Assignment #B: 图论和树算

Updated 1709 GMT+8 Apr 28, 2024

2024 spring, Complied by 赵语涵 2300012254

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

操作系统：windows 11

Python编程环境：Spyder IDE 5.2.2

1. 题目

28170: 算鹰

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

思路：dfs和mark标记得出相连的点

代码

```
#赵语涵2300012254
def dfs(x,y):
    global mark
    mark[x][y] = False
    for w in ways:
        a,b = x+w[0],y+w[1]
        if 0<=a<10 and 0<=b<10:
            if fig[a][b] == '.' and mark[a][b]:
                dfs(a,b)
    return

ways = [(-1,0),(0,1),(1,0),(0,-1)]
fig,mark = [],[]
for i in range(10):
    fig.append(input())
```

```
    mark.append([True]*10)

count = 0
for i in range(10):
    for j in range(10):
        if fig[i][j] == '.' and mark[i][j]:
            dfs(i,j)
            count += 1
print(count)
```

代码运行截图

#44973653提交状态 查看 提交 统计

状态: Accepted

源代码

```
#赵语涵2300012254
def dfs(x,y):
    global mark
    mark[x][y] = False
    for w in ways:
        a,b = x+w[0],y+w[1]
        if 0<=a<10 and 0<=b<10:
```

基本信息

#: 44973653
题目: 28170
提交人: 23n2300012254
内存: 3672kB
时间: 23ms
语言: Python3
提交时间: 2024-05-15 19:09:18

02754: 八皇后

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

思路：层数为i，根据已有层数的位置得到不能填的位置后填入能够填入的最小位置。成功填入则递归填下一位置，1至8全都不能填入则说明上一层的填入方法是不可行的，标记上一层原本位置后递归退回上一层，重新填除了原位置以外可填的最小数字。

代码

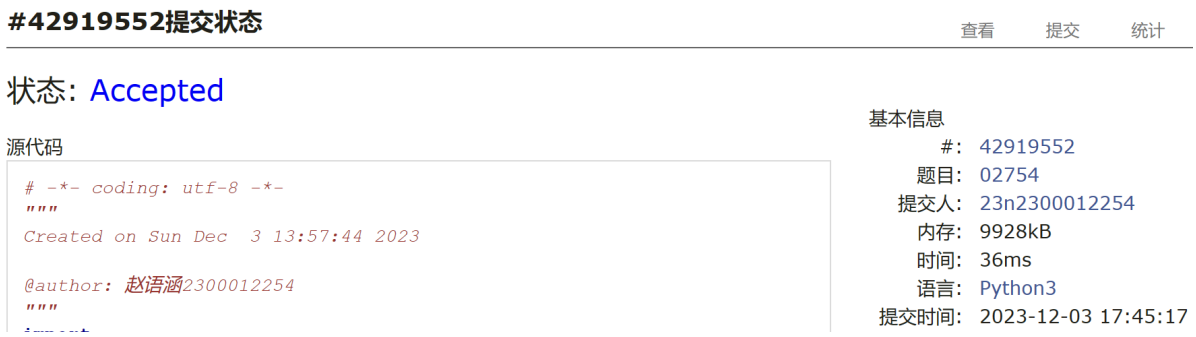
```
# -*- coding: utf-8 -*-
"""
Created on Sun Dec  3 13:57:44 2023

@author: 赵语涵2300012254
"""
import sys
sys.setrecursionlimit(1<<30)
jump = {x:set() for x in range(9)}
def fill(answer,i,jump):
    global final
    unsuit = set()
    for y in range(len(answer)):
        unsuit.update([answer[y],answer[y]-(i-y-1),answer[y]+(i-y-1)])
    for x in range(1,9):
```

```
        if x not in unsuit and x not in jump[i]:
            answer.append(x)
            if i < 8:
                answer = fill(answer, i+1,jump)
            else:
                final.append(''.join(map(str,answer)))
            break
    else:
        jump[i-1].add(answer[-1])
        jump[i] = set()
        answer.pop()
        answer = fill(answer,i-1,jump)
    if len(final) >= 92:
        return answer
    else:
        jump[i].add(answer[-1])
        answer.pop()
        answer = fill(answer,i,jump)

final = []
answer = fill([],1,jump)
for _ in range(int(input())):
    x = int(input())-1
    print(final[x])
```

代码运行截图



03151: Pots

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

思路：思路很简单的bfs但是写起来是真麻烦....

