Assignment #9: 图论：遍历，及 树算

Updated 1739 GMT+8 Apr 14, 2024

2024 spring, Complied by ==同学的姓名、院系==

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

04081: 树的转换

http://cs101.openjudge.cn/dsapre/04081/

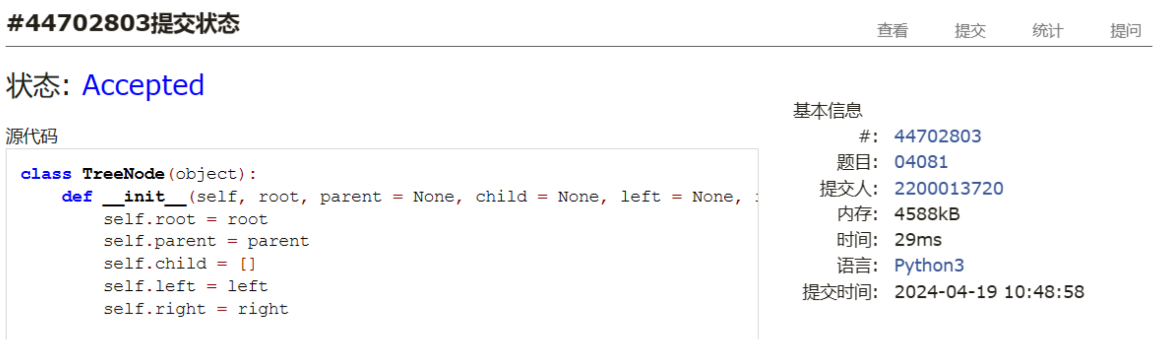
思路：朴实无华的建树、转换、找深度，但是一开始是想直接通过给出的字符串算出h1、h2的（因为不需要输出树的详细信息），不过没写出来。好像群里有同学分享了非建树做法，可以学习下

代码

```
#
class TreeNode(object):
    def __init__(self, root, parent = None, child = None, left = None, right = None):
        self.root = root
        self.parent = parent
        self.child = []
        self.left = left
        self.right = right

def build(order, count):
```

```python
    trees = [TreeNode(0)]
    target = 0
    height = 0
    result = 0
    for i in order:
        if i == 'd':
            count += 1
            trees.append(TreeNode(count))
            trees[-1].parent = trees[target]
            trees[target].child.append(trees[-1])
            target = trees[-1].root
            height += 1
        elif i == 'u':
            target = trees[target].parent.root
            height -= 1
        result = max(result, height)
    return trees, result

def rebuild(treelist):
    if treelist == []:
        pass
    elif len(treelist) == 1:
        tree = treelist[0]
        tree.left = rebuild(tree.child)
        return tree
    else:
        left = treelist[0]
        right = treelist[1:]
        left.right = rebuild(right)
        left.left = rebuild(left.child)
        return left

def finddepth(tree):
    if tree.left == None and tree.right == None:
        return 0
    elif tree.left == None:
        return 1 + finddepth(tree.right)
    elif tree.right == None:
        return 1 + finddepth(tree.left)
    else:
        return 1 + max(finddepth(tree.right),finddepth(tree.left))
order = input()
trees, height = build(order, 0)
tree = rebuild([trees[0]])
print("%s => %s"%(height, finddepth(tree)))
```

