Assignment #C: 矩阵、递归、贪心、和dfs simlar

Updated 1126 GMT+8 Nov 28, 2023

2023 fall, Complied by Xinjie Song, Phy

说明：

本周作业还是难题较多，建议提前开始作业，如果耗时太长，直接找答案看。两个题解，经常更新。所以最好从这个链接下载最新的，https://github.com/GMyhf/2020fall-cs101 。

1）请把每个题目解题思路（可选），源码Python, 或者C++（已经在Codeforces/Openjudge上AC），截图（包含Accepted, 学号），填写到下面作业模版中（推荐使用 typora https://typoraio.cn ，或者用word）。AC 或者没有AC，都请标上每个题目大致花费时间。

2）提交时候先提交pdf文件，再把md或者doc文件上传到右侧“作业评论”。Canvas需要有同学清晰头像、提交文件有pdf、作业评论有md或者doc。

3）如果不能在截止前提交作业，请写明原因。

编程环境

操作系统：Windows 11 22H2

Python编程环境：PyCharm 2023.2 (Community Edition)

C/C++编程环境：g++ (x86_64-win32-seh-rev0, Built by MinGW-W64 project) 8.1.0

1. 题目

如果耗时太长，直接看解题思路，或者源码

CF1881C. Perfect Square

brute force, implementation, 1200, https://codeforces.com/problemset/problem/1881/C

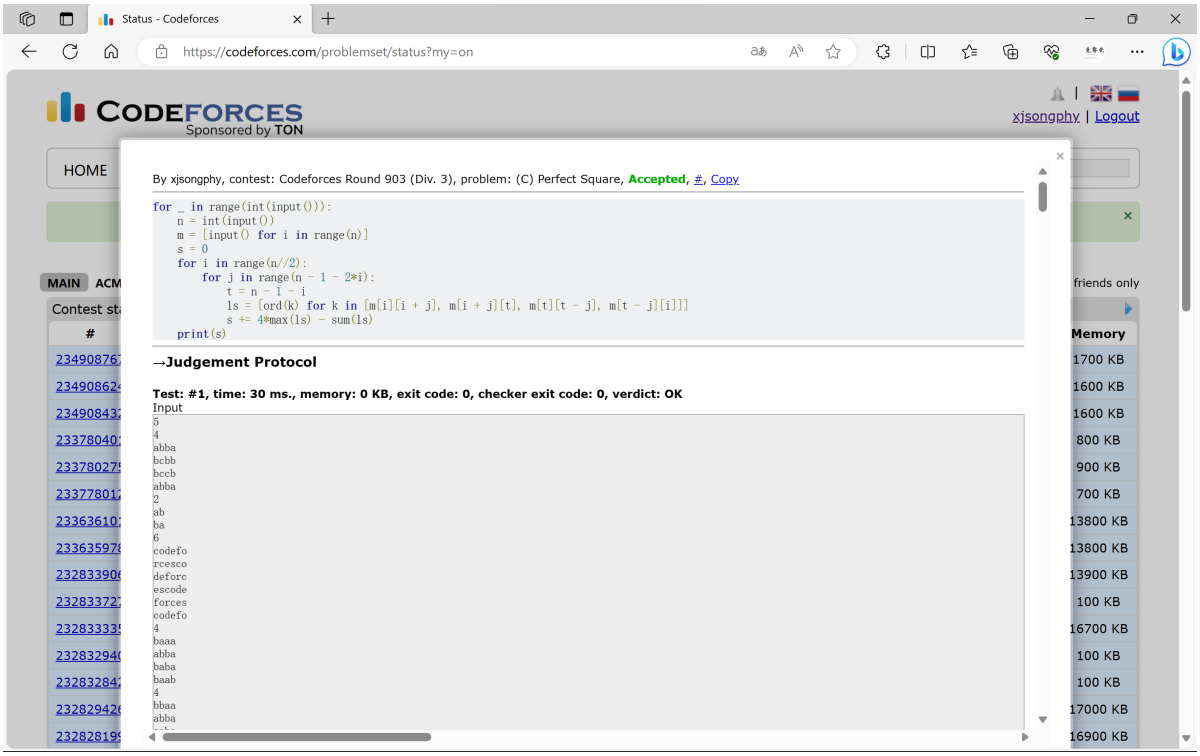
黄源森推荐：”一个一般的矩阵“。感觉现在CF problemset第一页的题（难度1000+的）都不是那么好做。

