Assignment #8: 图论：概念、遍历，及 树算

Updated 1919 GMT+8 Apr 8, 2024

2024 spring, Complied by ==张宇帆 心理与认知科学学院==

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

==（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

19943: 图的拉普拉斯矩阵

matrices, http://cs101.openjudge.cn/practice/19943/

请定义Vertex类，Graph类，然后实现

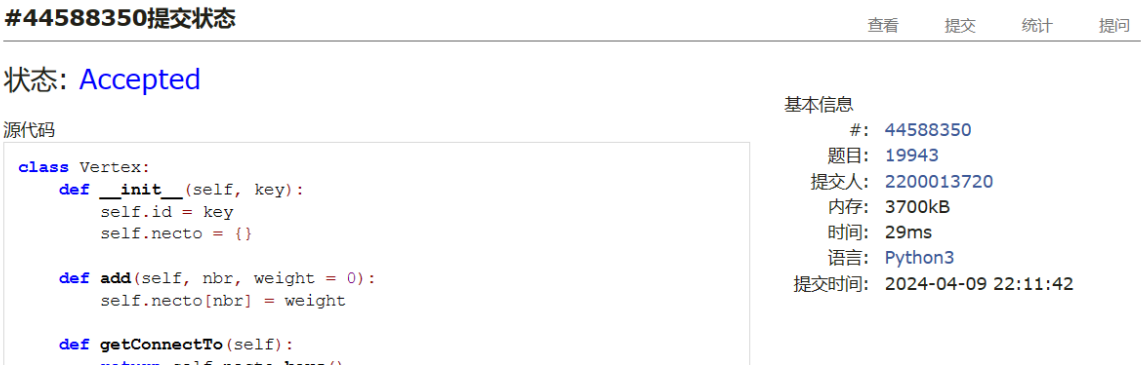
思路：朴实无华的图类创建（翻了翻之前写的，感觉好像粗暴点也差不多）

代码

```
#
class Vertex:
    def __init__(self, key):
        self.id = key
        self.necto = {}

    def add(self, nbr, weight = 0):
        self.necto[nbr] = weight

    def getConnectTo(self):
        return self.necto.keys()
```

```
    def getnum(self):
        return len(self.necto)

class Graph:
    def __init__(self):
        self.idlist = {}
        self.num = 0

    def addVertex(self, key):
        self.idlist[key] = Vertex(key)
        self.num += 1

    def addEdge(self, fr, to, weight = 0):
        if fr not in self.idlist:
            self.addVertex(fr)
        if to not in self.idlist:
            self.addVertex(to)
        self.idlist[fr].add(to, weight)

n,m = map(int,input().split())
graph = Graph()
for _ in range(m):
    a,b = map(int,input().split())
    graph.addEdge(a, b)
    graph.addEdge(b, a)
for i in range(n):
    result = ['0']*n
    if i in graph.idlist:
        target = graph.idlist[i]
        result[i] = str(target.getnum())
        connected = target.getConnectTo()
        for c in connected:
            result[c] = '-1'
    print(' '.join(result))
```

代码运行截图 ==（至少包含有"Accepted"）==



18160: 最大连通域面积

matrix/dfs similar, http://cs101.openjudge.cn/practice/18160

思路：这道题感觉最让人舒服的地方就是，可以直接把遍历过的点换成"."从而防止回溯

代码

```
#
def dfs(i, j, matrix, S, move):
    rowmax = len(matrix)
    colmax = len(matrix[0])
    if i >= rowmax or i < 0 or j >= colmax or j < 0:
        return S
    elif matrix[i][j] == '.':
        return S
    else:
        matrix[i][j] = '.' #直接把他变为.以防止回溯
        S += 1
        for m in move:
            S = dfs(i+m[0], j+m[1], matrix, S, move)
        return S

move = [[-1,-1],[-1,0],[-1,1],[0,-1],[0,0],[0,1],[1,-1],[1,0],[1,1]]
T = int(input())
for _ in range(T):
    matrix = []
    N,M = map(int,input().split())
    s = 0
    for row in range(N):
        matrix.append(list(map(str,input())))
    for i in range(N):
        for j in range(M):
            if matrix[i][j] == 'W':
                S = dfs(i, j, matrix, 0, move)
                s = max(s, S)
    print(s)
```

代码运行截图 ==（至少包含有"Accepted"）==

状态: Accepted

源代码

```
def dfs(i, j, matrix, S, move):
    rowmax = len(matrix)
    colmax = len(matrix[0])
    if i >= rowmax or i < 0 or j >= colmax or j < 0:
        return S
    elif matrix[i][j] == '.':
        return S
    else:
        matrix[i][j] = '.' #直接把他变为.以防止回溯
        S += 1
        for m in move:
            S = dfs(i+m[0], j+m[1], matrix, S, move)
        return S
```

