Assignment #C: 矩阵、递归、贪心、和dfs simlar

Updated 1700 GMT+8 Nov 28, 2023

2023 fall, Complied by 钟明衡 物理学院

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_NT x64 10.0.19045

Python编程环境：Visual Studio Code 1.76.1

C/C++编程环境：Visual Studio Code 1.76.1

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：

把字母转化成数字存到列表里面，然后对于每组$0 \leqslant i,j < \frac{n}{2}$，找出对应的四个位置$(i,j),(j,n-i-1),(n-i-1,n-j-1),(n-j-1,i)$上最大的那个数，用这四个数最大的一个的4倍减去这四个数的和，加到答案中即可

代码

```python
def num(s):
    global dic
    return dic[s]


dic = {'a': 0, 'b': 1, 'c': 2, 'd': 3, 'e': 4, 'f': 5, 'g': 6, 'h': 7, 'i': 8, 'j': 9, 'k': 10, 'l': 11, 'm': 12, 'n': 13, 'o': 14, 'p': 15, 'q': 16, 'r': 17, 's': 18, 't': 19, 'u': 20, 'v': 21, 'w': 22, 'x': 23, 'y': 24, 'z': 25}
t = int(input())
anss = []
for _ in range(t):
    n = int(input())
    l = []
    ans = 0
    for i in range(n):
        l.append(tuple(map(num, input())))
    for i in range(n//2):
        for j in range(n//2):
            ans += 4*max(l[i][j], l[j][n-i-1], l[n-i-1][n-j-1], l[n-j-1][i]) - \
                (l[i][j] + l[j][n-i-1] + l[n-i-1][n-j-1] + l[n-j-1][i])
    anss.append(ans)
for ans in anss:
    print(ans)


```

代码运行截图

General

#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged		
234663730	Practice: MinghengZhong	1881C - 20	Python 3	Accepted	436 ms	8684 KB	2023-11-28 08:01:45	2023-11-28 08:01:45		Compare

→ Source

Copy

```python
def num(s):
    global dic
    return dic[s]


dic = {'a': 0, 'b': 1, 'c': 2, 'd': 3, 'e': 4, 'f': 5, 'g': 6, 'h': 7, 'i': 8, 'j': 9, 'k': 10, 'l': 11, 'm': 12,
       'n': 13, 'o': 14, 'p': 15, 'q': 16, 'r': 17, 's': 18, 't': 19, 'u': 20, 'v': 21, 'w': 22, 'x': 23, 'y': 24, 'z': 25}
t = int(input())
anss = []
for _ in range(t):
    n = int(input())
    l = []
    ans = 0
    for i in range(n):
        l.append(tuple(map(num, input())))
    for i in range(n//2):
        for j in range(n//2):
            ans += 4*max(l[i][j], l[j][n-i-1], l[n-i-1][n-j-1], l[n-j-1][i]) - \
                (l[i][j] + l[j][n-i-1] + l[n-i-1][n-j-1] + l[n-j-1][i])
    anss.append(ans)
for ans in anss:
    print(ans)
```

OJ02694: 波兰表达式

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

思路：

采用递归，判断如下：

如果某位置是数，直接返回这个数

如果某个位置上是符号，那么就返回运算结果，运算需要的两个数，一个就是下一位，另一个是最先出现的“同级”位

“级”这样判断：初始是1，符号$+1$，数字-1，为0时返回

代码

```
s = []


def next(m):
    global s
    count = 1
    for i in range(m+1, len(s)):
        if count == 0:
            return i
        if s[i] == '+' or s[i] == '-' or s[i] == '*' or s[i] == '/':
            count += 1
        else:
            count -= 1


def poland(n):
    global s
    if s[n] == '+':
        return poland(n+1)+poland(next(n))
    elif s[n] == '-':
        return poland(n+1)-poland(next(n))
    elif s[n] == '*':
        return poland(n+1)*poland(next(n))
    elif s[n] == '/':
        return poland(n+1)/poland(next(n))
    else:
        return float(s[n])


s = input().split()
print('%.6f' % (poland(0)))

