

# Assignment #C: 矩阵、递归、贪心、和dfs similar

---

Updated 1126 GMT+8 Nov 28, 2023

2023 fall, Compiled by 同学的姓名、院系

## 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

(请改为同学的操作系统、编程环境等)

操作系统: macOS Ventura 13.4.1 (c)

Python编程环境: Spyder IDE 5.2.2, PyCharm 2023.1.4 (Professional Edition)

C/C++编程环境: Mac terminal vi (version 9.0.1424), g++/gcc (Apple clang version 14.0.3, clang-1403.0.22.14.1)

## 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:

四个字符一组进行比对

## 代码

```
1  '''
2  刘思瑞 2100017810
3  '''
4  num = int(input())
5  for k in range(num):
6      m = []
7      sum = 0
8      n = int(input())
9      for i in range(n):
10         s = input()
11         temp = []
12         for j in range(n):
13             temp.append(s[j])
14         m.append(temp)
15     for i in range(n//2):
16         for j in range(n//2):
17             t = max(m[i][j],m[n-1-i][n-1-j],m[j][n-1-i],m[n-1-j][i])
18             sum += 4*ord(t) - ord(m[i][j])-ord(m[n-1-i][n-1-j])-ord(m[j][n-
19             1-i]) - ord(m[n-1-j][i])
20         print(sum)
```

## 代码运行截图

By meinvader, contest: Codeforces Round 903 (Div. 3), problem: (C) Perfect Square, **Accepted**, #, [Copy](#)

```
'''
刘思瑞 2100017810
'''
num = int(input())
for k in range(num):
    m = []
    sum = 0
    n = int(input())
    for i in range(n):
        s = input()
        temp = []
        for j in range(n):
            temp.append(s[j])
        m.append(temp)
    for i in range(n//2):
        for j in range(n//2):
            t = max(m[i][j],m[n-1-i][n-1-j],m[j][n-1-i],m[n-1-j][i])
            sum += 4*ord(t) - ord(m[i][j])-ord(m[n-1-i][n-1-j])-ord(m[j][n-1-i]) - ord(m[n-1-j][i])
    print(sum)
```

## OJ02694: 波兰表达式

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

思路：

直接递归，但是学到了直接计算表达式的函数

## 代码

```
1  '''
2  刘思瑞 2100017810
3  '''
4  def calcu(calculate,i,j):
5      global calcul1
6      if calculate[i+1] not in calcul1:
7          if calculate[i+2] not in calcul1:
8              calculate[i] =
9              str(eval(calculate[i+1]+calculate[i]+calculate[i+2]))
10             del calculate[i+1]
11             del calculate[i+1]
12             i = j[-1]
13             j = j[:-1]
14         else:
15             j.append(i)
16             i = i+2
17     else:
18         j.append(i)
19         i = i+1
20     return calculate, i, j
21
22 calcul1 = ['+', '-', '*', '/']
23 calculate = list(input().split())
24 i = 0
25 j = [0]
26 while True:
27     calculate, i ,j = calcu(calculate,i,j)
28     if len(calculate) == 1:
29         break
30 print('%.6f' % float(calculate[0]))
```

代码运行截图

状态: Accepted

源代码

```
'''
刘思瑞 2100017810
'''
def calcu( calculate, i, j ):
    global calcul1
    if calculate[i+1] not in calcul1:
        if calculate[i+2] not in calcul1:
            calculate[i] = str(eval( calculate[i+1]+calculate[i]+calculate[i+2] ))
            del calculate[i+1]
            del calculate[i+1]
            i = j[-1]
            j = j[:-1]
        else:
            j.append(i)
            i = i+2
    else:
        j.append(i)
        i = i+1
    return calculate, i, j

calcul1 = ['+', '-', '*', '/']
calculate = list(input().split())
i = 0
j = [0]
while True:
    calculate, i, j = calcu( calculate, i, j )
    if len( calculate ) == 1:
        break
print( '%.6f' % float( calculate[0] ) )
```

## OJ18160: 最大连通域面积

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

思路:

递归搜索

代码

```
1  '''
2  刘思瑞 2100017810
3  '''
4  m, flag, N, M, summ = [], [], 0, 0, 0
5  def search( i, j ):
6      global m, flag, N, M, summ
7      if i != 0:
8          if ( (flag[i-1][j] == True) and (m[i-1][j] == 'w') ):
9              summ += 1
10             flag[i-1][j] = False
11             search( i-1, j )
12             if ( (flag[i-1][j+1] == True) and (m[i-1][j+1] == 'w') ):
13                 summ += 1
14                 flag[i-1][j+1] = False
15                 search( i-1, j+1 )
16             if j != 0:
```

```

17         if ((flag[i-1][j-1] == True) and (m[i-1][j-1] == 'w')):
18             summ += 1
19             flag[i-1][j-1] = False
20             search(i-1,j-1)
21     if ((flag[i][j+1] == True) and (m[i][j+1] == 'w')):
22         summ += 1
23         flag[i][j+1] = False
24         search(i,j+1)
25     if ((flag[i+1][j+1] == True) and (m[i+1][j+1] == 'w')):
26         summ += 1
27         flag[i+1][j+1] = False
28         search(i+1,j+1)
29     if ((flag[i+1][j] == True) and (m[i+1][j] == 'w')):
30         summ += 1
31         flag[i+1][j] = False
32         search(i+1,j)
33     if j != 0:
34         if ((flag[i][j-1] == True) and (m[i][j-1] == 'w')):
35             summ += 1
36             flag[i][j-1] = False
37             search(i,j-1)
38         if ((flag[i+1][j-1] == True) and (m[i+1][j-1] == 'w')):
39             summ += 1
40             flag[i+1][j-1] = False
41             search(i+1,j-1)
42     return
43
44
45 num = int(input())
46 for k in range(num):
47     m = []
48     flag = []
49     sum = 0
50     N,M = map(int,input().split())
51     for i in range(N):
52         flag.append([True]*(M)+[False])
53         s = input()
54         temp = []
55         for j in range(M):
56             temp.append(s[j])
57         temp.append('.')
58         m.append(temp)
59     m.append(['.']*(M+1))
60     flag.append([False]*(M+1))
61     for i in range(N):
62         for j in range(M):
63             if m[i][j] == 'w' and flag[i][j] == True:
64                 summ = 1
65                 flag[i][j] = False
66                 search(i,j)
67             sum = max(sum,summ)
68     print(sum)

```

状态: Accepted

源代码

```
'''
刘思瑞 2100017810
'''
m, flag, N, M, summ = [], [], 0, 0, 0
def search(i, j):
    global m, flag, N, M, summ
    if i != 0:
        if ((flag[i-1][j] == True) and (m[i-1][j] == 'W')):
            summ += 1
            flag[i-1][j] = False
            search(i-1, j)
        if ((flag[i-1][j+1] == True) and (m[i-1][j+1] == 'W')):
            summ += 1
            flag[i-1][j+1] = False
            search(i-1, j+1)
    if j != 0:
        if ((flag[i-1][j-1] == True) and (m[i-1][j-1] == 'W')):
            summ += 1
            flag[i-1][j-1] = False
            search(i-1, j-1)
    if ((flag[i][j+1] == True) and (m[i][j+1] == 'W')):
        summ += 1
        flag[i][j+1] = False
        search(i, j+1)
    if ((flag[i+1][j+1] == True) and (m[i+1][j+1] == 'W')):
        summ += 1
        flag[i+1][j+1] = False
        search(i+1, j+1)
    if ((flag[i+1][j] == True) and (m[i+1][j] == 'W')):
        summ += 1
        flag[i+1][j] = False
        search(i+1, j)
    if j != 0:
        if ((flag[i][j-1] == True) and (m[i][j-1] == 'W')):
            summ += 1
            flag[i][j-1] = False
```

## OJ02754: 八皇后

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

思路:

其实可以遍历到指定的元素最大值就停止

不过有点太麻烦了感觉没必要就不写了

## 代码

```
1  '''
2  刘思瑞 2100017810
3  '''
4  def search(queen,i):
5      global ans
6      if i == 8:
7          s=''
8          for i in queen:
9              s += str(i)
10             ans.append(int(s))
11             return
12     rest = [1,2,3,4,5,6,7,8]
13     for j in range(i):
14         for _ in [queen[j],queen[j]+i-j,queen[j]-i+j]:
15             if _ in rest:
16                 rest.remove(_)
17     for j in rest:
18         search(queen+[j],i+1)
19
20 ans = []
21 search([],0)
22 num = int(input())
23 for i in range(num):
24     print(ans[int(input())-1])
```

代码运行截图

状态: Accepted

源代码

```
'''
刘思瑞 2100017810
'''
def search(queen,i):
    global ans
    if i == 8:
        s=''
        for i in queen:
            s += str(i)
        ans.append(int(s))
        return
    rest = [1,2,3,4,5,6,7,8]
    for j in range(i):
        for _ in [queen[j],queen[j]+i-j,queen[j]-i+j]:
            if _ in rest:
                rest.remove(_)
    for j in rest:
        search(queen+[j],i+1)

ans = []
search([],0)
num = int(input())
for i in range(num):
    print(ans[int(input())-1])
```

## 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:

确实是贪心但是要考虑的太多了, 大概就分为三种箱子, 一种是4, 一种是2, 还有一种是4里面装了2变成1, 在这些情况中有时候是要考虑用空间换类别的因此会非常麻烦

