Assignment #B: 图论和树算

Updated 1709 GMT+8 Apr 28, 2024

2024 spring, Complied by 城环 吴至超

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：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

28170: 算鹰

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

思路：同样理解错了题目，至于想了好一会为什么要用dfs

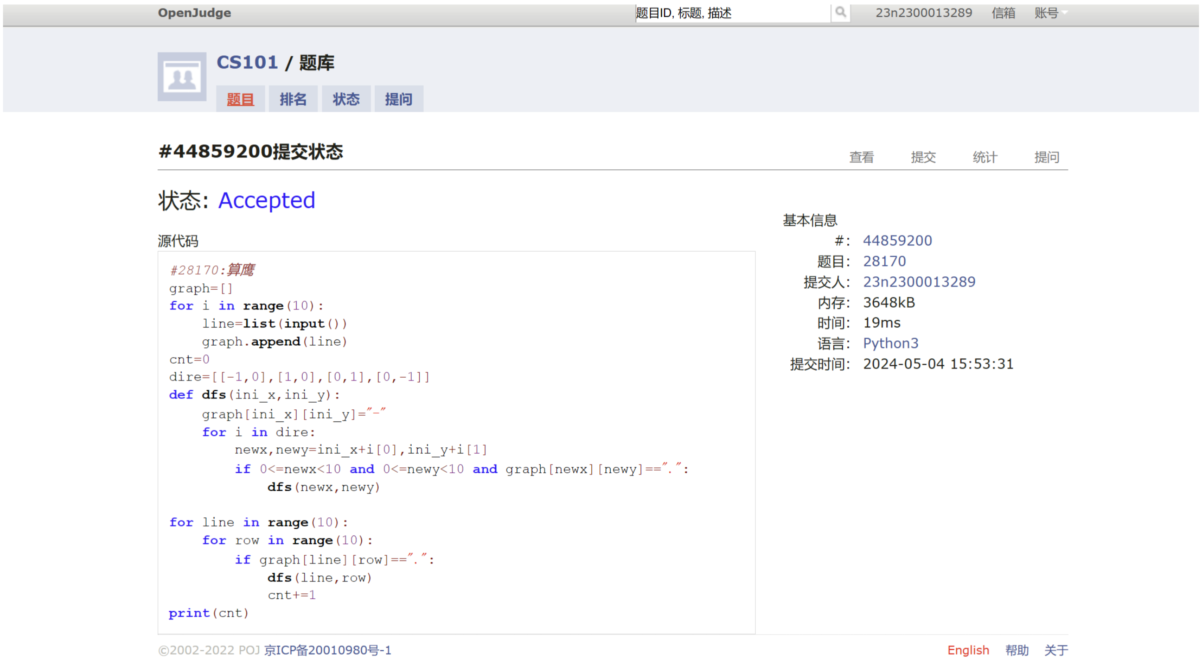
实际上，dfs的用处应该是清除一块联通的棋子吧

代码

```python
# #28170:算鹰
graph=[]
for i in range(10):
    line=list(input())
    graph.append(line)
cnt=0
dire=[[-1,0],[1,0],[0,1],[0,-1]]
```

```python
def dfs(ini_x,ini_y):
    graph[ini_x][ini_y]="-"
    for i in dire:
        newx,newy=ini_x+i[0],ini_y+i[1]
        if 0<=newx<10 and 0<=newy<10 and graph[newx][newy]==".":
            dfs(newx,newy)

for line in range(10):
    for row in range(10):
        if graph[line][row]==".":
            dfs(line,row)
            cnt+=1
print(cnt)
```

代码运行截图 == （至少包含有"Accepted"） ==



#44859200提交状态

状态: Accepted

源代码

```python
#28170:算鹰
graph=[]
for i in range(10):
    line=list(input())
    graph.append(line)
cnt=0
dire=[[-1,0],[1,0],[0,1],[0,-1]]
def dfs(ini_x,ini_y):
    graph[ini_x][ini_y]="-"
    for i in dire:
        newx,newy=ini_x+i[0],ini_y+i[1]
        if 0<=newx<10 and 0<=newy<10 and graph[newx][newy]==".":
            dfs(newx,newy)

for line in range(10):
    for row in range(10):
        if graph[line][row]==".":
            dfs(line,row)
            cnt+=1
print(cnt)
```