```

代码运行截图

#41616605提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
s = []


def next(m):
    global s
    count = 1
    for i in range(m+1, len(s)):
        if count == 0:
            return i
        if s[i] == '+' or s[i] == '-' or s[i] == '*' or s[i] == '/':
            count += 1
        else:
            count -= 1


def poland(n):
    global s
    if s[n] == '+':
        return poland(n+1)+poland(next(n))
    elif s[n] == '-':
        return poland(n+1)-poland(next(n))
    elif s[n] == '*':
        return poland(n+1)*poland(next(n))
    elif s[n] == '/':
        return poland(n+1)/poland(next(n))
    else:
        return float(s[n])


s = input().split()
print('%.6f' % (poland(0)))
```

基本信息

#: 41616605
题目: 02694
提交人: 23n2300011505(12号娱乐选手)
内存: 3616kB
时间: 34ms
语言: Python3
提交时间: 2023-10-12 23:31:24

OJ18160: 最大连通域面积

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

思路:

找到没用过的'W'点，从这个点开始dfs

搜周围一圈，如果是'W'就标记为搜过，然后从新的这个点开始搜，同时记录搜到了多少'W'

最后输出结果即可

代码

```
def dfs(i, j):
    global l, used
    count = 1
    for x in range(i-1, i+2):
        for y in range(j-1, j+2):
            if l[x][y] == 'W' and not used[x][y]:
                used[x][y] = True
                count += dfs(x, y)
```

```
    return count


t = int(input())
anss = []
for _ in range(t):
    n, m = map(int, input().split())
    used = [[False]*(m+2) for i in range(n+2)]
    l = ['.'*(m+2)]
    ans = 0
    for i in range(n):
        l.append('.'+input()+'.')
    l.append('.'*(m+2))
    for i in range(1, n+1):
        for j in range(1, m+1):
            if l[i][j] == 'W' and not used[i][j]:
                used[i][j] = True
                ans = max(ans, dfs(i, j))
    anss.append(ans)
for ans in anss:
    print(ans)

```

代码运行截图

#42803212提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def dfs(i, j):
    global l, used
    count = 1
    for x in range(i-1, i+2):
        for y in range(j-1, j+2):
            if l[x][y] == 'W' and not used[x][y]:
                used[x][y] = True
                count += dfs(x, y)
    return count


t = int(input())
anss = []
for _ in range(t):
    n, m = map(int, input().split())
    used = [[False]*(m+2) for i in range(n+2)]
    l = ['.'*(m+2)]
    ans = 0
    for i in range(n):
        l.append('.'+input()+'.')
    l.append('.'*(m+2))
    for i in range(1, n+1):
        for j in range(1, m+1):
            if l[i][j] == 'W' and not used[i][j]:
                used[i][j] = True
                ans = max(ans, dfs(i, j))
    anss.append(ans)
for ans in anss:
    print(ans)
```

基本信息

#: 42803212
题目: 18160
提交人: 23n2300011505(12号娱乐选手)
内存: 3696kB
时间: 93ms
语言: Python3
提交时间: 2023-11-28 13:38:29

OJ02754: 八皇后

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

思路：

直接枚举得到出92种情况，存起来再输出就可以

代码

```
count = 0
ans = []


def dp(i, a, b, c, m):
    global ans, count
    if i == 8:
        count += 1
        ans.append(m)
    else:
        for j in range(8):
            if a[j] and b[i+j] and c[i-j+7]:
                a[j] = False
                b[i+j] = False
                c[i-j+7] = False
                m += str(j+1)
                dp(i+1, a, b, c, m)
                m = m[:-1]
                c[i-j+7] = True
                b[i+j] = True
                a[j] = True
    return


dp(0, [True]*8, [True]*15, [True]*15, '')
t = int(input())
for _ in range(t):
    print(ans[int(input())-1])

```

代码运行截图

#42803453提交状态 查看 提交 统计 提问

状态: Accepted

源代码

```
count = 0
ans = []


def dp(i, a, b, c, m):
    global ans, count
    if i == 8:
        count += 1
        ans.append(m)
    else:
        for j in range(8):
            if a[j] and b[i+j] and c[i-j+7]:
                a[j] = False
                b[i+j] = False
                c[i-j+7] = False
                m += str(j+1)
                dp(i+1, a, b, c, m)
                m = m[:-1]
                c[i-j+7] = True
                b[i+j] = True
                a[j] = True
    return


dp(0, [True]*8, [True]*15, [True]*15, '')
t = int(input())
for _ in range(t):
    print(ans[int(input())-1])
```

基本信息

#: 42803453
题目: 02754
提交人: 23n2300011505(12号娱乐选手)
内存: 3664kB
时间: 38ms
语言: Python3
提交时间: 2023-11-28 13:51:15

