Assignment #8: 图论：概念、遍历，及 树算

Updated 1919 GMT+8 Apr 8, 2024

2024 spring, Complied 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 typorahttps://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

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

```
n,m=map(int,input().split())
l=[[0]*n for _ in range(n)]
for _ in range(m):
    a,b=map(int,input().split())
    l[a][a]+=1
    l[b][b]+=1
    l[a][b]-=1
    l[b][a]-=1
for row in l: print(*row)
```

代码运行截图

18160: 最大连通域面积

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

思路:

代码

```
def dfs(grid, visited, i, j):
    if i < 0 or i >= len(grid) or j < 0 or j >= len(grid[0]) or grid[i][j] != 'W' or
visited[i][j]:
        return 0

    visited[i][j] = True
    area = 1

    directions = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1, 0), (1, 1)]
    for dx, dy in directions:
        area += dfs(grid, visited, i + dx, j + dy)

    return area



def max_connected_area(grid):
    n = len(grid)
    m = len(grid[0])
    visited = [[False] * m for _ in range(n)]
    max_area = 0

    for i in range(n):
        for j in range(m):
            if grid[i][j] == 'W' and not visited[i][j]:
                max_area = max(max_area, dfs(grid, visited, i, j))

    return max_area



T = int(input())
for _ in range(T):
    N, M = map(int, input().split())
    grid = [input() for _ in range(N)]
    print(max_connected_area(grid))
```

代码运行截图

sy383: 最大权值连通块

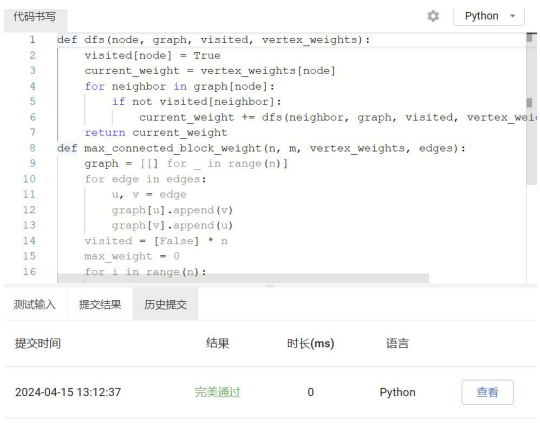
https://sunnywhy.com/sfbj/10/3/383

思路:

代码

```
def dfs(node, graph, visited, vertex_weights):
    visited[node] = True
    current_weight = vertex_weights[node]
    for neighbor in graph[node]:
        if not visited[neighbor]:
            current_weight += dfs(neighbor, graph, visited, vertex_weights)
    return current_weight
def max_connected_block_weight(n, m, vertex_weights, edges):
    graph = [[] for _ in range(n)]
    for edge in edges:
        u, v = edge
        graph[u].append(v)
        graph[v].append(u)
    visited = [False] * n
    max_weight = 0
    for i in range(n):
        if not visited[i]:
            max_weight = max(max_weight, dfs(i, graph, visited, vertex_weights))
    return max_weight
n, m = map(int, input().split())
vertex_weights = list(map(int, input().split()))
edges = []
for _ in range(m):
    u, v = map(int, input().split())
    edges.append((u, v))
print(max_connected_block_weight(n, m, vertex_weights, edges))
```

代码运行截图



03441: 4 Values whose Sum is 0

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

思路：

代码

```
#

```

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：

代码

```
class TrieNode:
    def __init__(self):
        self.children = {}
        self.is_end = False
class Trie:
    def __init__(self):
        self.root = TrieNode()
    def insert(self, word):
        node = self.root
        for char in word:
            if char not in node.children:
                node.children[char] = TrieNode()
            node = node.children[char]
            if node.is_end:
                return False
        node.is_end = True
        return len(node.children) == 0
def is_consistent(numbers):
    trie = Trie()
    for number in numbers:
        if not trie.insert(number):
            return "NO"
    return "YES"
t = int(input())
for _ in range(t):
    n = int(input())
    numbers = [input() for _ in range(n)]
    print(is_consistent(numbers))
```

代码运行截图 

04082: 树的镜面映射

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

思路:

代码

。

```
#

```

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

本周由于进入了新的内容，没什么做题经验，花了挺多时间思考题目但是没什么结果，后面还是照着gpt的答案来学习本周的题目。接下来我将花更多时间来学习相关内容。