基本信息

#:	44859200
题目:	28170
提交人:	23n2300013289
内存:	3648kB
时间:	19ms
语言:	Python3
提交时间:	2024-05-04 15:53:31

©2002-2022 POJ 京ICP备20010980号-1

02754: 八皇后

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

思路：第一次接触dfs+回溯，参考题解。

用一个一维数组conditions来记录各行的皇后所在列数

用issafe函数来判断当前行，当前列是否安全

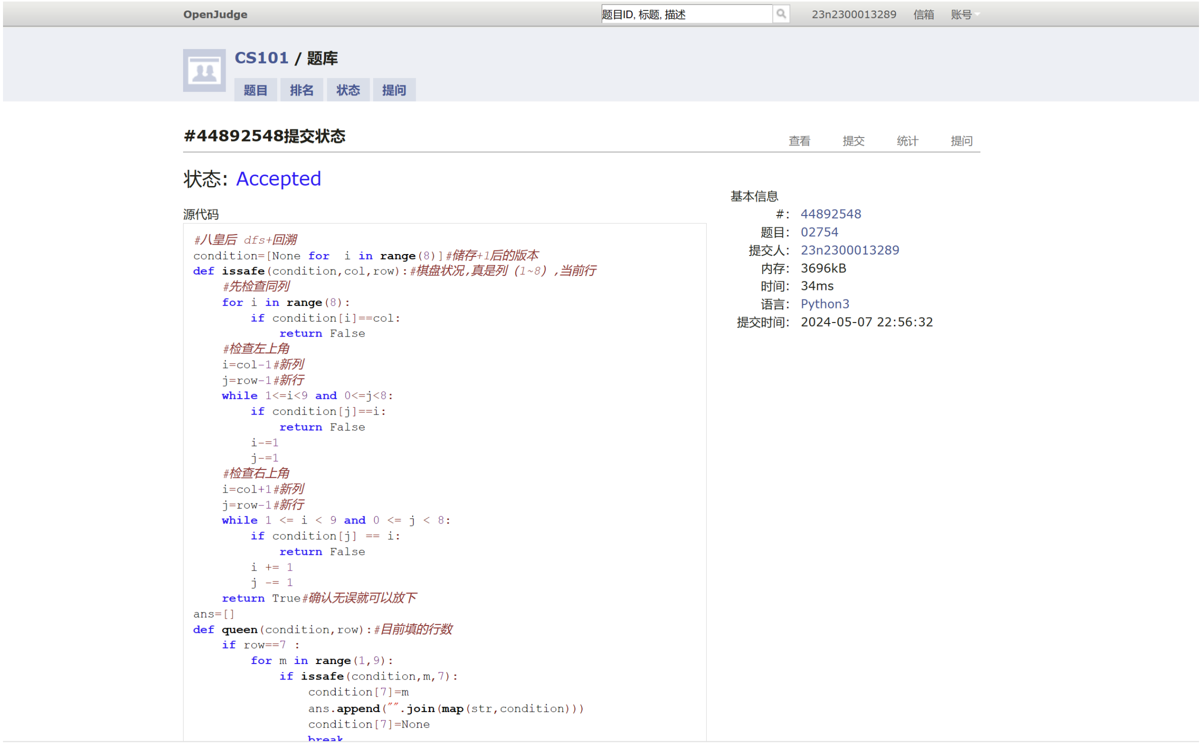
若一路安全，则dfs走到底，最后一行确认后，更改conditions状态来记录结果，然后改回None，算是回溯

