Assignment #B: 图论和树算

Updated 1709 GMT+8 Apr 28, 2024

2024 spring, Complied by 尹柚鑫 2100015878 光华管理学院

说明：

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），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"区有上传的md或者doc附件。

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：Win11

Python编程环境：jupter notebook

1. 题目

28170: 算鹰

dfs, http://cs101.openjudge.cn/practice/28170/

耗时：15mins

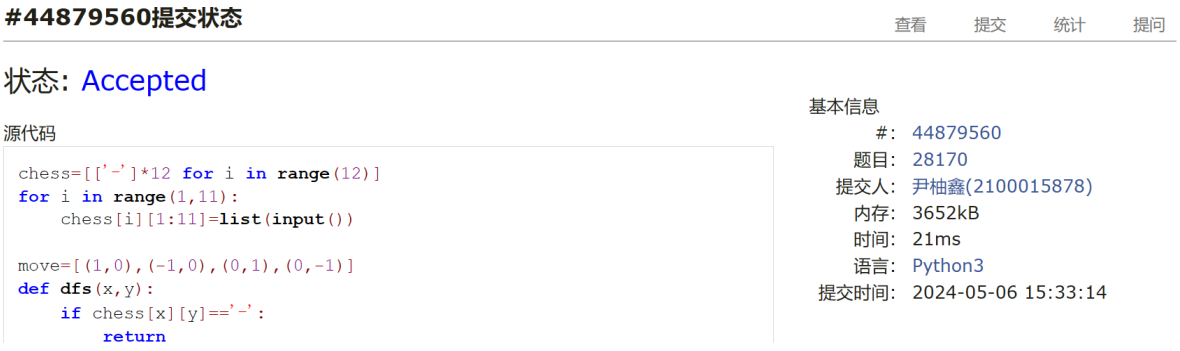
思路：和之前作业中求最大权值连通块的思路基本一样。dfs做遍历就可以

代码

```
#
chess=[['-']*12 for i in range(12)]
for i in range(1,11):
    chess[i][1:11]=list(input())

move=[(1,0),(-1,0),(0,1),(0,-1)]
def dfs(x,y):
    if chess[x][y]=='-':
        return
    chess[x][y]='-'
    for i in range(4):
        newx=x+move[i][0]
        newy=y+move[i][1]
        dfs(newx,newy)
```

```
res=0
for i in range(12):
    for j in range(12):
        if chess[i][j]=='.':
            res+=1
            dfs(i,j)

print(res)
```

代码运行截图 ==（至少包含有"Accepted"）==



02754: 八皇后

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

耗时：40mins

思路：参考了答案的第一种写法，由于这个棋盘是左右对称的，可以稍微优化一点点，只计算前46个答案即可，后46个可以由前46个转化得来

代码

```
#
answer = []

def Queen(s):
    if len(s)==0:
        for col in range(1, 5):
            Queen(s + str(col))
    else:
        for col in range(1, 9):
            for j in range(len(s)):
                if (str(col) == s[j] or # 两个皇后不能在同一列
                        abs(col - int(s[j])) == abs(len(s) - j)): # 两个皇后不能在同一斜线
                    break
            else:
                if len(s) == 7:
                    answer.append(s + str(col))
                else:
                    Queen(s + str(col))
```

```
Queen('')

n = int(input())
for _ in range(n):
    a = int(input())
    if a<=46:
        print(answer[a - 1])
    else:
        output=''
        for i in str(answer[92-a]):
            output+=str(9-int(i))
        print(int(output))
```

代码运行截图 ==（至少包含有"Accepted"）==

03151: Pots

bfs, http://cs101.openjudge.cn/practice/03151/

耗时：40mins

思路：和词梯问题，找最短走迷宫路径问题基本一样

词梯问题，之前使用结点的属性，来记录当前的距离和上一个点是谁

这里我用的是在一个词典记录这些属性的方法

这个题花了我很长时间，因为一开始我有些忘记BFS怎么写，并没有用stack来存储下面要操作的点。这就会导致程序在一个方向上一直深挖，递归深度超。使用了stack存储下面的操作点，才实际上是实现了BFS，考虑每一步所能到达的位置。这个错误也让我更好地理解了BFS的实现原理