代码

```
#赵语涵2300012254
from collections import deque
def command(x,old):
```

```
    pots = [old[0],old[1],old[2]+1,old[3]+str(x)]
    if x == 0:
        pots[0] = a
    elif x == 1:
        pots[1] = b
    elif x == 2:
        pots[0] = 0
    elif x == 3:
        pots[1] = 0
    elif x == 4:
        temp_0,temp_1 = max(pots[0]+pots[1]-b,0),min(b,pots[0]+pots[1])
        pots[0],pots[1] = temp_0,temp_1
    elif x == 5:
        temp_1,temp_0 = max(pots[0]+pots[1]-a,0),min(a,pots[0]+pots[1])
        pots[0],pots[1] = temp_0,temp_1
    return pots

a,b,c = map(int,input().split())
start = [0,0,0,'']  #pot1,pot2,步数,路径
visited = set()
result = deque([start])
commands = ['FILL(1)','FILL(2)','DROP(1)','DROP(2)','POUR(1,2)','POUR(2,1)']
while result:
    p = result.popleft()
    if p[0]==c or p[1]==c:
        print(p[2])
        n = 0
        for n in range(len(p[3])):
            print(commands[int(p[3][n])])
        break
    for x in range(6):
        new = command(x, p)
        if (n:=(new[0],new[1])) not in visited:
            result.append(new)
            visited.add(n)
else:
    print('impossible')
```

代码运行截图

#44975625提交状态

查看　提交　统计

状态: Accepted

源代码

```
#赵语涵2300012254
from collections import deque
def command(x,old):
    pots = [old[0],old[1],old[2]+1,old[3]+str(x)]
    if x == 0:
        pots[0] = a
```

基本信息

#: 44975625
题目: 03151
提交人: 23n2300012254
内存: 3752kB
时间: 24ms
语言: Python3
提交时间: 2024-05-15 21:58:47

05907: 二叉树的操作

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

思路：感觉方法很简单，但是要简洁地写出树交换的代码比较花功夫。这里在parent参数中直接加入了方向参数，不需要再查询判断parent树的左还是右子树为该子树。

代码

```
#赵语涵2300012254
class Node():
    def __init__(self,x):
        self.value = x
        self.child = {'left':None,'right':None}
        self.parent = None
    def seek(self):
        if self.child['left']:
            return self.child['left'].seek()
        else:
            return self.value

def change(a,b):
    if tree[a].parent:
        tree[a].parent[0].child[tree[a].parent[1]] = tree[b]
    if tree[b].parent:
        tree[b].parent[0].child[tree[b].parent[1]] = tree[a]
    tree[a].parent,tree[b].parent = tree[b].parent,tree[a].parent

for _ in range(int(input())):
    n,m = map(int,input().split())
    tree = [Node(x) for x in range(n)]
    for _ in range(n):
        v,l,r = map(int,input().split())
        if l != -1:
            tree[v].child['left'] = tree[l]
            tree[l].parent = (tree[v],'left')
        if r != -1:
            tree[v].child['right'] = tree[r]
            tree[r].parent = (tree[v],'right')
    for _ in range(m):
        ope = input().split()
        if ope[0]=='1':
            change(int(ope[1]),int(ope[2]))
        else:
            print(tree[int(ope[1])].seek())
```

代码运行截图

#44976543提交状态 查看 提交 统计

状态: Accepted

源代码

```
#赵语涵2300012254
class Node():
    def __init__(self,x):
        self.value = x
        self.child = {'left':None,'right':None}
        self.parent = None
    def seek(self):
```

基本信息

#: 44976543
题目: 05907
提交人: 23n2300012254
内存: 4880kB
时间: 81ms
语言: Python3
提交时间: 2024-05-16 00:08:50