若中途走不通，就回到当前行，把conditions当前行改回None，算是回溯，继续看看该行位置 is safe or not

代码

```
# #八皇后 dfs+回溯
condition=[None for  i in range(8)]#储存+1后的版本
def issafe(condition,col,row):#棋盘状况,真是列（1~8）,当前行
    #先检查同列
    for i in range(8):
        if condition[i]==col:
            return False
    #检查左上角
    i=col-1#新列
    j=row-1#新行
    while 1<=i<9 and 0<=j<8:
        if condition[j]==i:
            return False
        i-=1
        j-=1
    #检查右上角
    i=col+1#新列
    j=row-1#新行
    while 1 <= i < 9 and 0 <= j < 8:
        if condition[j] == i:
            return False
        i += 1
        j -= 1
    return True#确认无误就可以放下
ans=[]
def queen(condition,row):#目前填的行数
    if row==7 :
        for m in range(1,9):
            if issafe(condition,m,7):
                condition[7]=m
                ans.append("".join(map(str,condition)))
                condition[7]=None
                break
    elif row<7:
        for ii in range(1,9):
            if issafe(condition,ii,row):
                condition[row]=ii
                queen(condition,row+1)#递归下一行
            #回溯
                condition[row]=None
n=int(input())
queen(condition,0)
ans.sort()
for p in range(n):
    g=int(input())-1
    print(ans[g])
```

代码运行截图 ==（至少包含有"Accepted"）==



#44892548提交状态

状态: Accepted

源代码

```
#八皇后 dfs+回溯
condition=[None for  i in range(8)]#储存+1后的版本
def issafe(condition,col,row):#棋盘状况,真是列（1~8）,当前行
    #先检查同列
    for i in range(8):
        if condition[i]==col:
            return False
    #检查左上角
    i=col-1#新列
    j=row-1#新行
    while 1<=i<9 and 0<=j<8:
        if condition[j]==i:
            return False
        i-=1
        j-=1
    #检查右上角
    i=col+1#新列
    j=row-1#新行
    while 1 <= i < 9 and 0 <= j < 8:
        if condition[j] == i:
            return False
        i += 1
        j -= 1
    return True#确认无误就可以放下
ans=[]
def queen(condition,row):#目前填的行数
    if row==7 :
        for m in range(1,9):
            if issafe(condition,m,7):
                condition[7]=m
                ans.append("".join(map(str,condition)))
                condition[7]=None
                break
```

基本信息

#: 44892548
题目: 02754
提交人: 23n2300013289
内存: 3696kB
时间: 34ms
语言: Python3
提交时间: 2024-05-07 22:56:32

03151: Pots

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

思路：真没动力写了，先放一放orz

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

05907: 二叉树的操作

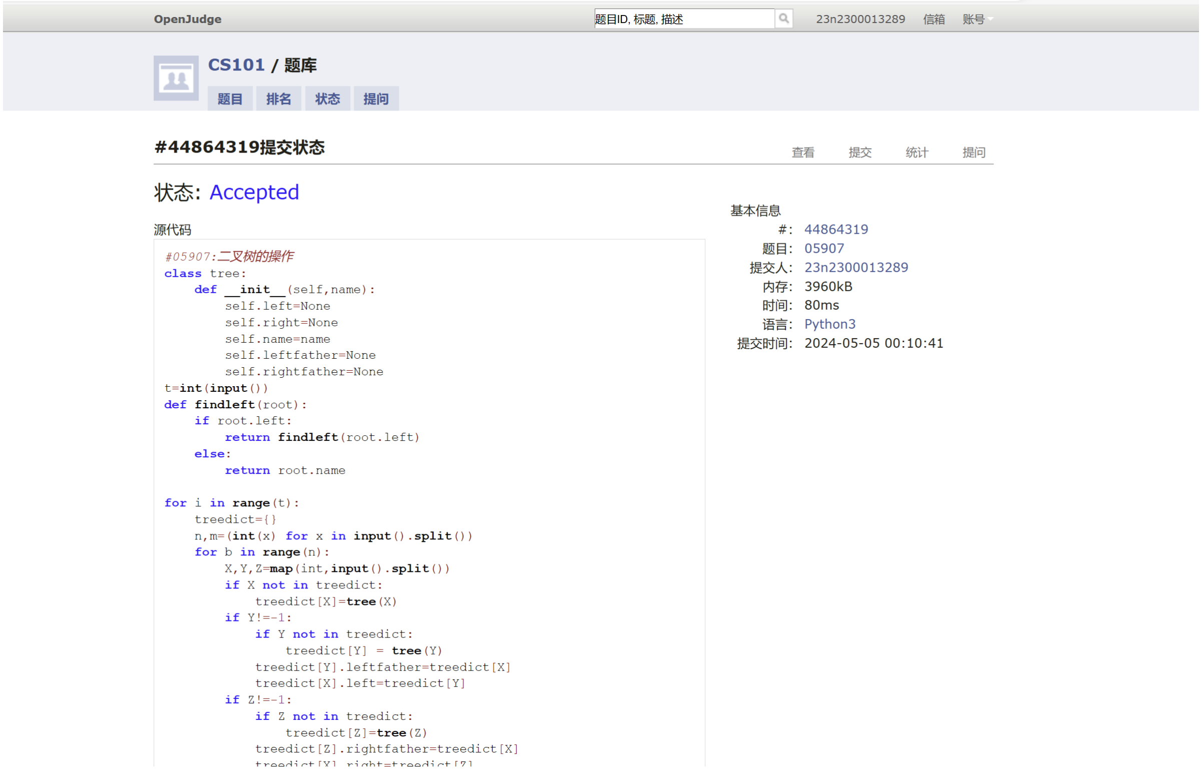
http://cs101.openjudge.cn/practice/05907/

