

Assignment #9: 图论：遍历，及 树算

Updated 1739 GMT+8 Apr 14, 2024

2024 spring, Compiled by 郑铭毅 数学科学学院

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

Windows 11

PyCharm

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：

代码

```
def tree_heights(s):
    old_height = 0
    max_old = 0
    new_height = 0
    max_new = 0
    stack = []
    for c in s:
        if c == 'd':
            old_height += 1
            max_old = max(max_old, old_height)
            new_height += 1
            stack.append(new_height)
            max_new = max(max_new, new_height)
        else:
            old_height -= 1
            new_height = stack[-1]
            stack.pop()
    return f"{max_old} => {max_new}"

s = input().strip()
print(tree_heights(s))
```

代码运行截图

OpenJudge
题目ID, 标题, 描述
2300010872
信箱
账号

CS101 / 数算pre每日选做
题目 排名 状态 提问

#44753914提交状态
查看 提交 统计 提问

状态: Accepted

源代码

```
def tree_heights(s):
    old_height = 0
    max_old = 0
    new_height = 0
    max_new = 0
    stack = []
    for c in s:
        if c == 'd':
            old_height += 1
            max_old = max(max_old, old_height)
            new_height += 1
            stack.append(new_height)
            max_new = max(max_new, new_height)
        else:
            old_height -= 1
            new_height = stack[-1]
            stack.pop()
    return f"{max_old} => {max_new}"
s = input().strip()
print(tree_heights(s))
```

基本信息
#: 44753914
题目: 04081
提交人: 2300010872
内存: 3656kB
时间: 29ms
语言: Python3
提交时间: 2024-04-22 19:13:12

©2002-2022 POJ 京ICP备20010980号-1
English 帮助 关于

08581: 扩展二叉树

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

思路:

```
class Treenode:
    def __init__(self, val):
        self.val = val
        self.left = None
        self.right = None
def build_tree(preorder):
    root_val = preorder.pop(0)
    if root_val == '.':
        return None
    root = Treenode(root_val)
    root.left = build_tree(preorder)
    root.right = build_tree(preorder)
    return root
def inorder_traversal(root):
    if not root:
        return []
    return inorder_traversal(root.left) + [root.val] + inorder_traversal(root.right)
def postorder_traversal(root):
    if not root:
        return []
    return postorder_traversal(root.left) + postorder_traversal(root.right) + [root.val]
preorder_seq = list(input().strip())
root = build_tree(preorder_seq)
inorder_seq = inorder_traversal(root)
postorder_seq = postorder_traversal(root)
print(''.join(inorder_seq))
print(''.join(postorder_seq))
```

代码运行截图

OpenJudge

题目ID, 标题, 描述

2300010872

信箱

账号

CS101 / 数算pre每日选做

题目

排名

状态

提问

#44754106提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
class Treenode:
    def __init__(self, val):
        self.val = val
        self.left = None
        self.right = None
    def build_tree(preorder):
        root_val = preorder.pop(0)
        if root_val == '.':
            return None
        root = Treenode(root_val)
        root.left = build_tree(preorder)
        root.right = build_tree(preorder)
        return root
    def inorder_traversal(root):
        if not root:
            return []
        return inorder_traversal(root.left) + [root.val] + inorder_traversal(root.right)
    def postorder_traversal(root):
        if not root:
            return []
        return postorder_traversal(root.left) + postorder_traversal(root.right) + [root.val]
preorder_seq = list(input().strip())
root = build_tree(preorder_seq)
inorder_seq = inorder_traversal(root)
postorder_seq = postorder_traversal(root)
print(' '.join(inorder_seq))
print(' '.join(postorder_seq))
```

基本信息

#: 44754106

题目: 08581

提交人: 2300010872

内存: 3656kB

时间: 30ms

语言: Python3

提交时间: 2024-04-22 19:25:59

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

22067: 快速堆猪

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

思路:

代码

```
1 #
2
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

04123: 马走日

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

思路:

代码

```
dx = [-1, -2, -2, -1, 1, 2, 2, 1]
dy = [-2, -1, 1, 2, 2, 1, -1, -2]

def is_valid_move(x, y, n, m, visited):
    return 0 <= x < n and 0 <= y < m and not visited[x][y]

def dfs(x, y, n, m, visited):
    if all(all(row) for row in visited):
        return 1

    count = 0

    for i in range(8):
        nx, ny = x + dx[i], y + dy[i]

        if is_valid_move(nx, ny, n, m, visited):
            visited[nx][ny] = True

            count += dfs(nx, ny, n, m, visited)

            visited[nx][ny] = False

    return count

T = int(input().strip())

for _ in range(T):
    n, m, x, y = map(int, input().strip().split())

    visited = [[False] * m for _ in range(n)]

    visited[x][y] = True

    result = dfs(x, y, n, m, visited)

    print(result)
```