代码运行截图 == （至少包含有"Accepted"） ==


#44702803提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
class TreeNode(object):
    def __init__(self, root, parent = None, child = None, left = None, 
        self.root = root
        self.parent = parent
        self.child = []
        self.left = left
        self.right = right
```

基本信息

#:	44702803
题目:	04081
提交人:	2200013720
内存:	4588kB
时间:	29ms
语言:	Python3
提交时间:	2024-04-19 10:48:58

08581: 扩展二叉树

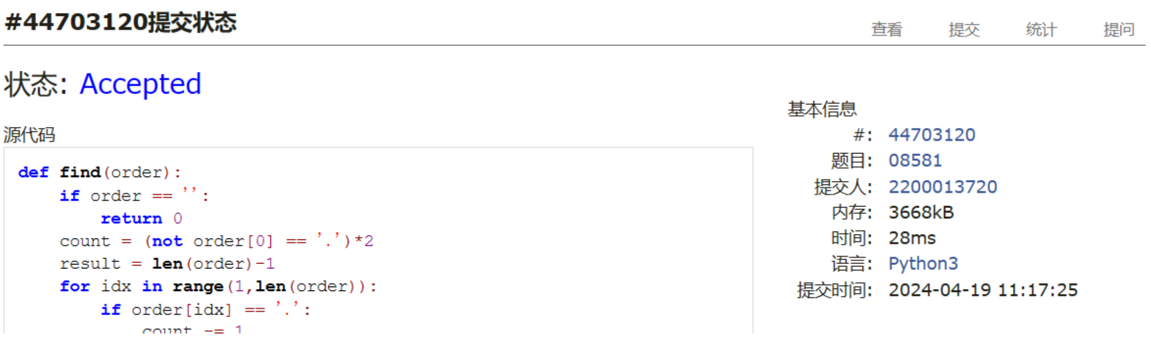
http://cs101.openjudge.cn/dsapre/08581/

思路：还是情不自禁地用非tree做法做了，具体思路就是：由于所给为先序遍历，所以可以确定根，然后通过find函数找到左子树的截止点，随后递归即可，关键在于find函数的实现

代码

```python
#
def find(order):
    if order == '':
        return 0
    count = (not order[0] == '.')*2
    result = len(order)-1
    for idx in range(1,len(order)):
        if order[idx] == '.':
            count -= 1
        else:
            count += 1
        if count == 0:
            result = idx
            break
    return result

def change(order):
    if order == '':
        return [], []
    mid_result = []
    pos_result = []
    root = order[0]
    idx = find(order[1:])
    adleft_mid, adleft_pos = change(order[1:idx+1])
    adright_mid, adright_pos = change(order[idx+1:])

    mid_result += adleft_mid
    if not root == '.':
        mid_result.append(root)
    mid_result += adright_mid

    pos_result += adleft_pos
    pos_result += adright_pos
    if not root == '.':
            pos_result.append(root)

    return mid_result, pos_result

order = input()
mid, pos = change(order)
print(''.join(mid))
print(''.join(pos))
```

代码运行截图 ==（至少包含有"Accepted"）==


#44703120提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def find(order):
    if order == '':
        return 0
    count = (not order[0] == '.')*2
    result = len(order)-1
    for idx in range(1,len(order)):
        if order[idx] == '.':
            count -= 1
```

基本信息

#: 44703120
题目: 08581
提交人: 2200013720
内存: 3668kB
时间: 28ms
语言: Python3
提交时间: 2024-04-19 11:17:25

22067: 快速堆猪

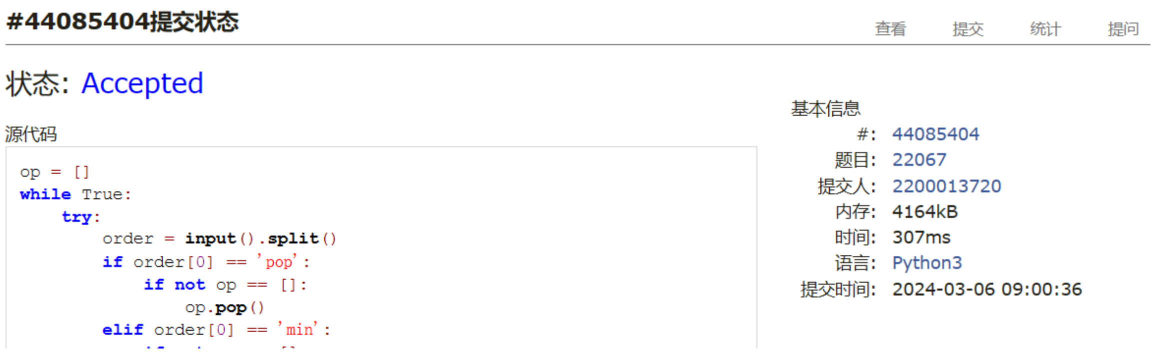
http://cs101.openjudge.cn/practice/22067/

