Assignment #A: 图论：遍历，树算及栈

Updated 2018 GMT+8 Apr 21, 2024

2024 spring, Complied by 夏天、生命科学学院

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

操作系统：Windows 10 家庭版

Python编程环境：Spyder （python 3.11）

1. 题目

20743: 整人的提词本

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

思路：用栈和队列，遇到右括号就开始弹出栈顶元素并入队直到栈顶元素是左括号为止，然后弹出左括号，队列中的元素按顺序入栈

代码

```
from collections import deque
s=input()
stack=[]
queue=deque([])
for char in s:
    if char!=')':
        stack.append(char)
    else:
        while stack[-1]!='(':
            queue.append(stack.pop())
        stack.pop()
        while queue:
            stack.append(queue.popleft())
print(''.join(stack))
```

代码运行截图 （至少包含有"Accepted"）

状态：Accepted

源代码

```
from collections import deque
s=input()
stack=[]
queue=deque([])
for char in s:
    if char!=')':
        stack.append(char)
    else:
        while stack[-1]!='(':
            queue.append(stack.pop())
        stack.pop()
        while queue:
            stack.append(queue.popleft())
print(''.join(stack))
```

基本信息

#: 44762745
题目: 20743
提交人: 23n2300012289
内存: 3604kB
时间: 27ms
语言: Python3
提交时间: 2024-04-23 15:00:41

02255: 重建二叉树

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

思路：作业布置重了？之前叫根据二叉树前中序序列建树，现在换了个名字（

代码

```
def build_tree(prefix,infix):
    if not prefix or not infix:
        return []
    root=prefix[0]
    root_index=infix.index(root)
    left_infix=infix[:root_index]
    right_infix=infix[root_index+1:]
    left_prefix=prefix[1:len(left_infix)+1]
    right_prefix=prefix[len(left_infix)+1:]
    tree=[]
    tree.extend(build_tree(left_prefix,left_infix))
    tree.extend(build_tree(right_prefix,right_infix))
    tree.append(root)
    return tree
while True:
    try:
        prefix,infix=input().split()
        print(''.join(build_tree(prefix,infix)))
    except EOFError:
        break
```

代码运行截图 （至少包含有"Accepted"）


状态: Accepted

源代码

```
def build_tree(prefix,infix):
    if not prefix or not infix:
        return []
    root=prefix[0]
    root_index=infix.index(root)
    left_infix=infix[:root_index]
    right_infix=infix[root_index+1:]
    left_prefix=prefix[1:len(left_infix)+1]
    right_prefix=prefix[len(left_infix)+1:]
    tree=[]
    tree.extend(build_tree(left_prefix,left_infix))
    tree.extend(build_tree(right_prefix,right_infix))
    tree.append(root)
    return tree
while True:
    try:
        prefix,infix=input().split()
        print(''.join(build_tree(prefix,infix)))
    except EOFError:
        break
```

基本信息

#:	44763003
题目:	02255
提交人:	23n2300012289
内存:	7304kB
时间:	30ms
语言:	Python3
提交时间:	2024-04-23 15:23:16

01426: Find The Multiple

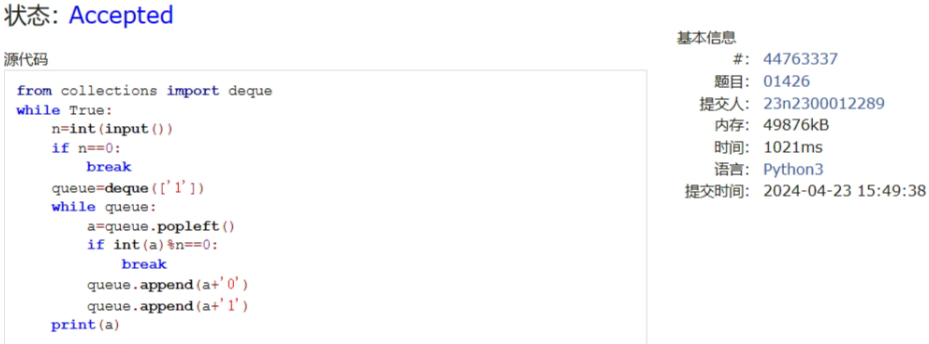
http://cs101.openjudge.cn/practice/01426/ 要求用bfs实现

思路：样例输出似乎是故意给了看起来最吓人的那一种，实际上就是很常规的bfs

代码

```
from collections import deque
while True:
    n=int(input())
    if n==0:
        break
    queue=deque(['1'])
    while queue:
        a=queue.popleft()
        if int(a)%n==0:
            break
        queue.append(a+'0')
        queue.append(a+'1')
    print(a)
```

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 True:
    n=int(input())
    if n==0:
        break
    queue=deque(['1'])
    while queue:
        a=queue.popleft()
        if int(a)%n==0:
            break
        queue.append(a+'0')
        queue.append(a+'1')
    print(a)
```

基本信息

#:	44763337
题目:	01426
提交人:	23n2300012289
内存:	49876kB
时间:	1021ms
语言:	Python3
提交时间:	2024-04-23 15:49:38

04115: 鸣人和佐助

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

思路：本以为是常规的bfs多加了一个参数，没想到就因为加了这个限制，代码疯狂地MLE.导致MLE的原因大概有两个：visited占内存，队列占内存。最后通过先判断是否以相同查克拉经过同一个位置再决定是否入队以减少内存

代码

