Assignment #8: 图论：概念、遍历，及 树算

Updated 1919 GMT+8 Apr 8, 2024

2024 spring, Complied by 李鹏辉，元培学院

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

Windows 10 Home, PyCharm 2022.3.2 (Community Edition)

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：约40min.

代码

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

    def add_neighbor(self, nbr, weight=0):
        self.connected_to[nbr] = weight


```

```python
class Graph:
    def __init__(self):
        self.vert_list = {}
        self.num_vertices = 0

    def add_vertex(self, key):
        self.vert_list[key] = Vertex(key)
        self.num_vertices += 1

    def __contains__(self, n):
        return n in self.vert_list

    def add_edge(self, f, t, weight=0):
        if f not in self.vert_list:
            self.add_vertex(f)
        if t not in self.vert_list:
            self.add_vertex(t)
        self.vert_list[f].add_neighbor(self.vert_list[t], weight)

    def __iter__(self):
        return iter(self.vert_list.values())


def q1():

    def llm(n, edges):
        graph = Graph()
        for i in range(n):
            graph.add_vertex(i)
        for edge in edges:
            a, b = edge
            graph.add_edge(a, b)
            graph.add_edge(b, a)
        matrix = []
        for vertex in graph:
            row = [0] * n
            row[vertex.id] = len(vertex.connected_to)
            for neighbor in vertex.connected_to.keys():
                row[neighbor.id] = -1
            matrix.append(row)
        return matrix

    m, n = map(int, input().split())
    edges = []
    for i in range(n):
        a, b = map(int, input().split())
        edges.append((a, b))

    re_matrix = llm(m, edges)
    for row in re_matrix:
        print(' '.join(map(str, row)))


q1()
```

代码运行截图 （至少包含有"Accepted"）

状态: Accepted

源代码

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

    def add_neighbor(self, nbr, weight=0):
```

基本信息

#: 44668178
题目: 19943
提交人: 2100017777
内存: 3716kB
时间: 29ms
语言: Python3
提交时间: 2024-04-15 21:27:36

18160: 最大连通域面积

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

思路：约30min.

代码

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

    def add_neighbor(self, nbr, weight=0):
        self.connected_to[nbr] = weight


class Graph:
    def __init__(self):
        self.vert_list = {}
        self.num_vertices = 0

    def add_vertex(self, key):
        self.vert_list[key] = Vertex(key)
        self.num_vertices += 1

    def __contains__(self, n):
        return n in self.vert_list

    def add_edge(self, f, t, weight=0):
        if f not in self.vert_list:
            self.add_vertex(f)
        if t not in self.vert_list:
            self.add_vertex(t)
        self.vert_list[f].add_neighbor(self.vert_list[t], weight)

    def __iter__(self):
        return iter(self.vert_list.values())


```

```python
def q2():

    def dfs(graph, vertex, visited):
        visited[vertex.id] = True
        area = 1
        for neighbor in vertex.connected_to:
            if not visited[neighbor.id]:
                area += dfs(graph, neighbor, visited)
        return area

    t = int(input())
    for _ in range(t):
        m, n = map(int, input().split())
        graph = Graph()
        grid = []
        # add vertices
        for i in range(m):
            row = input().strip()
            grid.append(row)
            for j in range(n):
                if row[j] == 'W':
                    graph.add_vertex((i, j))

        # add edges
        for i in range(m):
            for j in range(n):
                if (i, j) in graph.vert_list:
                    for dx in [-1, 0, 1]:
                        for dy in [-1, 0, 1]:
                            x, y = i + dx, j + dy
                            if (dx != 0 or dy != 0) and 0 <= x < m and 0 <= y < n and (x, y) in graph.vert_list:
                                graph.add_edge((i, j), (x, y))

        max_area = 0
        visited = {key: False for key in graph.vert_list.keys()}
        for i in range(m):
            for j in range(n):
                if (i, j) in graph.vert_list and not visited[(i, j)]:
                    vertex = graph.vert_list[(i, j)]
                    area = dfs(graph, vertex, visited)
                    max_area = max(max_area, area)
        print(max_area)


q2()
```

代码运行截图 （至少包含有"Accepted"）

状态: Accepted

源代码

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

    def add_neighbor(self, nbr, weight=0):
```

基本信息

#:	44668611
题目:	18160
提交人:	2100017777
内存:	4704kB
时间:	197ms
语言:	Python3
提交时间:	2024-04-15 21:57:03

sy383: 最大权值连通块

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

思路：约20min. 基于第二题改动。

代码

```python
class Vertex:
    def __init__(self, key, weight=0):
        self.id = key
        self.connected_to = {}
        self.weight = weight

    def add_neighbor(self, nbr, weight=0):
        self.connected_to[nbr] = weight


class Graph:
    def __init__(self):
        self.vert_list = {}
        self.num_vertices = 0

    def add_vertex(self, key, weight=0):
        self.vert_list[key] = Vertex(key, weight)
        self.num_vertices += 1

    def __contains__(self, n):
        return n in self.vert_list

    def add_edge(self, f, t, weight=0):
        if f not in self.vert_list:
            self.add_vertex(f)
        if t not in self.vert_list:
            self.add_vertex(t)
        self.vert_list[f].add_neighbor(self.vert_list[t], weight)

    def __iter__(self):
        return iter(self.vert_list.values())


def q3():