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：

优先装中间的四个位置，先用同一窝的4只来填满，然后再填同一窝的3只

如果中间位置已经超出了，就从旁边的两个位置里面扣除

这样以后，每窝还可能剩下1或2只，优先把2只的放到旁边，剩下的1只和2只组成一组放到中间

如果最后只剩下2只的，注意到可以把2只拆成2个1只，则每有3组2只，就只需要2个中间位置

最后判断位置是否有超出即可

代码

```
n, l = map(int, input().split())
l = list(map(int, input().split()))
t = n*2
c = [0, 0, 0, 0]
ans = True
for i in l:
    n -= i//4
```

```
    c[i % 4] += 1
n -= c[3]
if n < 0:
    t += 2*n
    n = 0
if t < 0:
    ans = False
elif t > c[2]:
    c[1] = max(c[1]-t+c[2], 0)
    c[2] = 0
else:
    c[2] -= t
a = abs(c[1]-c[2])
if c[1] > c[2]:
    if c[2]+a//2+a % 2 > n:
        ans = False
else:
    if c[1]+(a//3)*2+a % 3 > n:
        ans = False
if ans:
    print('YES')
else:
    print('NO')

```

代码运行截图

#42813832提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
n, l = map(int, input().split())
l = list(map(int, input().split()))
t = n*2
c = [0, 0, 0, 0]
ans = True
for i in l:
    n -= i//4
    c[i % 4] += 1
n -= c[3]
if n < 0:
    t += 2*n
    n = 0
if t < 0:
    ans = False
elif t > c[2]:
    c[1] = max(c[1]-t+c[2], 0)
    c[2] = 0
else:
    c[2] -= t
a = abs(c[1]-c[2])
if c[1] > c[2]:
    if c[2]+a//2+a % 2 > n:
        ans = False
else:
    if c[1]+(a//3)*2+a % 3 > n:
        ans = False
if ans:
    print('YES')
else:
    print('NO')
```

基本信息

#: 42813832
题目: 18146
提交人: 23n2300011505(12号娱乐选手)
内存: 3904kB
时间: 32ms
语言: Python3
提交时间: 2023-11-28 20:34:22

OJ02287: 田忌赛马

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

思路:

首先将马从快到慢排序，然后贪心，策略如下:

如果田忌最快的马比齐王最快的马快，就都用最快的马赢一次

如果田忌最慢的马比齐王最慢的马快，就都用最慢的马赢一次

如果都不行，就拿田忌最慢的马和齐王最快的马比

代码

```
while True:
    n = int(input())
    if n == 0:
        break
    l1 = sorted(list(map(int, input().split())), reverse=True)
    l2 = sorted(list(map(int, input().split())), reverse=True)
    win, i1, i2, j1, j2 = 0, 0, 0, n-1, n-1
    while (i1 <= j1):
        if l1[i1] > l2[i2]:
```

```
            i1 += 1
            i2 += 1
            win += 1
        else:
            if l1[j1] > l2[j2]:
                j1 -= 1
                j2 -= 1
                win += 1
            else:
                if l1[j1] < l2[i2]:
                    win -= 1
                j1 -= 1
                i2 += 1
    print(win*200)

```

代码运行截图

#42806436提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
while True:
    n = int(input())
    if n == 0:
        break
    l1 = sorted(list(map(int, input().split())), reverse=True)
    l2 = sorted(list(map(int, input().split())), reverse=True)
    win, i1, i2, j1, j2 = 0, 0, 0, n-1, n-1
    while (i1 <= j1):
        if l1[i1] > l2[i2]:
            i1 += 1
            i2 += 1
            win += 1
        else:
            if l1[j1] > l2[j2]:
                j1 -= 1
                j2 -= 1
                win += 1
            else:
                if l1[j1] < l2[i2]:
                    win -= 1
                j1 -= 1
                i2 += 1
    print(win*200)
```

基本信息

#: 42806436
题目: 02287
提交人: 23n2300011505(12号娱乐选手)
内存: 3880kB
时间: 65ms
语言: Python3
提交时间: 2023-11-28 15:53:49

2. 学习总结和收获

对比之前写过的题目，感觉大体上dfs比dp简单，dp比贪心简单

感觉dfs大部分是计算机思维，dp数学+计算机（因为要想递推式），而贪心基本上是纯数学