

Assignment #A: 图论：算法，树算及栈

Updated 2018 GMT+8 Apr 21, 2024

2024 spring, Compiled by 陈奕好 工学院

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：macOS Sonoma 14.4 (23E214)

Python编程环境：PyCharm 2023.3.1 (Professional Edition)

1. 题目

20743: 整人的提词本

<http://cs101.openjudge.cn/practice/20743/>

思路：直接折叠，偶数次括号内折叠不变，奇数次折叠reverse，这里把括号内的反复折叠，结果不变

代码

```
1 string = input()
2 opt = []
3 turn = 0
4 for i in range(len(string)):
5     if string[i] == "(":
6         opt.append(i)
7         turn += 1
8     elif string[i] == ")":
9         left = opt.pop()
```

```

10         string = string[:left] + " " + string[left+1:i][::-1] + " " + string[i+1:]
11         turn -= 1
12     print("".join(string.split()))
13     # Path: 20744:整人的提词本.py
14
15

```

代码运行截图 (至少包含有"Accepted")

#44784468提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```

string = input()
opt = []
s = ""
turn = 0
for i in range(len(string)):
    if string[i] == "(":
        opt.append(i)
        turn += 1
    elif string[i] == ")":
        left = opt.pop()
        string = string[:left] + " " + string[left+1:i][::-1] + " " + string[i+1:]
        turn -= 1
print("".join(string.split()))
# Path: 20744:整人的提词本.py

```

基本信息

#: 44784468
 题目: 20743
 提交人: 23n2300011030(陈奕好)
 内存: 3612kB
 时间: 27ms
 语言: Python3
 提交时间: 2024-04-24 14:18:05

©2002–2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

02255: 重建二叉树

<http://cs101.openjudge.cn/practice/02255/>

思路：用了index，先前序找根，在mid分树。

代码

```

1 class TreeNode:
2     def __init__(self, value):
3         self.val = value
4         self.left = None
5         self.right = None
6
7
8 def rebuild(pre, mid):
9     if not pre:
10         return None
11     node = TreeNode(pre[0])

```

```

12     k = mid.index(pre[0])
13     node.left = rebuild(pre[1:k+1], mid[:k])
14     node.right = rebuild(pre[k+1:], mid[k+1:])
15     return node
16
17
18 def postOrder(root):
19     if root is None:
20         return []
21     return postOrder(root.left) + postOrder(root.right) + [root.val]
22
23
24 while True:
25     try:
26         pre, mid = input().split()
27         root = rebuild(pre, mid)
28         print("".join(postOrder(root)))
29     except EOFError:
30         break
31
32

```

代码运行截图 (至少包含有"Accepted")

#44798886提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```

class TreeNode:
    def __init__(self, value):
        self.val = value
        self.left = None
        self.right = None

def rebuild(pre, mid):
    if not pre:
        return None
    node = TreeNode(pre[0])
    k = mid.index(pre[0])
    node.left = rebuild(pre[1:k+1], mid[:k])
    node.right = rebuild(pre[k+1:], mid[k+1:])
    return node

def postOrder(root):
    if root is None:
        return []
    return postOrder(root.left) + postOrder(root.right) + [root.val]

while True:
    try:
        pre, mid = input().split()
        root = rebuild(pre, mid)

```

基本信息

#: 44798886
 题目: 02255
 提交人: 23n2300011030(陈奕好)
 内存: 3536kB
 时间: 33ms
 语言: Python3
 提交时间: 2024-04-26 08:45:44

01426: Find The Multiple

<http://cs101.openjudge.cn/practice/01426/>

要求用bfs实现

思路：bfs的一种强剪枝，没想到余数法这么好用

代码

```
1  from collections import deque
2  while 1:
3      n=int(input())
4      if n==0:
5          break
6      q=deque([(1, '1')])
7      vis={1}
8      while q:
9          remainder,num=q.popleft()
10         if remainder==0:
11             print(num)
12             break
13         for digit in [0,1]:
14             new_remainder=(remainder*10+digit)%n
15             if new_remainder not in vis:
16                 vis.add(new_remainder)
17                 q.append((new_remainder,num+str(digit)))
18
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```
from collections import deque
while 1:
    n=int(input())
    if n==0:
        break
    q=deque([(1, '1')])
    vis={1}
    while q:
        remainder,num=q.popleft()
        if remainder==0:
            print(num)
            break
        for digit in [0,1]:
            new_remainder=(remainder*10+digit)%n
            if new_remainder not in vis:
                vis.add(new_remainder)
                q.append((new_remainder,num+str(digit)))
```

基本信息

#: 44784632
题目: 01426
提交人: 23n2300011030(陈奕好)
内存: 3628kB
时间: 54ms
语言: Python3
提交时间: 2024-04-24 14:42:02