思路：一开始真的卡了很久，后面借鉴了群里大佬的维护数组思路也是成功写了出来

代码

```
#
op = []
while True:
    try:
        order = input().split()
        if order[0] == 'pop':
            if not op == []:
                op.pop()
        elif order[0] == 'min':
            if not op == []:
                print(op[-1])
        else:
            if not op == []:
                op.append(min(op[-1],int(order[1])))
            else:
                op.append(int(order[1]))
    except EOFError:
        break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==


#44085404提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
op = []
while True:
    try:
        order = input().split()
        if order[0] == 'pop':
            if not op == []:
                op.pop()
        elif order[0] == 'min':
```

基本信息

#:	44085404
题目:	22067
提交人:	2200013720
内存:	4164kB
时间:	307ms
语言:	Python3
提交时间:	2024-03-06 09:00:36

04123: 马走日

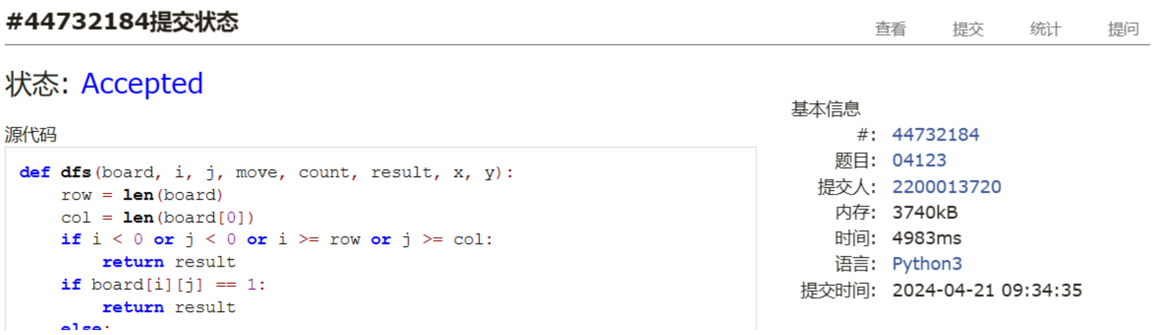
dfs, http://cs101.openjudge.cn/practice/04123

思路：经典的dfs思路，只是需要注意一下result+1的判断时机就好了

代码

```
#
def dfs(board, i, j, move, count, result, x, y):
    row = len(board)
    col = len(board[0])
    if i < 0 or j < 0 or i >= row or j >= col:
        return result
    if board[i][j] == 1:
        return result
    else:
        board[i][j] = 1
        count += 1
        if count == row*col:
            result += 1
        else:
            for m in move:
                result = dfs(board, i+m[0], j+m[1], move, count, result, x, y)
        if not i == x or not j == y:
            board[i][j] = 0
            count -= 1
        return result

t = int(input())
for _ in range(t):
    n, m, x, y = map(int,input().split())
    board = []
    move = [[-2,-1],[-2,1],[2,-1],[2,1],[-1,-2],[-1,2],[1,-2],[1,2]]
    for i in range(n):
        board.append([0]*m)
    print(dfs(board, x, y, move, 0, 0, x, y))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==


#44732184提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def dfs(board, i, j, move, count, result, x, y):
    row = len(board)
    col = len(board[0])
    if i < 0 or j < 0 or i >= row or j >= col:
        return result
    if board[i][j] == 1:
        return result
```

基本信息

#:	44732184
题目:	04123
提交人:	2200013720
内存:	3740kB
时间:	4983ms
语言:	Python3
提交时间:	2024-04-21 09:34:35