代码

```
1 '''
2 刘思瑞 2100017810
3 '''
4 n,k,crow,large,small,flag,temp = 0,0,[],0,0,False,0
5 def echo(d):
6     global n,k,crow,large,small,flag,temp
7     if d ==0:
8         return
9     if d==1:
10         if temp:
11             temp -= 1
```



```

12         return
13     else:
14         if large:
15             large, small = large-1, small+1
16             return
17         elif small:
18             small-=1
19             return
20     flag = False
21     return
22 if d ==2:
23     if small:
24         small -= 1
25         return
26     else:
27         if temp>=2:
28             temp-=2
29             return
30         if large:
31             if temp:
32                 large -=1
33                 small+=1
34                 temp -=1
35                 return
36             else:
37                 large, temp = large-1, temp+1
38                 return
39         if temp>=2:
40             temp-=2
41             return
42     flag = False
43     return
44 if d ==3:
45     if large:
46         large -= 1
47         return
48     else:
49         if temp:
50             if small:
51                 temp -=1
52                 small-=1
53                 return
54             else:
55                 if small:
56                     if large:
57                         small-=1
58                         large-=1
59                         temp+=1
60                     elif small>=2:
61                         small-=2
62                         return
63     flag = False
64     return
65 if d == 4:
66     if large:
67         large -= 1

```

```
68         return
69     else:
70         if small >= 2:
71             small -= 2
72             return
73     else:
74         if small:
75             if temp >= 2:
76                 small -= 1
77                 temp -= 2
78                 return
79             elif temp >= 4:
80                 temp -= 4
81                 return
82     flag = False
83     return
84
85 def define(c):
86     global n, k, crow, large, small, flag, temp
87     for i in range(k):
88         rest = crow[i] // 4
89         for j in range(rest):
90             echo(4)
91             echo(crow[i] % 4)
92             if not flag:
93                 return 'NO'
94     return 'YES'
95
96 n, k = map(int, input().split())
97 crow = list(map(int, input().split()))
98 small = 2 * n
99 large = n
100 temp = 0
101 flag = True
102 crow.sort()
103 print(define(1))
```

代码运行截图

状态: Accepted

源代码

```
'''
刘思瑞 2100017810
'''
n,k,crow,large,small,flag,temp = 0,0,[],0,0,False,0
def echo(d):
    global n,k,crow,large,small,flag,temp
    if d==0:
        return
    if d==1:
        if temp:
            temp -= 1
            return
        else:
            if large:
                large,small = large-1,small+1
                return
            elif small:
                small-=1
                return
            flag = False
            return
    if d==2:
        if small:
            small -= 1
            return
        else:
            if temp>=2:
                temp-=2
                return
```

## OJ02287: 田忌赛马

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

思路:

按小马来比较

代码

```
1 '''
2 刘思瑞 2100017810
3 '''
4 def money(tian,king,num):
5     tian.sort(reverse=True)
6     king.sort(reverse=True)
7     sum = 0
8     while True:
9         if tian == []:
```

```
10         print(sum*200)
11         return
12     besttian = tian[0]
13     worsetian = tian[-1]
14     bestking = king[0]
15     worseking = king[-1]
16     if worsetian > worseking:
17         tian,king = tian[:-1],king[:-1]
18         sum+=1
19         continue
20     if worsetian < worseking:
21         tian,king = tian[:-1],king[1:]
22         sum-=1
23         continue
24     if worseking == worsetian:
25         if bestking < besttian:
26             tian,king = tian[:-1],king[:-1]
27         elif bestking > besttian:
28             tian,king = tian[:-1],king[1:]
29             sum-=1
30         else:
31             if worsetian < bestking:
32                 sum-=1
33             tian,king = tian[:-1],king[1:]
34         continue
35
36 while True:
37     n = int(input())
38     if not n:
39         break
40     tian = list(map(int,input().split()))
41     king = list(map(int,input().split()))
42     money(tian,king,n)
```

代码运行截图

状态: Accepted

源代码

```
'''
刘思瑞 2100017810
'''
def money(tian, king, num):
    tian.sort(reverse=True)
    king.sort(reverse=True)
    sum = 0
    while True:
        if tian == []:
            print(sum*200)
            return
        besttian = tian[0]
        worsetian = tian[-1]
        bestking = king[0]
        worseking = king[-1]
        if worsetian > worseking:
            tian, king = tian[:-1], king[:-1]
            sum+=1
            continue
        if worsetian < worseking:
            tian, king = tian[:-1], king[1:]
            sum-=1
            continue
        if worseking == worsetian:
            if bestking < besttian:
                tian, king = tian[:-1], king[:-1]
            elif bestking > besttian:
                tian, king = tian[:-1], king[1:]
            sum-=1
```

com...

## 2. 学习总结和收获

感觉这次作业主要是debug非常痛苦，特别是乌鸦坐飞机，实在是对着参考数据改的