代码运行截图

OpenJudge

题目ID, 标题, 描述

2300010872

信封

题号

CS101 / 题库

题目 排名 状态 提问

#44754437提交状态

查看 提交 统计 提问

状态: Accepted

源代码

dx = [-1, -2, -2, -1, 1, 2, 2, 1]
dy = [-2, -1, 1, 2, 2, 1, -1, -2]
def is_valid_move(x, y, n, m, visited):
 return 0 <= x < n and 0 <= y < m and not visited[x][y]
def dfs(x, y, n, m, visited):
 if all(all(row) for row in visited):
 return 1
 count = 0
 for i in range(8):
 nx, ny = x + dx[i], y + dy[i]
 if is_valid_move(nx, ny, n, m, visited):
 visited[nx][ny] = True
 count += dfs(nx, ny, n, m, visited)
 visited[nx][ny] = False
 return count
T = int(input().strip())
for _ in range(T):
 n, m, x, y = map(int, input().strip().split())
 visited = [[False] * m for _ in range(n)]
 visited[x][y] = True
 result = dfs(x, y, n, m, visited)
 print(result)

基本信息
#: 44754437
题目: 04123
提交人: 2300010872
内存: 3684kB
时间: 5128ms
语言: Python3
提交时间: 2024-04-22 19:46:18

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

28046: 词梯

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

思路:

代码

```
1 #
2
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

28050: 骑士周游

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

思路:

代码

```

dx = [-1, -2, -2, -1, 1, 2, 2, 1]
dy = [-2, -1, 1, 2, 2, 1, -1, -2]

def is_valid_move(x, y, n, visited):
    return 0 <= x < n and 0 <= y < n and not visited[x][y]

def get_valid_moves(x, y, n, visited):
    valid_moves = []
    for i in range(8):
        nx, ny = x + dx[i], y + dy[i]
        if is_valid_move(nx, ny, n, visited):
            count = 0
            for j in range(8):
                tx, ty = nx + dx[j], ny + dy[j]
                if is_valid_move(tx, ty, n, visited):
                    count += 1
            valid_moves.append((count, nx, ny))
    valid_moves.sort()
    return [(nx, ny) for _, nx, ny in valid_moves]

def knights_tour(n, sr, sc):
    visited = [[False] * n for _ in range(n)]
    visited[sr][sc] = True
    count = 1
    x, y = sr, sc
    while count < n * n:
        next_moves = get_valid_moves(x, y, n, visited)
        if not next_moves:
            return False
        nx, ny = next_moves[0]
        visited[nx][ny] = True
        count += 1
        x, y = nx, ny
    return True

n = int(input().strip())
sr, sc = map(int, input().strip().split())
if knights_tour(n, sr, sc):
    print("success")
else:
    print("fail")

```

代码运行截图

CS101 / 题库

题目 排名 状态 题解

#44754572提交状态

状态: Accepted

基础信息

语言:

提交:

通过:

统计

题解

语言:

提交:

通过:

统计

题解

源代码

```
def is_valid_move(x, y, m, nx, ny):  
    dx = [-1, -2, -2, -1, 1, 2, 2, 1]  
    dy = [-1, -1, 0, 1, 1, 2, 2, -1, -2]  
    if is_valid_move(nx, ny, m, x, y):  
        return True  
    return 0 <= x < n and 0 <= y < n and not visited[nx][ny]  
  
def valid_move(x, y, m, nx, ny, visited):  
    for i in range(8):  
        dx, dy = dx[i], dy[i]  
        if is_valid_move(nx, ny, visited):  
            count += 1  
  
    for j in range(8):  
        tx, ty = tx[j], ty[j]  
        if is_valid_move(tx, ty, m, visited):  
            count += 1  
  
    valid_moves.append((count, mx, my))  
  
valid_moves.sort()  
return (mx, my) == (n-1, n-1)  
  
def KnightTour(n):  
    visited = [[False] * n for _ in range(n)]  
    visited[0][0] = True  
    count = 1  
    mx, my = 0, 0  
    while count < n * n:  
        next_moves = get_valid_moves(mx, my, visited)  
        if not next_moves:  
            return False  
        nx, ny = next_moves[0]  
        visited[nx][ny] = True  
        count += 1  
        mx, my = nx, ny  
    return True  
  
n = int(input().strip())  
sr, sc = map(int, input().strip().split())  
if KnightTour(n, sr, sc):  
    print("Success")  
else:  
    print("Fail")
```

编译选项

g++ -std=c++11 -O2 -DDEBUG -I./lib -lstdc++ -lm -pthread

评测记录

测试点

得分

耗时

内存

提交时间

评测时间

1

100%

0.01s

1024K

2024-04-22 19:51:47

2024-04-22 19:51:47

2. 学习总结和收获

题目对我来说还是相当有难度的，dfs算法还是有点难以理解。期中周过后应该会花更多时间复习一下前面的树之类的，以及再看看这两周的题目，学习一下算法。