思路：leftfather表示是原父亲的左儿子，rightfather表示是原父亲的右儿子

代码

```python
# #05907:二叉树的操作
class tree:
    def __init__(self,name):
        self.left=None
        self.right=None
        self.name=name
        self.leftfather=None
        self.rightfather=None
t=int(input())
def findleft(root):
    if root.left:
        return findleft(root.left)
    else:
        return root.name

for i in range(t):
    treedict={}
    n,m=(int(x) for x in input().split())
    for b in range(n):
        X,Y,Z=map(int,input().split())
        if X not in treedict:
            treedict[X]=tree(X)
        if Y!=-1:
            if Y not in treedict:
                treedict[Y] = tree(Y)
            treedict[Y].leftfather=treedict[X]
            treedict[X].left=treedict[Y]
        if Z!=-1:
            if Z not in treedict:
                treedict[Z]=tree(Z)
            treedict[Z].rightfather=treedict[X]
            treedict[X].right=treedict[Z]
    for p in range(m):
        sample=[int(x) for x in input().split()]
        if sample[0]==1:
            a,b=sample[1],sample[2]
            atree=treedict[a]
            btree=treedict[b]
            if atree.leftfather:
                atree_fa=atree.leftfather
                siga=1
            else:
                atree_fa=atree.rightfather
                siga=0
            if btree.leftfather:
                btree_fa=btree.leftfather
                sigb=1
            else:
                btree_fa=btree.rightfather
                sigb=0

            if sigb==1:
                btree_fa.left=atree
                atree.leftfather=btree_fa
                atree.rightfather=None
            else:
                btree_fa.right=atree
                atree.rightfather=btree_fa
                atree.leftfather=None
            if siga==1:
                atree_fa.left=btree
                btree.leftfather=atree_fa
                btree.rightfather=None
            else:
                atree_fa.right=btree
                btree.rightfather = atree_fa
                btree.leftfather=None
        else:
            print(findleft(treedict[sample[1]]))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44864319提交状态

状态: Accepted

源代码

```
#05907:二叉树的操作
class tree:
    def __init__(self,name):
        self.left=None
        self.right=None
        self.name=name
        self.leftfather=None
        self.rightfather=None
t=int(input())
def findleft(root):
    if root.left:
        return findleft(root.left)
    else:
        return root.name