代码

```
#
def fill(position,target):
    if target==0:
        position[0]=max_a
    else:
        position[1]=max_b
    return position
```

```
def drop(position,target):
    if target==0:
        position[0]=0
    else:
        position[1]=0
    return position

def pour(position,target):#从target这个杯子倒出去
    if target==0:
        newx=max(0,position[0]-max_b+position[1])
        newy=min(max_b,position[0]+position[1])
        position[0]=newx
        position[1]=newy
    else:
        newy=max(0,position[1]-max_a+position[0])
        newx=min(max_a,position[0]+position[1])
        position[0]=newx
        position[1]=newy
    return position

def operation(now_a,now_b,purpose):

    chess[now_a][now_b]=1
    queue.append((now_a,now_b))

    while queue:
        now_a,now_b=queue.pop(0)
        #print(now_a,now_b)
        chess[now_a][now_b]=1

        for i in range(6):
            if i==0:
                new_a,new_b=drop([now_a,now_b],0)
                if chess[new_a][new_b]==0:
                    path[(new_a,new_b)]=['DROP(1)',
(now_a,now_b),path[(now_a,now_b)][2]+1]
                    queue.append((new_a,new_b))
            elif i==1:
                new_a,new_b=drop([now_a,now_b],1)
                if chess[new_a][new_b]==0:
                    path[(new_a,new_b)]=['DROP(2)',
(now_a,now_b),path[(now_a,now_b)][2]+1]
                    queue.append((new_a,new_b))
            elif i==2:
                new_a,new_b=fill([now_a,now_b],0)
                if chess[new_a][new_b]==0:
                    path[(new_a,new_b)]=['FILL(1)',
(now_a,now_b),path[(now_a,now_b)][2]+1]
                    queue.append((new_a,new_b))
            elif i==3:
                new_a,new_b=fill([now_a,now_b],1)
                if chess[new_a][new_b]==0:
                    path[(new_a,new_b)]=['FILL(2)',
(now_a,now_b),path[(now_a,now_b)][2]+1]
                    queue.append((new_a,new_b))
```

```
            elif i==4:
                new_a,new_b=pour([now_a,now_b],0)
                if chess[new_a][new_b]==0:
                    path[(new_a,new_b)]=['POUR(1,2)',
(now_a,now_b),path[(now_a,now_b)][2]+1]
                    queue.append((new_a,new_b))
            elif i==5:
                new_a,new_b=pour([now_a,now_b],1)
                if chess[new_a][new_b]==0:
                    path[(new_a,new_b)]=['POUR(2,1)',
(now_a,now_b),path[(now_a,now_b)][2]+1]
                    queue.append((new_a,new_b))

max_a,max_b,max_c=map(int,input().split())
chess=[[0]*(max_b+1) for i in range(max_a+1)]

path={}
path[(0,0)]=['',None,0]
queue=[]

operation(0,0,max_c)

found=False

for key in path.keys():
    if max_c in key:
        found=True
        mykey=key
    if found==True:
         break

if found:
    passer=mykey
    print(path[passer][2])
    opera=[]
    while path[passer][1]!=None:
        opera.append(path[passer][0])
        passer=path[passer][1]
    for i in opera[::-1]:
        print(i)
else:
    print('impossible')
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44887882提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def fill(position,target):
    if target==0:
        position[0]=max_a
    else:
        position[1]=max_b
    return position

def drop(position,target):
```

基本信息

#: 44887882
题目: 03151
提交人: 尹柚鑫(2100015878)
内存: 3952kB
时间: 24ms
语言: Python3
提交时间: 2024-05-07 15:11:53