```

```
    def dfs(graph, vertex, visited):
        visited[vertex.id] = True
        weight = vertex.weight
        for neighbor in vertex.connected_to:
            if not visited[neighbor.id]:
                weight += dfs(graph, neighbor, visited)
        return weight

    m, n = map(int, input().split())
    weights = list(map(int, input().split()))
    graph = Graph()
    for i, weight in enumerate(weights):
        graph.add_vertex(i, weight)
    for _ in range(n):
        a, b = map(int, input().split())
        graph.add_edge(a, b)
        graph.add_edge(b, a)

    max_weight = 0
    visited = {key: False for key in graph.vert_list.keys()}
    for vertex in graph:
        if not visited[vertex.id]:
            max_weight = max(max_weight, dfs(graph, vertex, visited))

    print(max_weight)


q3()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

代码书写

Python

```python
class Vertex:
    def __init__(self, key, weight=0):
        self.id = key
        self.connected_to = {}
        self.weight = weight

    def add_neighbor(self, nbr, weight=0):
        self.connected_to[nbr] = weight

```

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：约20min.

代码

```python
def q4():
    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)

    count = 0
    AB_sums = {}
    for a in A:
        for b in B:
            AB_sum = a + b
            if AB_sum in AB_sums:
                AB_sums[AB_sum] += 1
            else:
```

```
                AB_sums[AB_sum] = 1

    for c in C:
        for d in D:
            CD_sum = c + d
            if -CD_sum in AB_sums:
                count += AB_sums[-CD_sum]

    print(count)


q4()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

状态: Accepted

源代码

```
def q4():
    n = int(input())
    A, B, C, D = [], [], [], []
    for _ in range(n):
        a, b, c, d = map(int, input().split())
        A.append(a)
```

基本信息

#:	44669580
题目:	03441
提交人:	2100017777
内存:	171608kB
时间:	3247ms
语言:	Python3
提交时间:	2024-04-15 23:30:44

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：约50分钟。

代码

```
class Trie:
    def __init__(self):
        self.children = {}
        self.is_end = False


def q5():

    def insert(t, word):
        nonlocal result
        if t.is_end:
            result = False
            return
        if not word:
            if t.children:
                result = False
                return
```

```
            t.is_end = True
        else:
            letter = word[0]
            if letter not in t.children:
                t.children[letter] = Trie()
            insert(t.children[letter], word[1:])

    for _ in range(int(input())):
        t = Trie()
        result = True
        for i in range(int(input())):
            word = input()
            if not result:
                continue
            insert(t, word)
        if result:
            print('YES')
        else:
            print('NO')


q5()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

状态: Accepted

源代码

```
class Trie:
    def __init__(self):
        self.children = {}
        self.is_end = False
```

基本信息

#:	44670025
题目:	04089
提交人:	2100017777
内存:	24184kB
时间:	367ms
语言:	Python3
提交时间:	2024-04-16 01:20:51

04082: 树的镜面映射

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

思路：约40min.

代码

```
class BTree:
    def __init__(self, node, left=None, right=None):
        self.node = node
        self.left = left
        self.right = right
        self.is_leaf = True


def q6():
```

```
    input()
    raw_nodes = input().split()

    def build():
        nonlocal raw_nodes
        root = raw_nodes[0]
        t = BTree(root[0])
        # wait
        raw_nodes = raw_nodes[1:]
        if root[1] == '0':
            t.is_leaf = False
            t.left = build()
            t.right = build()
        return t

    root_tree = build()
    current_level = [root_tree]
    next_level = []
    cl_result = []
    final_result = []
    while current_level:
        for t in current_level:
            c_n = t
            while True:
                if c_n.node != '$':
                    cl_result.append(c_n.node)
                if c_n.right:
                    next_level.append(c_n.left)
                    c_n = c_n.right
                else: break

        final_result.extend(cl_result[::-1])
        current_level = next_level[:]
        next_level = []
        cl_result = []
    print(' '.join(final_result))


q6()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

状态: Accepted

源代码

```
class BTree:
    def __init__(self, node, left=None, right=None):
        self.node = node
        self.left = left
        self.right = right
        self.is_leaf = True
```

基本信息

#:	44669887
题目:	04082
提交人:	2100017777
内存:	3680kB
时间:	26ms
语言:	Python3
提交时间:	2024-04-16 00:28:30

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。

第五题的输出需要注意，按照题意，尽管有些NO的情况可以提早判断，依然要等到一组数据全部输入完后再输出NO or YES.