for i in range(t):
    treedict={}
    n,m=(int(x) for x in input().split())
    for b in range(n):
        X,Y,Z=map(int,input().split())
        if X not in treedict:
            treedict[X]=tree(X)
        if Y!=-1:
            if Y not in treedict:
                treedict[Y] = tree(Y)
            treedict[Y].leftfather=treedict[X]
            treedict[X].left=treedict[Y]
        if Z!=-1:
            if Z not in treedict:
                treedict[Z]=tree(Z)
            treedict[Z].rightfather=treedict[X]
            treedict[X].right=treedict[Z]
```

基本信息

#:	44864319
题目:	05907
提交人:	23n2300013289
内存:	3960kB
时间:	80ms
语言:	Python3
提交时间:	2024-05-05 00:10:41

18250: 冰阔落 I

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

思路：如果不用并查集的话，数据不太强，压线过

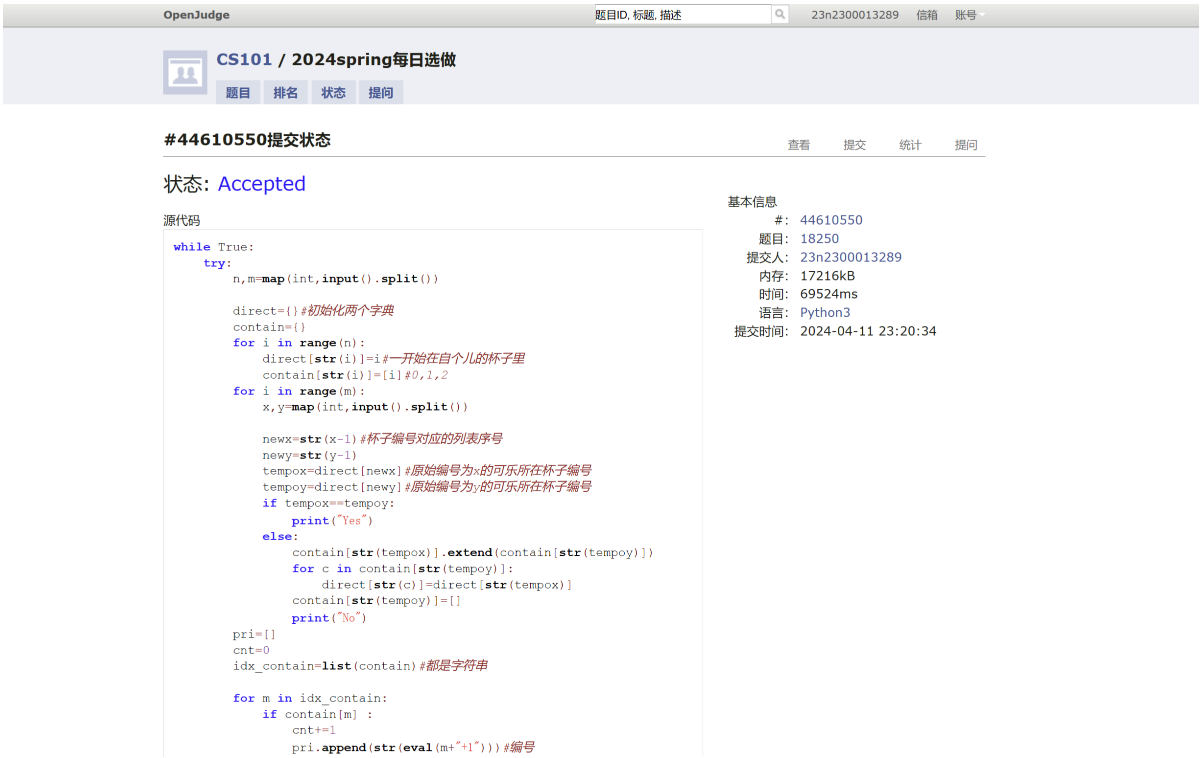
但是还是推荐用并查集

代码

```python
# 不用并查集版
while True:
    try:
        n,m=map(int,input().split())

        direct={}#初始化两个字典
        contain={}
        for i in range(n):
            direct[str(i)]=i#一开始在自个儿的杯子里
            contain[str(i)]=[i]#0,1,2
        for i in range(m):
            x,y=map(int,input().split())

            newx=str(x-1)#杯子编号对应的列表序号
            newy=str(y-1)
            tempox=direct[newx]#原始编号为x的可乐所在杯子编号
            tempoy=direct[newy]#原始编号为y的可乐所在杯子编号
            if tempox==tempoy:
                print("Yes")
            else:
                contain[str(tempox)].extend(contain[str(tempoy)])
                for c in contain[str(tempoy)]:
                    direct[str(c)]=direct[str(tempox)]
                contain[str(tempoy)]=[]
                print("No")
        pri=[]
        cnt=0
        idx_contain=list(contain)#都是字符串

        for m in idx_contain:
            if contain[m] :
                cnt+=1
                pri.append(str(eval(m+"+1")))#编号
        print(cnt)
        print(" ".join(pri))
    except EOFError:
        break

#用并查集版
def root(x,a):
    if a[x]==x:
        return x
    a[x]=root(a[x],a)
    return a[x]

def find(l,r,a,cnt):
    ll=root(l,a)
    rr=root(r,a)
    if ll==rr:
        print('Yes')
    else:
        a[rr]=ll
        cnt-=1
        print('No')
    return cnt
```