```
from collections import deque
M,N,T=map(int,input().split())
board=[]
for _ in range(M):
    line=input()
    board.append(line)
    if '@' in line:
        sx,sy=_,line.index('@')
queue=deque([(sx,sy,T,0)])
visited={(sx,sy,T)}
flag=False
while queue:
    x,y,T,time=queue.popleft()
    if board[x][y]=='+':
        flag=True
        break
    directions=[(0,1),(0,-1),(1,0),(-1,0)]
    for dx,dy in directions:
        nx,ny=x+dx,y+dy
        if nx<0 or nx>=M or ny<0 or ny>=N:
            continue
        if board[nx][ny]=='#':
            if T>0 and (nx,ny,T-1) not in visited:
                queue.append((nx,ny,T-1,time+1))
                visited.add((nx,ny,T-1))
        elif (nx,ny,T) not in visited:
            queue.append((nx,ny,T,time+1))
            visited.add((nx,ny,T))
if flag:
    print(time)
else:
    print(-1)
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

20106: 走山路

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

思路：再走山路，上学期刚开始还不知道为什么用堆而不用队列；这学期没有一丝犹豫直接用堆

代码

```
import heapq
m,n,p=map(int,input().split())
board=[list((input().split()))for _ in range(m)]
for _ in range(p):
    flag=False
    sx,sy,ex,ey=map(int,input().split())
    if board[sx][sy]=='#' or board[ex][ey]=='#':
        print('NO')
        continue
    visited=[[False]*n for _ in range(m)]
    heap=[]
    heapq.heappush(heap,(0,sx,sy))
    while heap:
        energy,x,y=heapq.heappop(heap)
        if x==ex and y==ey:
            flag=True
            break
        visited[x][y]=True
        for dx,dy in [(1,0),(-1,0),(0,1),(0,-1)]:
            nx,ny=dx+x,dy+y
            if nx<0 or nx>=m or ny<0 or ny>=n or visited[nx][ny] or board[nx][ny]=='#':
                continue
            heapq.heappush(heap,(energy+abs(int(board[x][y])-int(board[nx][ny])),nx,ny))
    if flag:
        print(energy)
    else:
        print('NO')
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

05442: 兔子与星空

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

思路：两种算法可解决最小生成树问题

Prim算法：建立最小生成树时，将顶点按是否已包含在树中分为A,B两类。初始状态所有点都属于B类，然后任取一个点作为起始点，将它移至A类，在B类中查找与起始点相连且权值最小的点，再将该点移至A类。每次都从B类中查找与A类中的点相连且权值最小的点直到B类为空为止

Kruskal算法：将所有权值按升序排列，每次对最小权值进行判断，如果不形成环就添加；否则不添加。是否形成环要用到并查集(回头找时间尝试用这个算法做这道题)

代码

```
from collections import defaultdict
n=int(input())
Graph=defaultdict(dict)
for _ in range(n-1):
    edges=input().split()
    start_vertex=edges[0]
    edge_num=int(edges[1])
    for i in range(1,edge_num+1):
        end_vertex=edges[2*i]
        weight=int(edges[2*i+1])
        Graph[start_vertex][end_vertex]=weight
        Graph[end_vertex][start_vertex]=weight
total_weight=0
visited=set('A')
while len(visited)<len(Graph):
    min_weight=float('inf')
    min_edge=None
    for node in visited:
        for edge in Graph[node]:
            if edge not in visited:
                if Graph[node][edge]<min_weight:
                    min_weight=Graph[node][edge]
                    min_edge=edge
    if min_edge:
        total_weight+=min_weight
        visited.add(min_edge)
print(total_weight)
```

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 defaultdict
n=int(input())
Graph=defaultdict(dict)
for _ in range(n-1):
    edges=input().split()
    start_vertex=edges[0]
    edge_num=int(edges[1])
    for i in range(1,edge_num+1):
        end_vertex=edges[2*i]
        weight=int(edges[2*i+1])
        Graph[start_vertex][end_vertex]=weight
        Graph[end_vertex][start_vertex]=weight
total_weight=0
visited=set('A')
while len(visited)<len(Graph):
    min_weight=float('inf')
    min_edge=None
    for node in visited:
        for edge in Graph[node]:
            if edge not in visited:
                if Graph[node][edge]<min_weight:
                    min_weight=Graph[node][edge]
                    min_edge=edge
    if min_edge:
        total_weight+=min_weight
        visited.add(min_edge)
print(total_weight)
```

基本信息

#:	44768476
题目:	05442
提交人:	23n2300012289
内存:	3680kB
时间:	25ms
语言:	Python3
提交时间:	2024-04-23 21:10:18

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。

滚动复习栈和树有助于提高敲相关代码的熟练度（确信）

可能是上学期学过dfs、bfs的缘故，这次作业有关搜索的题目除了鸣人和佐助优化时卡了点时间，其他很快就完成了

于是把省出来的时间用来学习最小生成树，又学到了两个新算法（Kruskal算法还没来得及实践，下次一定！）