28046: 词梯

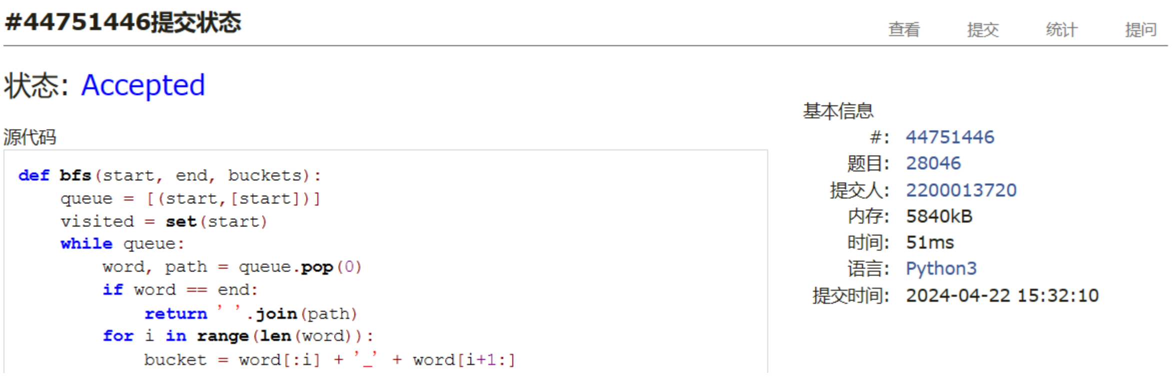
bfs, http://cs101.openjudge.cn/practice/28046/

思路：借鉴了Github上同学的思路，感觉到了桶的精妙，至少在解决问题方面桶会更加容易理解且实现

代码

```python
# 
def bfs(start, end, buckets):
    queue = [(start,[start])]
    visited = set(start)
    while queue:
        word, path = queue.pop(0)
        if word == end:
            return ' '.join(path)
        for i in range(len(word)):
            bucket = word[:i] + '_' + word[i+1:]
            for nbr in buckets[bucket]:
                if nbr not in visited:
                    queue.append((nbr, path + [nbr]))
                    visited.add(nbr)
    return 'NO'

from collections import defaultdict
buckets = defaultdict(list)
n = int(input())
words = []
for _ in range(n):
    word = input()
    words.append(word)
    for i in range(len(word)):
        bucket = word[:i] + '_' + word[i+1:]
        buckets[bucket].append(word)
start, end = input().split()
print(bfs(start, end, buckets))
```

代码运行截图 == （AC代码截图，至少包含有"Accepted"） ==



#44751446提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def bfs(start, end, buckets):
    queue = [(start, [start])]
    visited = set(start)
    while queue:
        word, path = queue.pop(0)
        if word == end:
            return ' '.join(path)
        for i in range(len(word)):
            bucket = word[:i] + '_' + word[i+1:]
```

基本信息

#: 44751446
题目: 28046
提交人: 2200013720
内存: 5840kB
时间: 51ms
语言: Python3
提交时间: 2024-04-22 15:32:10

28050: 骑士周游

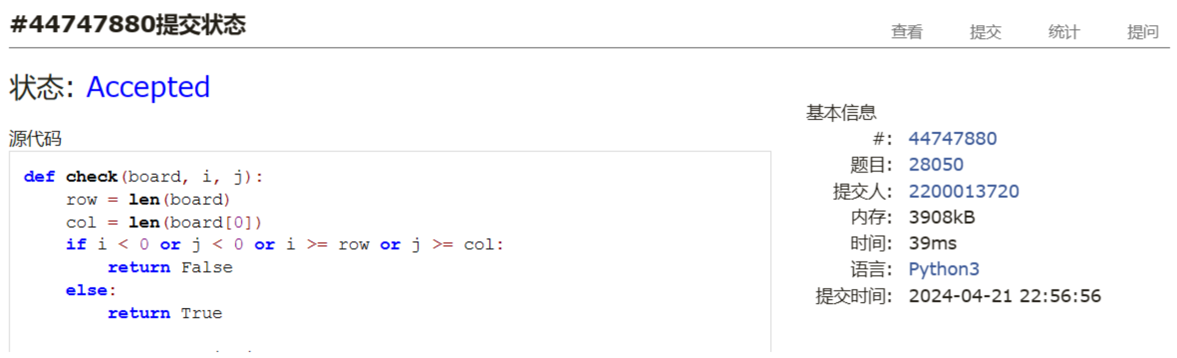
dfs, http://cs101.openjudge.cn/practice/28050/

思路：一开始用经典的dfs思路，然后TLE了。去网上查了下发现需要启发式算法，于是去了解了一下，感觉很有意思，但是不明白其中的原理所以感觉一知半解，如果考试遇到需要启发式算法的题我可能还是做不出来。不过理解了Warnsdorff启发式之后还是能比较快地写出来（需要注意的是，change使用那一步需要用newmove，以防覆盖原来的move导致总是输出fail，因为这个debug了十多分钟）。最后的用时真的令人震惊，启发式算法的优化太强了。

代码