```python
while True:
    try:
        n,m=map(int,input().split())
        cnt=n
        a=[i for i in range(n+1)]
        for i in range(m):
            l,r=map(int,input().split())
            cnt=find(l,r,a,cnt)
        print(cnt)
        for i in range(1,n+1):
            if a[i]==i:
                print(i,end=' ')
        print()
    except EOFError:
        break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44610550提交状态

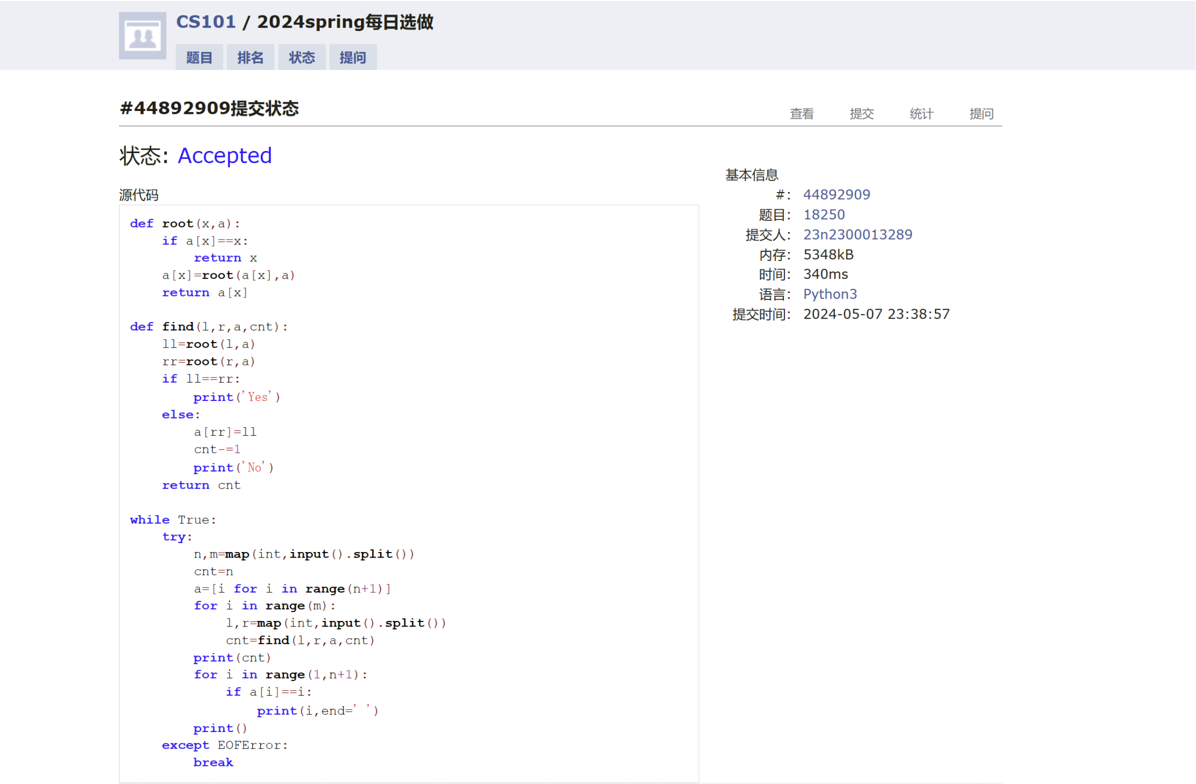
状态: Accepted

源代码

```python
while True:
    try:
        n,m=map(int,input().split())

        direct={}#初始化两个字典
        contain={}
        for i in range(n):
            direct[str(i)]=i#一开始在自个儿的杯子里
            contain[str(i)]=[i]#0,1,2
        for i in range(m):
            x,y=map(int,input().split())

            newx=str(x-1)#杯子编号对应的列表序号
            newy=str(y-1)
            tempox=direct[newx]#原始编号为x的可乐所在杯子编号
            tempoy=direct[newy]#原始编号为y的可乐所在杯子编号
            if tempox==tempoy:
                print(″Yes″)
            else:
                contain[str(tempox)].extend(contain[str(tempoy)])
                for c in contain[str(tempoy)]:
                    direct[str(c)]=direct[str(tempox)]
                contain[str(tempoy)]=[]
                print(″No″)
        pri=[]
        cnt=0
        idx_contain=list(contain)#都是字符串

        for m in idx_contain:
            if contain[m] :
                cnt+=1
                pri.append(str(eval(m+″+1″)))#编号
```

