路就很少了。所以当时我抱着穷举（瞎猜）的心态，写了几个代码，但都不行。真的想不到别的思路了，遂看题解。结果……所有题解的思路和我最初思路从数学角度都是极为相似的，且最坏复杂度也都是  $n^2$ ……我意识到自己一个问题：始终致力于优化最坏的情况，但实际上绝大多数时候，所需求的都是代码的平均运行速度（这也是快速排序优于归并排序的原因之一）

## M18160:最大连通域面积

和红黑矿那题几乎一样

The screenshot shows the OpenJudge platform interface. The top navigation bar includes 'OpenJudge', '题目ID, 标题, 描述' (Topic ID, Title, Description), a search icon, and user information 'cmyjf' (用户名), '信箱' (Email), and '账号' (Account). Below the navigation is a header for 'CS101 / 计算思维算法实践' (CS101 / Computational Thinking Algorithm Practice) with tabs for '题目' (Topic), '排名' (Ranking), '状态' (Status), and '提问' (Ask). The main content area displays the submission status for topic #50852607:

**#50852607 提交状态**

状态: Accepted

源代码:

```
def dfs(x,y):
    if ma[x][y] == '.':
        return 0
    else:
        ans = 1
        ma[x][y] = '.'
        ans += dfs(max(0,x-1),y)
        ans += dfs(min(n-1,x+1), y)
        ans += dfs(x,max(0,y-1))
        ans += dfs(x,min(m-1,y+1))
        ans += dfs(max(0,x-1),min(m-1,y+1))
        ans += dfs(min(n-1,x+1), max(0,y-1))
        ans += dfs(max(0,x-1),max(0,y-1))
        ans += dfs(min(n-1,x+1),min(m-1,y+1))
    return ans

t = int(input())
li = []
for _ in range(t):
    n,m = map(int,input().split())
    ma = []
    for _ in range(n):
        a = input()
        b = []
        for i in a:
            b.append(i)
        ma.append(b)
    ans = 0
    for x in range(n):
```

基本信息:

- #: 50852607
- 题目: M18160
- 提交人: cmyjf
- 内存: 3712kB
- 时间: 146ms
- 语言: Python3

提交时间: 2025-11-15 16:26:19

## sy134: 全排列 III 中等

用 permutations 就是快

(⊙) 晴问

课程 训练营 算法笔记 题库 题单 比赛 语言入门教程 2026考研算法全程训练营

《2026考研算法：全程训练营（初试 & 机试）》已经上线：<https://sunnywhy.com/camp/3415> 按 F11 即可退出全屏模式

**题目** 题解

代码书写 Python

第二行按升序给出n个可能重复的正整数（每个正整数均不超过100）。

**输出描述**

每个全排列一行，输出所有全排列。

输出顺序为：两个全排列A和B，若满足前 $k-1$ 项对应相同，但有 $A_k < B_k$ ，那么将全排列A优先输出（例如[1, 2, 3]比[1, 3, 2]优先输出）。

在输出时，全排列中的每个数之间用一个空格隔开，行末不允许有多余的空格。不允许出现相同的全排列。

**样例1**

输入 复制  
3 1 1 3

输出 复制  
1 1 3  
1 3 1  
3 1 1

代码示例（Python）：

```

1 import itertools
2 from itertools import permutations
3 n = input().split()
4 li = list(map(int, input().split()))
5 a = permutations(li)
6 a = set(a)
7 a = list(a)
8 a.sort()
9 for i in a:
10     for j in range(len(i)-1):
11         print(i[j], end=' ')
12     print(i[-1])

```

测试输入 提交结果 历史提交

完美通过 查看题解

100% 数据通过测试 详情  
运行时长: 0 ms

收起面板 运行 提交

## sy136: 组合 II 中等

自己写的递归，随后又查询了 Python 中的 `itertools.combinations_with_replacement` 和 `itertools.combinations` 函数

