

# Assignment #8: 图论：概念、遍历，及 树算

Updated 1919 GMT+8 Apr 8, 2024

2024 spring, Compiled 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

### 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：

正常的操作问题

代码

```
1  '''
2  2100017810  刘思瑞
3  '''
4  n, m = map(int, input().split())
5
6  A = [[0]*n for _ in range(n)]
7  D = [0]*n
8
9  for _ in range(m):
```

```

10     a, b = map(int, input().split())
11     A[a][b] = 1
12     A[b][a] = 1
13     D[a] += 1
14     D[b] += 1
15
16 L = [[D[i] if i == j else -A[i][j] for j in range(n)] for i in range(n)]
17
18 for row in L:
19     print(*row)

```

代码运行截图

状态: Accepted

源代码

```

'''
2100017810 刘思瑞
'''
n, m = map(int, input().split())

A = [[0]*n for _ in range(n)]
D = [0]*n

for _ in range(m):
    a, b = map(int, input().split())
    A[a][b] = 1
    A[b][a] = 1
    D[a] += 1
    D[b] += 1

L = [[D[i] if i == j else -A[i][j] for j in range(n)] for i in range(n)]

for row in L:
    print(*row)

```

## 18160: 最大连通域面积

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

思路:

dfs

代码

```

1  '''
2  刘思瑞 2100017810

```

```

3  '''
4  m,flag,N,M,summ = [],[],0,0,0
5  def search(i,j):
6      global m,flag,N,M,summ
7      if i != 0:
8          if ((flag[i-1][j] == True) and (m[i-1][j] == 'w')):
9              summ += 1
10             flag[i-1][j] = False
11             search(i-1,j)
12             if ((flag[i-1][j+1] == True) and (m[i-1][j+1] == 'w')):
13                 summ += 1
14                 flag[i-1][j+1] = False
15                 search(i-1,j+1)
16             if j != 0:
17                 if ((flag[i-1][j-1] == True) and (m[i-1][j-1] == 'w')):
18                     summ += 1
19                     flag[i-1][j-1] = False
20                     search(i-1,j-1)
21             if ((flag[i][j+1] == True) and (m[i][j+1] == 'w')):
22                 summ += 1
23                 flag[i][j+1] = False
24                 search(i,j+1)
25             if ((flag[i+1][j+1] == True) and (m[i+1][j+1] == 'w')):
26                 summ += 1
27                 flag[i+1][j+1] = False
28                 search(i+1,j+1)
29             if ((flag[i+1][j] == True) and (m[i+1][j] == 'w')):
30                 summ += 1
31                 flag[i+1][j] = False
32                 search(i+1,j)
33             if j != 0:
34                 if ((flag[i][j-1] == True) and (m[i][j-1] == 'w')):
35                     summ += 1
36                     flag[i][j-1] = False
37                     search(i,j-1)
38                 if ((flag[i+1][j-1] == True) and (m[i+1][j-1] == 'w')):
39                     summ += 1
40                     flag[i+1][j-1] = False
41                     search(i+1,j-1)
42             return
43
44
45  num = int(input())
46  for k in range(num):
47      m = []
48      flag = []
49      sum = 0
50      N,M = map(int,input().split())
51      for i in range(N):
52          flag.append([True]*(M)+[False])
53          s = input()
54          temp = []
55          for j in range(M):
56              temp.append(s[j])
57          temp.append('.')
58          m.append(temp)

```

```

59     m.append(['.']*(M+1))
60     flag.append([False]*(M+1))
61     for i in range(N):
62         for j in range(M):
63             if m[i][j] == 'W' and flag[i][j] == True:
64                 summ = 1
65                 flag[i][j] = False
66                 search(i,j)
67                 sum = max(sum, summ)
68     print(sum)

```

代码运行截图

## #42964992提交状态

状态: Accepted

源代码

```

'''
刘思瑞 2100017810
'''
m, flag, N, M, summ = [], [], 0, 0, 0
def search(i, j):
    global m, flag, N, M, summ
    if i != 0:
        if ((flag[i-1][j] == True) and (m[i-1][j] == 'W')):
            summ += 1
            flag[i-1][j] = False
            search(i-1, j)
        if ((flag[i-1][j+1] == True) and (m[i-1][j+1] == 'W')):
            summ += 1
            flag[i-1][j+1] = False
            search(i-1, j+1)
    if j != 0:
        if ((flag[i-1][j-1] == True) and (m[i-1][j-1] == 'W')):
            summ += 1
            flag[i-1][j-1] = False
            search(i-1, j-1)
    if ((flag[i][j+1] == True) and (m[i][j+1] == 'W')):
        summ += 1
        flag[i][j+1] = False
        search(i, j+1)