思路：旋转后重合的4个点为一组，每组所需步数为将这组每个点上的字符变为这组字典序最大的字符的步数的和。

代码

```
for _ in range(int(input())):
    n = int(input())
    m = [input() for i in range(n)]
    s = 0
    for i in range(n//2):
        for j in range(n - 1 - 2*i):
            t = n - 1 - i
            ls = [ord(k) for k in [m[i][i + j], m[i + j][t], m[t][t - j], m[t -
j][i]]]
            s += 4*max(ls) - sum(ls)
    print(s)
```

代码运行截图



OJ02694: 波兰表达式

recursion, data structure, http://cs101.openjudge.cn/practice/02694/

思路：正常递归即可

代码

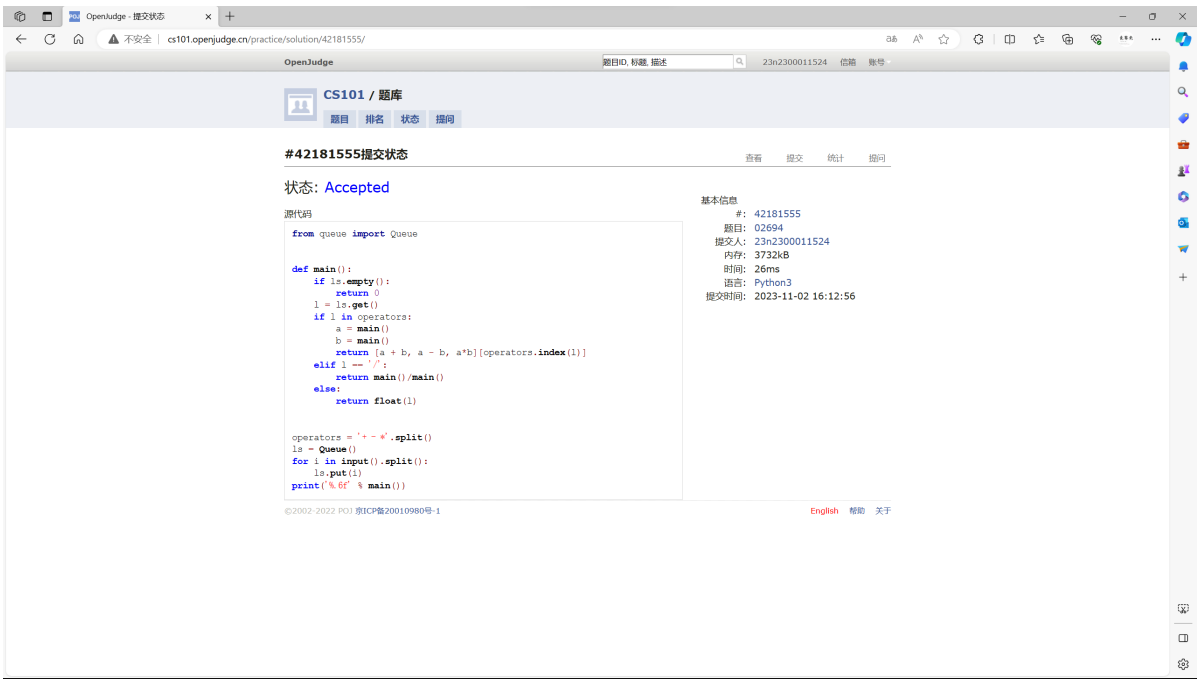
```
from queue import Queue


def main():
```

```
    if ls.empty():
        return 0
    l = ls.get()
    if l in operators:
        a = main()
        b = main()
        return [a + b, a - b, a*b][operators.index(l)]
    elif l == '/':
        return main()/main()
    else:
        return float(l)


operators = '+ - *'.split()
ls = Queue()
for i in input().split():
    ls.put(i)
print('%.6f' % main())
```