04115: 鸣人和佐助

bfs, <http://cs101.openjudge.cn/practice/04115/>

思路：大于号加上=就可过了，=的情况太多了

代码

```
1  # 真就拯救行动了
2  import heapq
3  def find(matrix, N, M):
4      for i in range(N):
5          for j in range(M):
6              if matrix[i][j] == '@':
7                  return i, j
8      return -2, -2
9
10
11 def bfs(matrix, N, M, T, i, j):
12     dir = [(0, 1), (0, -1), (1, 0), (-1, 0)]
13     queue = [(0, i, j, T)]
14     heapq.heapify(queue)
15     visited = [[-1] * M for _ in range(N)]
16     visited[i][j] = T
17     while queue:
18         step, i, j, cha = heapq.heappop(queue)
19         for dx, dy in dir:
20             x, y = i + dx, j + dy
21             if 0 <= x < N and 0 <= y < M and matrix[x][y] == '+':
22
```

```

23         return step + 1
24
25     if 0 <= x < N and 0 <= y < M:
26         if matrix[x][y] == '#' and cha > 0:
27             if visited[x][y] >= cha - 1:
28                 continue
29             else:
30                 heapq.heappush(queue, (step + 1, x, y, cha - 1))
31                 visited[x][y] = cha - 1
32         elif matrix[x][y] == '*':
33             if visited[x][y] >= cha:
34                 continue
35             else:
36                 heapq.heappush(queue, (step + 1, x, y, cha))
37                 visited[x][y] = cha
38
39     return -1
40
41 while True:
42     try:
43         N, M, T = map(int, input().split())
44         matrix = [list(map(str, input())) for _ in range(N)]
45         i, j = find(matrix, N, M)
46         res = bfs(matrix, N, M, T, i, j)
47         print(res)
48     except EOFError:
49         break
50
51

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```
# 真就拯救行动了
import heapq
def find(matrix, N, M):
    for i in range(N):
        for j in range(M):
            if matrix[i][j] == '@':
                return i, j
    return -2, -2

def bfs(matrix, N, M, T, i, j):
    dir = [(0, 1), (0, -1), (1, 0), (-1, 0)]
    queue = [(0, i, j, T)]
    heapq.heapify(queue)
    visited = [[-1] * M for _ in range(N)]
    visited[i][j] = T
    while queue:
        step, i, j, cha = heapq.heappop(queue)
        for dx, dy in dir:
            x, y = i + dx, j + dy
            if 0 <= x < N and 0 <= y < M and matrix[x][y] == '+':
                return step + 1

            if 0 <= x < N and 0 <= y < M:
                if matrix[x][y] == '#' and cha > 0:
                    if visited[x][y] >= cha - 1:
                        continue
                    else:
                        heapq.heappush(queue, (step + 1, x, y, cha - 1))
                        visited[x][y] = cha - 1
                elif matrix[x][y] == '*':
                    if visited[x][y] >= cha:
                        continue
                    else:
```

基本信息

#: 44799016

题目: 04115

提交人: 23n2300011030(陈奕好)

内存: 4168kB

时间: 91ms

语言: Python3

提交时间: 2024-04-26 09:18:39

20106: 走山路

Dijkstra, <http://cs101.openjudge.cn/practice/20106/>

思路：老题了，主要是改推进条件

代码

```
1  from heapq import heappop, heappush
2
3
4  def bfs(x1, y1):
5      q = [(0, x1, y1)] # 初始化堆
6      v = set() # 走过的路径
7      while q:
8          t, x, y = heappop(q) # 省略建堆
9          v.add((x, y))
```

```

10         if x == x2 and y == y2:
11             return t
12         for dx, dy in dir: # 这里把 (x,y) 因为的所有情况走完，因为是算高度差，所以一步走到已经
                                是最优路径。
13             nx, ny = x+dx, y+dy
14             if 0 <= nx < m and 0 <= ny < n and ma[nx][ny] != '#' and (nx, ny) not in
v: # can_visit() 模版, if 不能则不返回堆
15                 nt = t+abs(int(ma[nx][ny])-int(ma[x][y])) # 根据题意加上相对高度差
16                 heappush(q, (nt, nx, ny)) # bfs 压入堆 (每次处理优先处理当前最优解，但还是贪
心)
17         return 'NO'
18
19
20 # 主程序
21 m, n, p = map(int, input().split())
22 ma = [list(input().split()) for _ in range(m)]
23 dir = [(1, 0), (-1, 0), (0, 1), (0, -1)] # 移动方向
24 for _ in range(p):
25     x1, y1, x2, y2 = map(int, input().split())
26     if ma[x1][y1] == '#' or ma[x2][y2] == '#': # 题目给的补充死亡条件
27         print('NO')
28         continue
29     print(bfs(x1, y1))
30