```

## sy383: 最大权值连通块

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

思路:

dfs

代码

```
1  '''
2  2100017810 刘思瑞
3  '''
4  def dfs(u):
5      vis[u] = True
6      weight_sum = weight[u]
7      for v in G[u]:
8          if not vis[v]:
9              weight_sum += dfs(v)
10     return weight_sum
11
12     n, m = map(int, input().split())
13     weight = list(map(int, input().split()))
14     G = [[] for _ in range(n)]
15     vis = [False] * n
16
17     for _ in range(m):
18         u, v = map(int, input().split())
19         G[u].append(v)
20         G[v].append(u)
21
22     max_weight_sum = 0
23     for i in range(n):
24         if not vis[i]:
25             max_weight_sum = max(max_weight_sum, dfs(i))
26
27     print(max_weight_sum)
```

代码运行截图

```

1  '''
2  2100017810 刘思瑞
3  '''
4  def dfs(u):
5      vis[u] = True
6      weight_sum = weight[u]
7      for v in G[u]:
8          if not vis[v]:
9              weight_sum += dfs(v)
10     return weight_sum
11
12     n, m = map(int, input().split())
13     weight = list(map(int, input().split()))

```

测试输入

提交结果

历史提交

完美通过

[查看题解](#)

100% 数据通过测试

运行时长: 0 ms

## 03441: 4 Values whose Sum is 0

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

思路:

搜索

代码

```

1  n = int(input())
2  A = []
3  B = []
4  C = []
5  D = []
6
7  for _ in range(n):
8      a, b, c, d = map(int, input().split())
9      A.append(a)
10     B.append(b)
11     C.append(c)
12     D.append(d)
13
14     sums = {}
15
16     for i in range(n):

```

```

17     for j in range(n):
18         ab_sum = A[i] + B[j]
19         if ab_sum in sums:
20             sums[ab_sum] += 1
21         else:
22             sums[ab_sum] = 1
23
24 count = 0
25
26 for i in range(n):
27     for j in range(n):
28         cd_sum = -(C[i] + D[j])
29         if cd_sum in sums:
30             count += sums[cd_sum]
31
32 print(count)

```

代码运行截图

---

状态: **Accepted**

源代码

```

n = int(input())
A = []
B = []
C = []
D = []

for _ in range(n):
    a, b, c, d = map(int, input().split())
    A.append(a)
    B.append(b)
    C.append(c)
    D.append(d)

sums = {}

for i in range(n):
    for j in range(n):
        ab_sum = A[i] + B[j]
        if ab_sum in sums:
            sums[ab_sum] += 1
        else:
            sums[ab_sum] = 1

count = 0

for i in range(n):
    for j in range(n):
        cd_sum = -(C[i] + D[j])
        if cd_sum in sums:

```

## 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:

tire

代码

```
1  '''
2  2100017810 刘思瑞
3  '''
4  class TrieNode:
5      def __init__(self):
6          self.children = {}
7          self.is_end = False
8
9  def insert(root, number):
10     node = root
11     for digit in number:
12         if digit not in node.children:
13             node.children[digit] = TrieNode()
14         node = node.children[digit]
15         if node.is_end:
16             return False
17     node.is_end = True
18     if node.children:
19         return False
20     return True
21
22 def is_consistent(numbers):
23     root = TrieNode()
24     for number in numbers:
25         if not insert(root, number):
26             return False
27     return True
28
29 t = int(input())
30
31 for _ in range(t):
32     n = int(input())
33     numbers = [input().strip() for _ in range(n)]
34     if is_consistent(numbers):
35         print("YES")
36     else:
37         print("NO")
```

代码运行截图



状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''
class TrieNode:
    def __init__(self):
        self.children = {}
        self.is_end = False

    def insert(root, number):
        node = root
        for digit in number:
            if digit not in node.children:
                node.children[digit] = TrieNode()
            node = node.children[digit]
            if node.is_end:
                return False
        node.is_end = True
        if node.children:
            return False
        return True

    def is_consistent(numbers):
        root = TrieNode()
        for number in numbers:
            if not insert(root, number):
                return False
        return True

t = int(input())
```

## 04082: 树的镜面映射

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

思路:

代码

```
1 | #
2 |
```

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

---

这周期中周没来得及做太多