```
#
def check(board, i, j):
    row = len(board)
    col = len(board[0])
    if i < 0 or j < 0 or i >= row or j >= col:
        return False
    else:
        return True

def change(board, i, j, move):
    res_list = []
    for m in move:
        new_i, new_j = i+m[0], j+m[1]
        if not check(board, new_i, new_j) or board[new_i][new_j] == 1:
            pass
        else:
            count = 0
            for n in move:
                if check(board, new_i+n[0], new_j+n[1]) and board[new_i+n[0]]
[new_j+n[1]] == 0:
                    count += 1
            res_list.append((count, m))
    res_list.sort(key = lambda x: x[0])
    return [y[1] for y in res_list]


def dfs(board, i, j, move, count, result, x, y):
    row = len(board)
    col = len(board[0])
    if i < 0 or j < 0 or i >= row or j >= col or result:
        return result
    if board[i][j] == 1:
        return result
    else:
        board[i][j] = 1
        count += 1
        if count == row*col:
            return True
        newmove = change(board, i, j, move) ## 新建move，不能覆盖原来的move
        for m in newmove:
            result = dfs(board, i+m[0], j+m[1], move, count, result, x, y)
            if result:
                return True
        if not i == x or not j == y:
            board[i][j] = 0
            count -= 1
        return result

n = int(input())
x, y = map(int,input().split())
board = []
move = [[-2,-1],[-2,1],[2,-1],[2,1],[-1,-2],[-1,2],[1,-2],[1,2]]
for i in range(n):
    board.append([0]*n)
judge = dfs(board, x, y, move, 0, False, x, y)
print(['fail','success'][judge])
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44747880提交状态

状态: Accepted

源代码

```
def check(board, i, j):
    row = len(board)
    col = len(board[0])
    if i < 0 or j < 0 or i >= row or j >= col:
        return False
    else:
        return True
```

基本信息

#: 44747880
题目: 28050
提交人: 2200013720
内存: 3908kB
时间: 39ms
语言: Python3
提交时间: 2024-04-21 22:56:56

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。==

这次作业做得断断续续地，算是复习了下dfs和图，收获可能就是桶的实现与应用以及启发式算法了吧，也给自己一个小小的挑战，尝试不用建树把有关树的前后序输出的题解答一遍。

期中周终于过去了，我最艰难的一段时期也过去了，所以周末给自己放了个假，没怎么管作业的事情，偶尔会拿起作业里的某道题看看想想，完全放松了三天。总算是有时间把数算的进度提一提了，好多每日选做都还没做，要加把劲赶上大部队了。尽管等我冷静下来后才发觉，接下来的半个学期我也不会过得很轻松如意，但是已经度过了最艰难的时期了，还怕什么？

最高兴的事情，莫过于终于有心思离开电脑，出去外面看看花了。