18250: 冰阔落 I

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

思路：找并查集并且不需要考虑把谁作为祖先的很简单的写法。自己写的一直RE，也找不到问题，把代码一点点朝题解改（虽然改的地方也不觉得自己的就是错的。。。

发在群里成功找到错误的地方了好耶

代码

```python
#2300012254
def find(x):
    if parent[x] != x:
        parent[x] = find(parent[x])
    return parent[x]

def union(x, y):
    root_x = find(x)
    root_y = find(y)
    if root_x != root_y:
        parent[root_y] = root_x

while True:
    try:
        n, m = map(int, input().split())
        parent = list(range(n + 1))

        for _ in range(m):
            a, b = map(int, input().split())
            if find(a) == find(b):
                print('Yes')
            else:
                print('No')
                union(a, b)

        unique_parents = set(find(x) for x in range(1, n + 1))  # 获取不同集合的根节点
```

```
        ans = sorted(unique_parents)  # 输出有冰阔落的杯子编号
        print(len(ans))
        print(*ans)

    except EOFError:
        break
```

代码运行截图

#44994073提交状态 查看 提交 统计

状态: Accepted

源代码

```
#2300012254
def find(x):
    if parent[x] != x:
        parent[x] = find(parent[x])
    return parent[x]

def union(x, y):
```

基本信息

#: 44994073
题目: 18250
提交人: 23n2300012254
内存: 5500kB
时间: 369ms
语言: Python3
提交时间: 2024-05-17 21:08:42

05443: 兔子与樱花

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

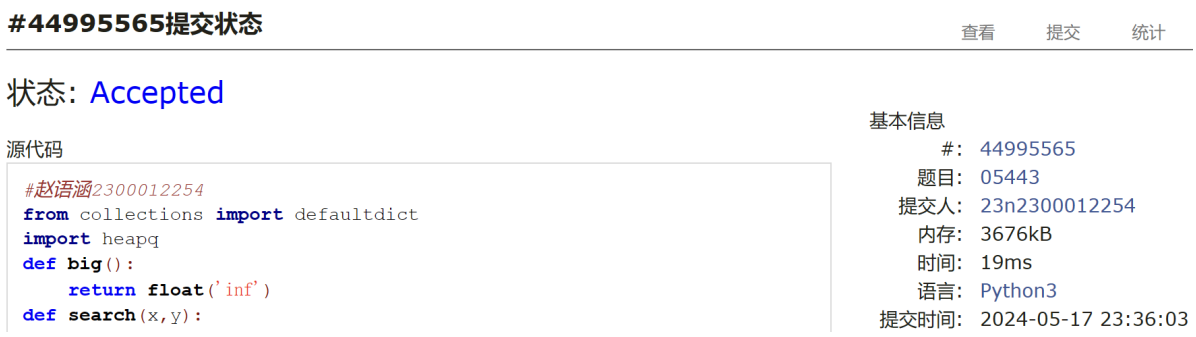
思路：用的上学期学过的dijkstra算法，但是自己开始并没有想到用heapq的简洁写法，看了题解才想起来。复习了一下写法。在抠细节上花的时间有点多

代码

```
#赵语涵2300012254
from collections import defaultdict
import heapq
def big():
    return float('inf')
def search(x,y):
    processed = set()
    a = [(0,x,None)]
    parent = {x:None}
    heapq.heapify(a)
    while True:
        x = heapq.heappop(a)
        if x[1] in processed:
            continue
        parent[x[1]] = x[2]
        if x[1] == y:
            return parent
        processed.add(x[1])
        for p in edges[x[1]]:
            if p not in processed:
```

```
                heapq.heappush(a, (x[0]+edges[x[1]][p],p,x[1]))
edges = defaultdict(dict)
for p in range(int(input())):
    input()
for q in range(int(input())):
    p1,p2,d = input().split()
    edges[p1][p2] = int(d)
    edges[p2][p1] = int(d)
for r in range(int(input())):
    x,y = input().split()
    parent = search(x, y)
    last,ans = y,[]
    while (p:=parent[last]):
        ans.append(last)
        ans.append('('+str(edges[p][last])+')')
        last = p
    ans.append(x)
    print('->'.join(reversed(ans)))
```

代码运行截图



2. 学习总结和收获

作业的思路都比较简单能很快想到，但是在一些细节上容易出错，需要debug很久。复习了并查集和dijkstra等的一些模式写法