基本信息

#:	44610550
题目:	18250
提交人:	23n2300013289
内存:	17216kB
时间:	69524ms
语言:	Python3
提交时间:	2024-04-11 23:20:34



#44892909提交状态

状态: Accepted

源代码

```
def root(x,a):
    if a[x]==x:
        return x
    a[x]=root(a[x],a)
    return a[x]

def find(l,r,a,cnt):
    ll=root(l,a)
    rr=root(r,a)
    if ll==rr:
        print('Yes')
    else:
        a[rr]=ll
        cnt-=1
        print('No')
    return cnt

while True:
    try:
        n,m=map(int,input().split())
        cnt=n
        a=[i for i in range(n+1)]
        for i in range(m):
            l,r=map(int,input().split())
            cnt=find(l,r,a,cnt)
        print(cnt)
        for i in range(1,n+1):
            if a[i]==i:
                print(i,end=' ')
        print()
    except EOFError:
        break
```

基本信息

#:	44892909
题目:	18250
提交人:	23n2300013289
内存:	5348kB
时间:	340ms
语言:	Python3
提交时间:	2024-05-07 23:38:57

05443: 兔子与樱花

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

思路：才发现上周的走山路把heap+bfs理解成迪杰斯特拉了（囧orz

这次更新了认知

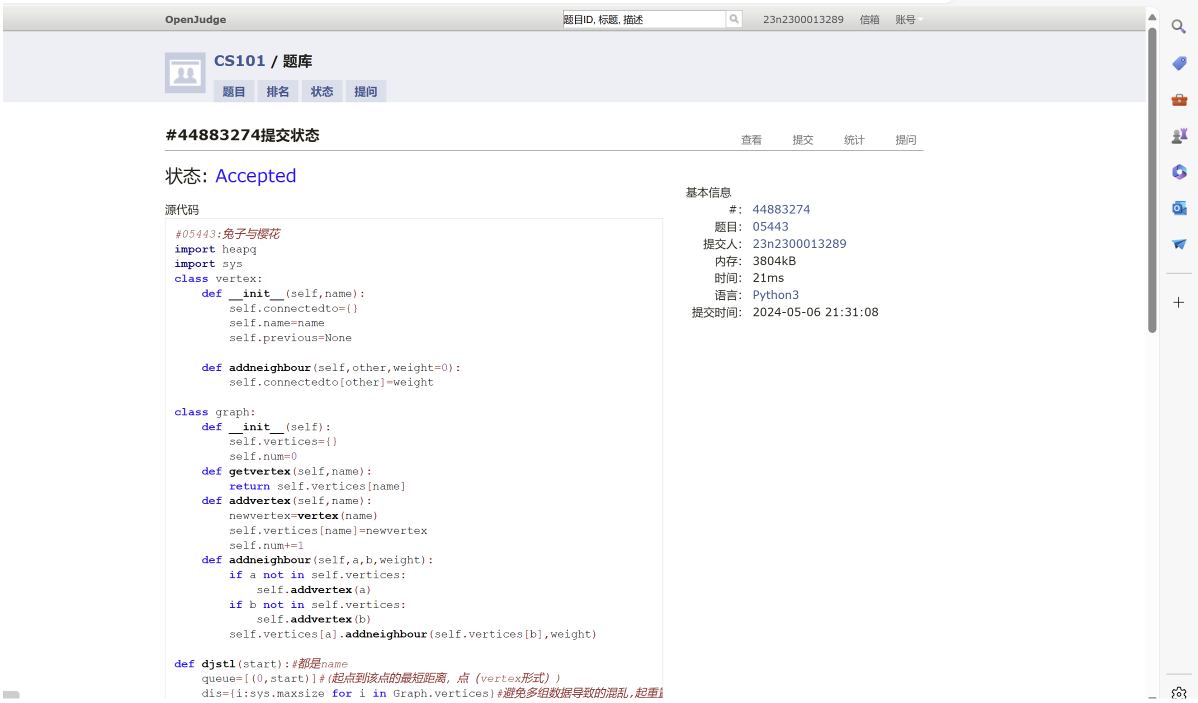
很经典的迪杰斯特拉算法

代码

```python
# #05443:兔子与樱花
import heapq
import sys
class vertex:
    def __init__(self,name):
        self.connectedto={}
        self.name=name
        self.previous=None

    def addneighbour(self,other,weight=0):
        self.connectedto[other]=weight

class graph:
    def __init__(self):
        self.vertices={}
        self.num=0
    def getvertex(self,name):
        return self.vertices[name]
    def addvertex(self,name):
        newvertex=vertex(name)
        self.vertices[name]=newvertex
        self.num+=1
    def addneighbour(self,a,b,weight):
        if a not in self.vertices:
            self.addvertex(a)
        if b not in self.vertices:
            self.addvertex(b)
        self.vertices[a].addneighbour(self.vertices[b],weight)

def djstl(start):#都是name
    queue=[(0,start)]#(起点到该点的最短距离，点（vertex形式）)
    dis={i:sys.maxsize for i in Graph.vertices}#避免多组数据导致的混乱,起重置作用
    previous={i:None for i in Graph.vertices}#同上
    dis[start]=0
    visited=set()#放节点
    while queue:
        currentdis,currentvertex=heapq.heappop(queue)
        if currentvertex not in visited:
            visited.add(currentvertex)#表示最短路径已经确定
            neighbour=currentvertex.connectedto#全是节点
            for i in neighbour:
                newdis=currentdis+currentvertex.connectedto[i]#新的，起点到邻居点的距离