代码运行截图



OJ18160: 最大连通域面积

dfs similar, http://cs101.openjudge.cn/practice/18160

思路：常规思路

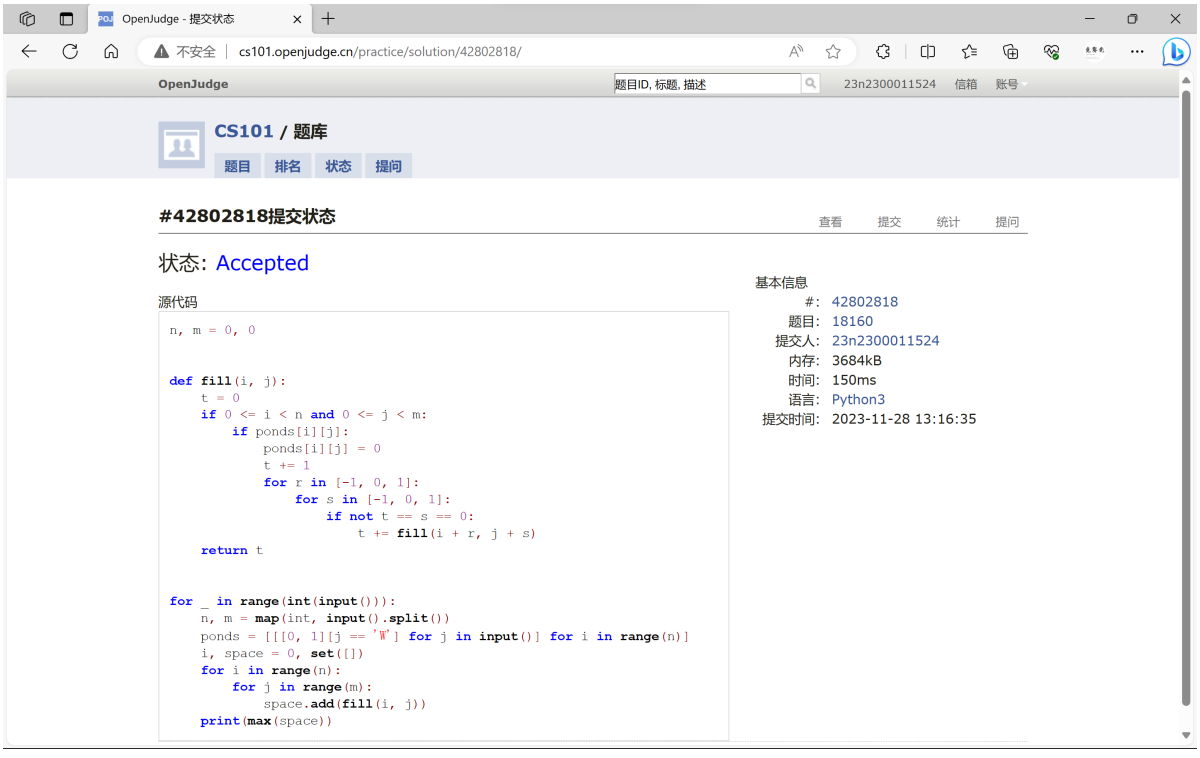
代码

```
n, m = 0, 0


def fill(i, j):
    t = 0
    if 0 <= i < n and 0 <= j < m:
        if ponds[i][j]:
            ponds[i][j] = 0
            t += 1
            for r in [-1, 0, 1]:
                for s in [-1, 0, 1]:
                    if not t == s == 0:
                        t += fill(i + r, j + s)
    return t


for _ in range(int(input())):
    n, m = map(int, input().split())
    ponds = [[[0, 1][j == 'W'] for j in input()] for i in range(n)]
    i, space = 0, set([])
    for i in range(n):
        for j in range(m):
            space.add(fill(i, j))
    print(max(space))
```

代码运行截图



OJ02754: 八皇后

dfs, http://cs101.openjudge.cn/practice/02754

思路：注意审题，方向不能改变.....