05907: 二叉树的操作

http://cs101.openjudge.cn/practice/05907/

耗时：30mins

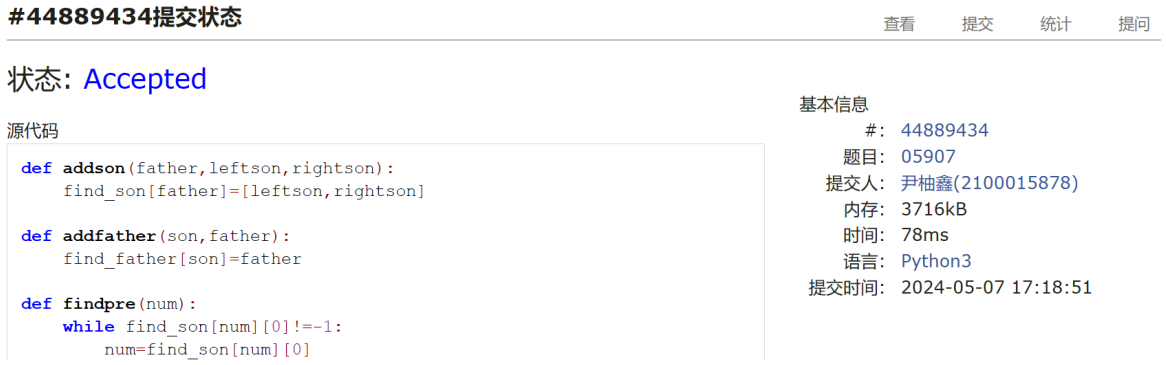
思路：使用一个儿子列表一个爸爸列表，建树。操作不难，一开始在交换的函数中，只更改了爸爸的儿子，没有更改儿子的爸爸，导致报错，看了很久才看出来

代码

```
#
def addson(father,leftson,rightson):
    find_son[father]=[leftson,rightson]

def addfather(son,father):
    find_father[son]=father

def findpre(num):
    while find_son[num][0]!=-1:
        num=find_son[num][0]
    return num

def exchange(num1,num2):
    father1=find_father[num1]
    num1_index=find_son[father1].index(num1)

    father2=find_father[num2]
    num2_index=find_son[father2].index(num2)

    find_son[father1][num1_index]=num2
    find_son[father2][num2_index]=num1
    find_father[num2]=father1
    find_father[num1]=father2

for _ in range(int(input())):
    find_son={}
    find_father={}
    n,m = map(int,input().split())
    for _ in range(n):
        father,leftson,rightson=map(int,input().split())
        addson(father,leftson,rightson)
        addfather(leftson,father)
```

```
        addfather(rightson,father)
    for _ in range(m):
        operation=list(map(int,input().split()))
        if operation[0]==2:
            num=findpre(operation[1])
            print(num)
        else:
            exchange(operation[1],operation[2])
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



18250: 冰阔落 I

Disjoint set, http://cs101.openjudge.cn/practice/18250/

思路:

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

05443: 兔子与樱花

http://cs101.openjudge.cn/practice/05443/

思路:

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

2. 学习总结和收获

==如果作业题目简单，有否额外练习题目，比如：OJ“2024spring每日选做”、CF、LeetCode、洛谷等网站题目。==

1.浅拷贝和深拷贝，浅拷贝是全部相互影响，深拷贝互不影响

import copy

ss=copy.copy(ii)

dd=copy.deepcopy(ii)

2.

在Python中，`else` 与 `for` 或 `while` 循环结合使用时，有特定的行为：

- `else` 块会在循环正常结束时执行，即没有遇到 `break` 语句导致的提前退出。
- 如果循环因为 `break` 而提前退出，则 `else` 块不会执行。

  for i in range(10):
  　　print(i)
  　　if i == 6:
  　　　　break
  else:
  　　print(9)

  这里，for循环过程中，会被break，因此就不会输出9，如果for循环中没有被break，就会输出9