```

代码运行截图 (AC代码截图，至少包含有"Accepted")

状态: Accepted

源代码

```
from heapq import heappop, heappush

def bfs(x1, y1):
    q = [(0, x1, y1)] # 初始化堆
    v = set() # 走过的路径
    while q:
        t, x, y = heappop(q) # 省略建堆
        v.add((x, y))
        if x == x2 and y == y2:
            return t
        for dx, dy in dir: # 这里把 (x,y) 因为的所有情况走完, 因为是算高度差, 所以
            nx, ny = x+dx, y+dy
            if 0 <= nx < m and 0 <= ny < n and ma[nx][ny] != '#' and (nt := t+abs(int(ma[nx][ny])-int(ma[x][y]))) # 根据题意加上
            heappush(q, (nt, nx, ny)) # bfs压入堆 (每次处理优先处理当前

    return 'NO'

# 主程序
m, n, p = map(int, input().split())
ma = [list(input().split()) for _ in range(m)]
dir = [(1, 0), (-1, 0), (0, 1), (0, -1)] # 移动方向
for _ in range(p):
    x1, y1, x2, y2 = map(int, input().split())
    if ma[x1][y1] == '#' or ma[x2][y2] == '#': # 题目给的补充死亡条件
        print('NO')
        continue
    print(bfs(x1, y1))
```

基本信息

#: 43297352
题目: 20106
提交人: 23n2300011030(陈奕好)
内存: 4152kB
时间: 1082ms
语言: Python3
提交时间: 2023-12-22 18:23:40

05442: 兔子与星空

Prim, <http://cs101.openjudge.cn/practice/05442/>

思路：快五一了，这道题先看的题解，这是一道bfs+贪心的树遍历问题，heap使用优化了路径。

代码

```
1 import heapq
2
3 def prim(graph, start):
4     mst = []
5     used = set([start]) # 已经使用过的点
6     edges = [
7         (cost, start, to)
8         for to, cost in graph[start].items()
9     ] # (cost, frm, to) 的列表
10    heapq.heapify(edges) # 转换成最小堆
11
```

```

12     while edges: # 当还有边可以选择时
13         cost, frm, to = heapq.heappop(edges) # 弹出最小边
14         if to not in used: # 如果这个点还没被使用过
15             used.add(to) # 标记为已使用
16             mst.append((frm, to, cost)) # 加入到最小生成树中
17             for to_next, cost2 in graph[to].items(): # 将与这个点相连的边加入到堆中
18                 if to_next not in used: # 如果这个点还没被使用过
19                     heapq.heappush(edges, (cost2, to, to_next)) # 加入到堆中
20
21     return mst # 返回最小生成树
22
23 n = int(input())
24 graph = {chr(i+65): {} for i in range(n)}
25 for i in range(n-1):
26     data = input().split()
27     node = data[0]
28     for j in range(2, len(data), 2):
29         graph[node][data[j]] = int(data[j+1])
30         graph[data[j]][node] = int(data[j+1])
31
32 mst = prim(graph, 'A') # 从A开始生成最小生成树
33 print(sum([cost for frm, to, cost in mst])) # 输出最小生成树的总权值

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: **Accepted**

源代码

```
import heapq

def prim(graph, start):
    mst = []
    used = set([start]) # 已经使用过的点
    edges = [
        (cost, start, to)
        for to, cost in graph[start].items()
    ] # (cost, frm, to) 的列表
    heapq.heapify(edges) # 转换成最小堆

    while edges: # 当还有边可以选择时
        cost, frm, to = heapq.heappop(edges) # 弹出最小边
        if to not in used: # 如果这个点还没被使用过
            used.add(to) # 标记为已使用
            mst.append((frm, to, cost)) # 加入到最小生成树中
            for to_next, cost2 in graph[to].items(): # 将与这个点相连的边
                if to_next not in used: # 如果这个点还没被使用过
                    heapq.heappush(edges, (cost2, to, to_next)) # 加入到

    return mst # 返回最小生成树

n = int(input())
graph = {chr(i+65): {} for i in range(n)}
for i in range(n-1):
    data = input().split()
    node = data[0]
    for j in range(2, len(data), 2):
        graph[node][data[j]] = int(data[j+1])
        graph[data[j]][node] = int(data[j+1])

mst = prim(graph, 'A') # 从A开始生成最小生成树
print(sum([cost for frm, to, cost in mst])) # 输出最小生成树的总权值
```

基本信息

#: 44799072
题目: 05442
提交人: 23n2300011030(陈奕好)
内存: 4024kB
时间: 30ms
语言: Python3
提交时间: 2024-04-26 09:29:25

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。

感觉再这样下去就会荒废了，数学压力比上学期还大，数算不能做陪葬品，51要狠狠的卷。