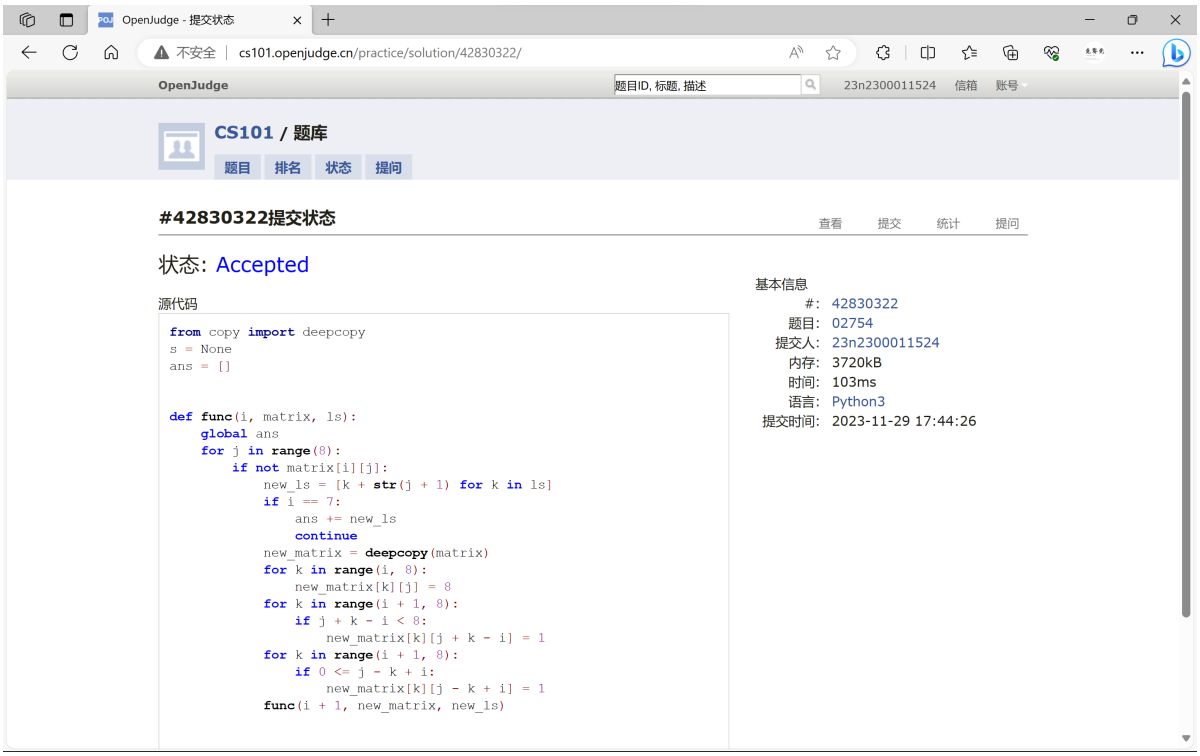
代码

```
from copy import deepcopy
s = None
ans = []


def func(i, matrix, ls):
    global ans
    for j in range(8):
        if not matrix[i][j]:
            new_ls = [k + str(j + 1) for k in ls]
            if i == 7:
                ans += new_ls
                continue
            new_matrix = deepcopy(matrix)
            for k in range(i, 8):
                new_matrix[k][j] = 8
            for k in range(i + 1, 8):
                if j + k - i < 8:
                    new_matrix[k][j + k - i] = 1
            for k in range(i + 1, 8):
                if 0 <= j - k + i:
                    new_matrix[k][j - k + i] = 1
            func(i + 1, new_matrix, new_ls)


func(0, [[0]*8 for _ in range(8)], [''])
ans.sort()
print(' '.join(ans))
output = ''
for _ in range(int(input())):
    output += ans[int(input()) - 1] + '\n'
print(output)
```