(⊙) 晴问

课程 训练营 算法笔记 题库 题单 比赛 语言入门教程 2026考研算法全程训练营

《2026考研算法：全程训练营（初试 & 机试）》已经上线：<https://sunnywhy.com/camp/3415> 按 F11 即可退出全屏模式

**题目** 题解

代码书写 Python

每个组合一行，输出所有组合。

输出顺序为：两个组合A和B，若各自升序后满足前 $k-1$ 项对应相同，但有 $A_k < B_k$ ，那么将组合A优先输出（例如[1, 5, 9]比[1, 5, 10]优先输出）。

在输出组合时，组合内部按升序输出，组合中的每个数之间用一个空格隔开，行末不允许有多余的空格。不允许出现相同的组合。

**样例1**

输入 复制  
3 2  
1 2 3

输出 复制  
1 2  
1 3  
2 3

代码示例（Python）：

```

4 a = []
5 ans = []
6 def dfs(li, l, p):
7     if l < k-p:
8         pass
9     elif p == k:
10        for i in range(l):
11            a.append(li[i])
12        ans.append(a[:])
13        a.pop()
14    else:
15        for i in range(l-1):
16            a.append(li[i])
17            dfs(li[i+1:], l-1-i, p+1)
18        a.pop()
19    dfs(li, n, 1)
20    for i in ans:
21        for j in range(len(i)-1):
22            print(i[j], end=' ')
23        print(i[-1])

```

测试输入 提交结果 历史提交

完美通过 查看题解

100% 数据通过测试 详情  
运行时长: 0 ms

收起面板 运行 提交

## sy137: 组合 III 中等

## 类似上题

The screenshot shows a programming competition interface. The problem title is "组合数对" (Combination Pairs). The input description specifies two positive integers  $n, k$  ( $1 \leq k \leq n \leq 12$ ) representing the number of elements in each set. The output description specifies that combinations should be printed in lexicographical order. The sample input is "3 2", and the sample output is "1 1 3 2". The code editor contains a Python script that implements a DFS algorithm to generate combinations and prints them in the required order. The status bar indicates the code has passed all tests.

## M04123: 马走日

The screenshot shows the OpenJudge platform with the problem ID M04123. The status is "Accepted". The code implements a backtracking algorithm to find paths for a knight on an  $n \times m$  chessboard. It uses a global variable 'ans' to store the count of valid paths and a matrix 'ma' to mark visited cells. The code handles boundary conditions and recursive calls for the knight's moves. The basic information panel shows the submission details: user cmvjf, language Python3, and submission time 2025-11-15 17:36:23.

## T02287: Tian Ji -- The Horse Racing

很有趣的贪心，证明过程和操作思路几乎相同，可能只有最后一步比较跳跃，但也不

是想不到（反证法 yyds）

OpenJudge

题目ID, 标题, 描述  cmjyjf 信箱 账号

**CS101 / 计算思维算法实践**

**#50856165提交状态**

状态: Accepted

基本信息

#: 50856165  
题目: T02287  
提交人: cmjyjf  
内存: 3976kB  
时间: 56ms  
语言: Python3  
提交时间: 2025-11-15 19:56:43

源代码

```
from collections import deque
while 1:
    n = int(input())
    if n == 0:
        break
    else:
        l11 = list(map(int, input().split()))
        l12 = list(map(int, input().split()))
        l11.sort()
        l12.sort()
        ans = 0
        de1 = deque(l11)
        de2 = deque(l12)
        while de1:
            if de1[-1] > de2[-1]:
                de1.pop()
                de2.pop()
                ans += 1
            elif de1[-1] < de2[-1]:
                de1.popleft()
                de2.pop()
                ans -= 1
            else:
                if de1[0] < de2[0]:
                    de1.popleft()
                    de2.pop()
                    ans -= 1
                elif de1[0] > de2[0]:
                    de1.popleft()
```