基本信息

#:	44588529
题目:	18160
提交人:	2200013720
内存:	3828kB
时间:	133ms
语言:	Python3
提交时间:	2024-04-09 22:27:05

sy383: 最大权值连通块

https://sunnywhy.com/sfbj/10/3/383

思路：这个能直观地感受到图的便利之处，而且其实还可以再改进点：在findmax中创建字典储存同一连通线上的点，不过这个实现起来可能有点绕）

代码

```
#
class Vertex:
    def __init__(self, key, val):
        self.id = key
        self.necto = {}
        self.val = val

    def add(self, nbr, weight = 0):
        self.necto[nbr] = weight

    def getConnectTo(self):
        return self.necto.keys()

    def getnum(self):
        return len(self.necto)

class Graph:
    def __init__(self):
        self.idlist = {}
        self.num = 0

    def addVertex(self, key, key_val):
        self.idlist[key] = Vertex(key, key_val)
        self.num += 1

    def addEdge(self, fr, to, fr_val, to_val, weight = 0):
        if fr not in self.idlist:
            self.addVertex(fr, fr_val)
        if to not in self.idlist:
            self.addVertex(to, to_val)
        self.idlist[fr].add(to, weight)

def findmax(Vertex, have, graph):
    res = 0
    if Vertex.id not in have:
        res += Vertex.val
        have.add(Vertex.id)
        connect = Vertex.getConnectTo()
        for c in connect:
            target = graph.idlist[c]
            if target.id not in have:
                res += findmax(target, have, graph)
                have.add(c)
```

```
    return res


n,m = map(int,input().split())
graph = Graph()
value = list(map(int,input().split()))
for i in range(n):
    graph.addVertex(i, value[i])
for _ in range(m):
    a,b = map(int,input().split())
    graph.addEdge(a, b, value[a], value[b])
    graph.addEdge(b, a, value[b], value[a])
result = 0
for i in range(n):
    res = findmax(graph.idlist[i], set(), graph)
    result = max(res, result)
print(result)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

03441: 4 Values whose Sum is 0

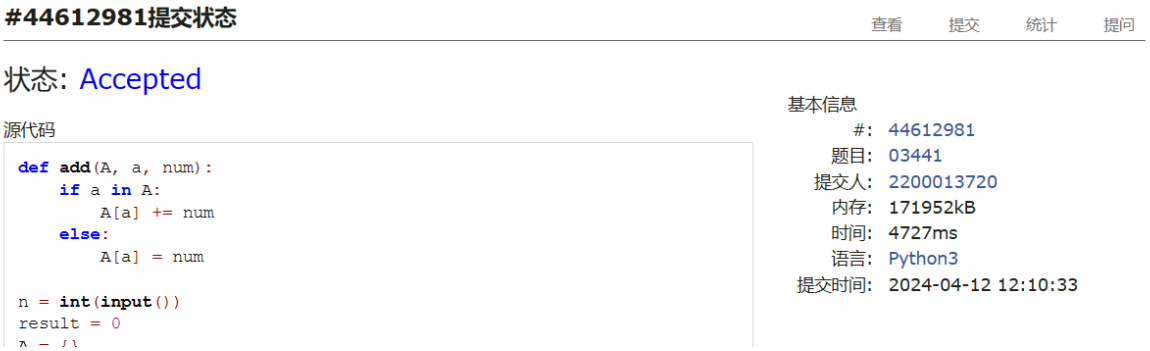
data structure/binary search, http://cs101.openjudge.cn/practice/03441

思路：这道题因为看到了之前学计概的时候一堆WA和TLE所以迟迟不敢下手，结果改用字典一下子就过了，所以大部分时候可能不是不会做，而是真的自己吓死自己

代码

```
#
def add(A, a, num):
    if a in A:
        A[a] += num
    else:
        A[a] = num

n = int(input())
result = 0
A = {}
B = {}
C = {}
D = {}
for _ in range(n):
    a,b,c,d = map(int,input().split())
    add(A,a,1)
    add(B,b,1)
    add(C,-c,1)
    add(D,-d,1)
sumAB = {}
for a in A:
    for b in B:
        add(sumAB, a+b, A[a]*B[b])
for c in C:
    for d in D:
        if c+d in sumAB:
            result += sumAB[c+d]*C[c]*D[d]
print(result)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