代码运行截图



OJ18146: 乌鸦坐飞机

http://cs101.openjudge.cn/routine/18146/

查达闻推荐：乌鸦坐飞机和装箱子那道题很像，其实难度不比装箱子高 但是考虑的情况确实不少。

思路：注意下面的特殊情况即可

$$A-A-0-B$$

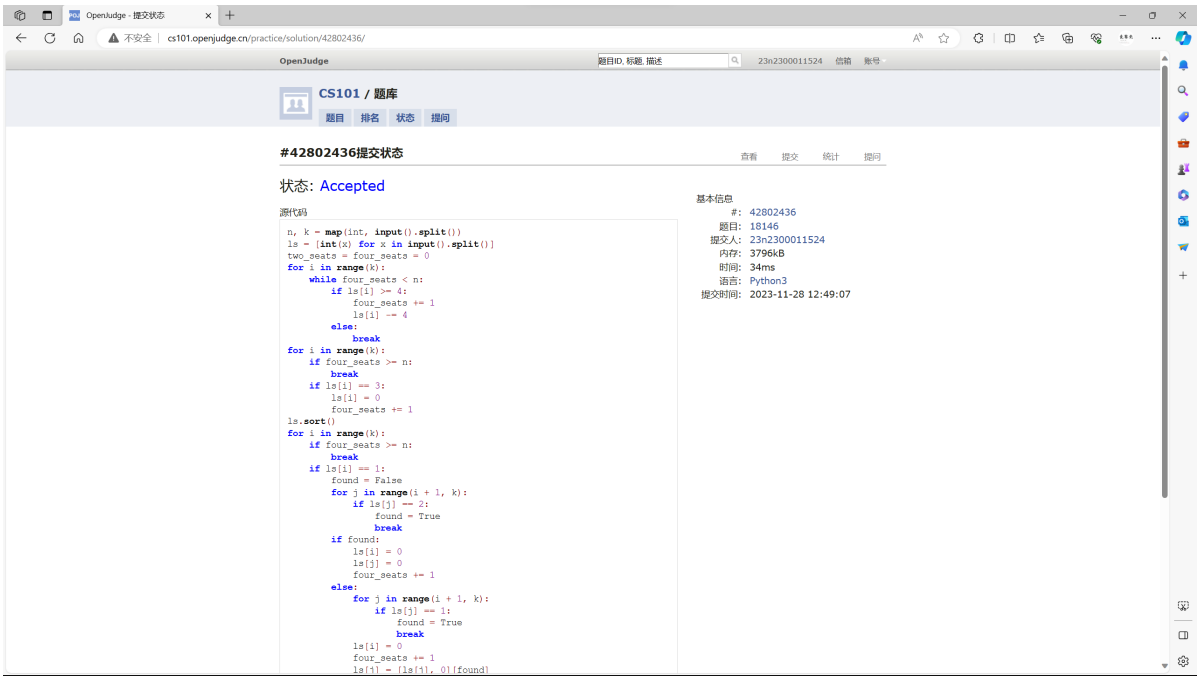
$$C-C-0-B$$

代码

```
n, k = map(int, input().split())
ls = [int(x) for x in input().split()]
two_seats = four_seats = 0
for i in range(k):
    while four_seats < n:
        if ls[i] >= 4:
            four_seats += 1
            ls[i] -= 4
        else:
            break
for i in range(k):
    if four_seats >= n:
        break
    if ls[i] == 3:
        ls[i] = 0
        four_seats += 1
ls.sort()
```

```
for i in range(k):
    if four_seats >= n:
        break
    if ls[i] == 1:
        found = False
        for j in range(i + 1, k):
            if ls[j] == 2:
                found = True
                break
        if found:
            ls[i] = 0
            ls[j] = 0
            four_seats += 1
        else:
            for j in range(i + 1, k):
                if ls[j] == 1:
                    found = True
                    break
            ls[i] = 0
            four_seats += 1
            ls[j] = [ls[j], 0][found]
for i in range(k):
    if four_seats >= n:
        break
    if ls[i] == 2:
        ls[i] = 0
        if n - four_seats >= 2 and ls.count(2) >= 3:
            ls[ls.index(2)] = 0
            ls[ls.index(2)] = 0
            four_seats += 2
        else:
            four_seats += 1
for i in range(k):
    two_seats += ls[i] // 2 + ls[i] % 2
if four_seats <= n and two_seats <= 2 * n:
    print('YES')
else:
    print('NO')
```