                if newdis<dis[i.name] :
                    previous[i.name]=currentvertex
                    dis[i.name]=newdis
                    heapq.heappush(queue,(newdis,i))
    return previous
#建图
p=int(input())
Graph=graph()
for i in range(p):
    sample=input()
    Graph.addvertex(sample)
m=int(input())
for t in range(m):
    sample=[x for x in input().split()]
    a=sample[0]
    b=sample[1]
    lenn=int(sample[2])
    Graph.addneighbour(a,b,lenn)
    Graph.addneighbour(b,a,lenn)

s=int(input())
for c in range(s):
    start,end=map(str,input().split())
    if start==end:
        print(start)
    else:
        pri_pre=djstl(Graph.getvertex(start))
        ans=[]
        flag=0
        newend = Graph.getvertex(end)
        #回溯
        while newend.name!=start:
```

```
        tem=pri_pre[newend.name]
        if tem==None:#如果为None,表示目前的newend是start
            flag=1
            ans.append(newend.name)
            break
        ans.append(newend.name)
        tt=newend.connectedto[tem]
        ans.append(f"({tt})")
        newend=tem
    ans.append(start)
    print("->".join(map(str,ans[::-1])))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44883274提交状态

状态: Accepted

源代码

```
#05443:兔子与樱花
import heapq
import sys
class vertex:
    def __init__(self,name):
        self.connectedto={}
        self.name=name
        self.previous=None

    def addneighbour(self,other,weight=0):
        self.connectedto[other]=weight

class graph:
    def __init__(self):
        self.vertices={}
        self.num=0
    def getvertex(self,name):
        return self.vertices[name]
    def addvertex(self,name):
        newvertex=vertex(name)
        self.vertices[name]=newvertex
        self.num+=1
    def addneighbour(self,a,b,weight):
        if a not in self.vertices:
            self.addvertex(a)
        if b not in self.vertices:
            self.addvertex(b)
        self.vertices[a].addneighbour(self.vertices[b],weight)

def djstl(start):#都是name
    queue=[(0,start)]#(起点到该点的最短距离，点（vertex形式））
    dis={i:sys.maxsize for i in Graph.vertices}#避免多组数据导致的混乱,起重
```

基本信息

#:	44883274
题目:	05443
提交人:	23n2300013289
内存:	3804kB
时间:	21ms
语言:	Python3
提交时间:	2024-05-06 21:31:08

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。==

遇到树的题目会很开心

对于图的bfs、dfs、迪杰斯特拉、krustal、prim......一堆的算法逐渐增进理解中

上周好不容易做完作业就去招待好朋友去了，结果又来六道题，属实是觉得有点累了，希望能坚持到终点！