04089: 电话号码

trie, http://cs101.openjudge.cn/practice/04089/

Trie 数据结构可能需要自学下。

思路：当初是在每日选做上做的，用了个十分暴力的字典集合，以字典值代表集合元素的长度，集合元素即为每个号码对应长度的前缀，不过相比于Trie结构这种用的内存可能偏多

代码

```
#
t = int(input())
for _ in range(t):
    n = int(input())
    code = set()
    telephone = []
    for i in range(n):
        number = input()
        telephone.append([number, len(number)])
    telephone.sort(key = lambda x:x[1], reverse=True)
    maxl = telephone[0][1]
    minl = telephone[-1][1]
    length = {i:set() for i in range(minl, maxl+1)}
    check = False
    for number in telephone:
        number = number[0]
        l = len(number)
        if number in length[l]:
            check = True
            break
        else:
            for i in range(minl, l+1):
                length[i].add(number[:i])
    if check:
        print('NO')
    else:
        print('YES')
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

状态: Accepted

源代码

```
t = int(input())
for _ in range(t):
    n = int(input())
    code = set()
    telephone = []
    for i in range(n):
        number = input()
        telephone.append([number, len(number)])
    telephone.sort(key = lambda x:x[1], reverse=True)
    maxl = telephone[0][1]
```

基本信息

#:	44618438
题目:	04089
提交人:	2200013720
内存:	11420kB
时间:	168ms
语言:	Python3
提交时间:	2024-04-12 18:49:45

04082: 树的镜面映射

http://cs101.openjudge.cn/practice/04082/

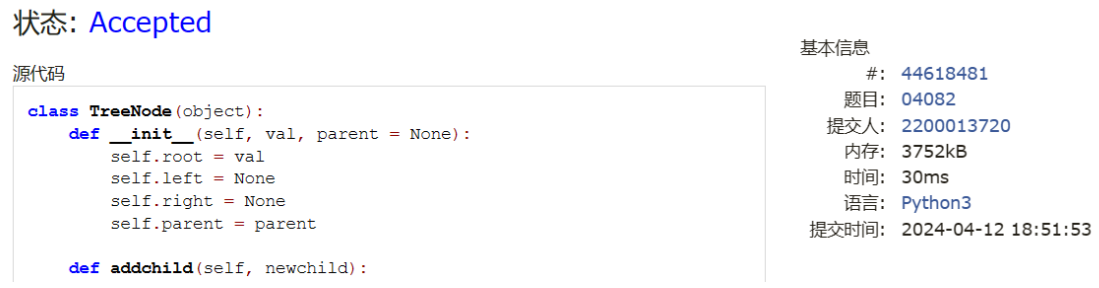
思路：在初次WA后一直固执地以为自己是遍历出了问题，但是总是想不通为什么，直到多日之后突然想起，在重建的时候给自己挖了个坑，于是更改了重建树的办法就AC了（被自己气死）。总的思路就是依据后缀为0/1重建伪满二叉树，然后依据伪满二叉树直接输出结果即可，因为依据伪满二叉树重建多叉树的过程本身就是层次遍历的过程。

代码

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

    def addchild(self, newchild):
        if self.left == None:
            self.left = newchild
        elif self.right == None:
            self.right = newchild

def build(temps):
    if temps == []:
        return None
    elif len(temps) == 1:
        return TreeNode(temps[0][0])
    else:
        root = TreeNode(temps[0][0])
        while len(temps) >= 2:
            temp = temps[1]
            newchild = TreeNode(temp[0], parent = root)
            if temp[1] == '1':
                root.addchild(newchild)
                while not root.right == None and not root.parent == None:
                    root = root.parent
            else:
                root.addchild(newchild)
                if not root.right == None:
                    root = root.right
                else:
                    root = root.left
            temps = temps[1:]
        return root

def getright(treenode):
    trees = []
    if not treenode.root[0] == '$':
        trees.append(treenode)
        if not treenode.right == None:
            addtr = getright(treenode.right)
            trees += addtr
    return trees

def loop(trees):
    result = []
    addtree = []
    if len(trees) == 0:
        return result
    else:
```

```
        for idx in range(len(trees)):
            temp = trees[idx]
            result.append(temp.root)
            if not temp.left == None:
                readd = getright(temp.left)
                addtree += readd
        result.reverse()
        result += loop(addtree)
        return result

N = int(input())
temps = input().split()
tree = build(temps)
result = loop([tree])
print(' '.join(result))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



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。==

感觉作业的难度还行，两道图的题目肉眼可见的简单，现在的精力堪堪跟上作业节奏和自己的安排，每日选做已经是“试着做做，做不出来就之后再做”的心态了，不过期中过后其他课程的作业压力就会小很多，到时候会努力赶上来的！已经坚持了半学期了，再多坚持一会！