代码运行截图



OJ02287: 田忌赛马

greedy, http://cs101.openjudge.cn/practice/02287

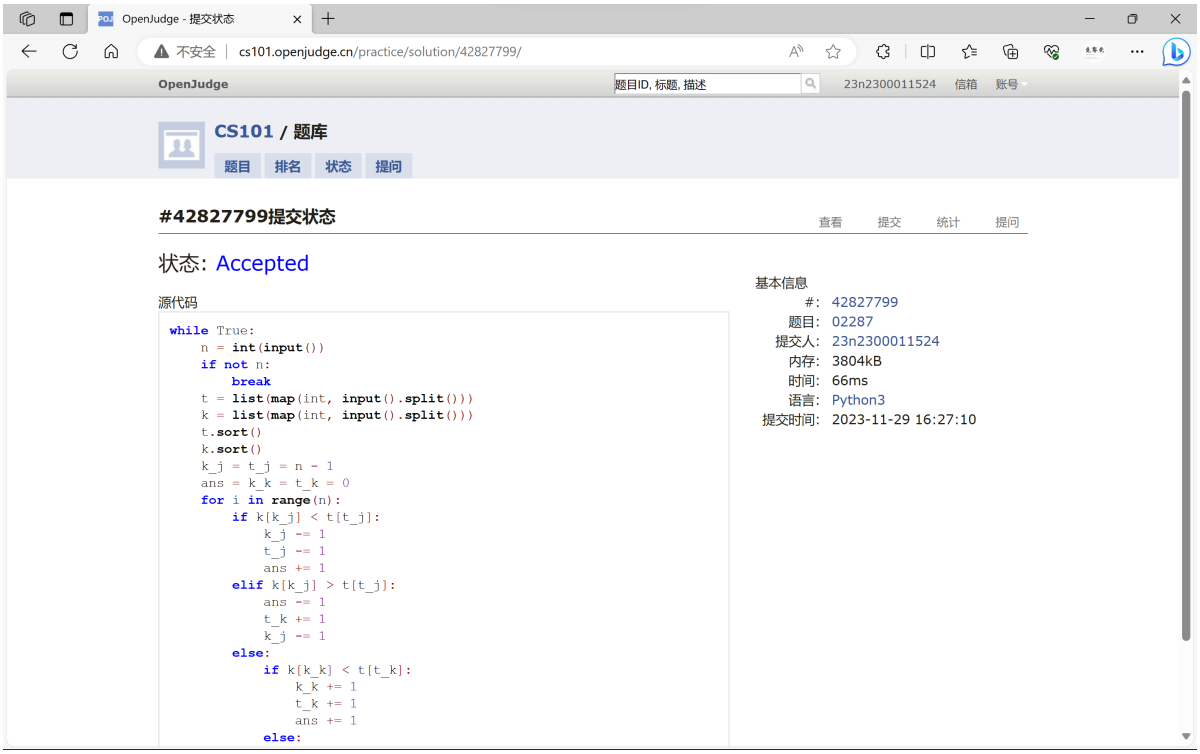
思路：贪心思路也可以理解为尽可能使田忌每个棋子能在其被使用的那个对局获胜，若否，让更小的棋子平局或失败以保留更大的棋子。

代码

```
while True:
    n = int(input())
    if not n:
        break
    t = list(map(int, input().split()))
    k = list(map(int, input().split()))
    t.sort()
    k.sort()
    k_j = t_j = n - 1
    ans = k_k = t_k = 0
    for i in range(n):
        if k[k_j] < t[t_j]:
            k_j -= 1
            t_j -= 1
            ans += 1
        elif k[k_j] > t[t_j]:
            ans -= 1
            t_k += 1
            k_j -= 1
        else:
            if k[k_k] < t[t_k]:
                k_k += 1
                t_k += 1
```

```
                ans += 1
            else:
                ans -= (k[k_j] > t[t_k])
                k_j -= 1
                t_k += 1
    print(ans*200)
```

代码运行截图



2. 学习总结和收获

田忌赛马理解了一会贪心策略的正确性，乌鸦坐飞机是曾经看题解才想起来的特殊情况，其余题目难度不大，注意审题即可。

截至2023年11月29日，OJ完成题目122道，CF完成题目49道。