Assignment #B: 图论和树算

Updated 1709 GMT+8 Apr 28, 2024

2024 spring, Complied by 陈奕好 工学院

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 Sonoma 14.4 (23E214)

Python编程环境：PyCharm 2023.3.1 (Professional Edition)

1. 题目

28170: 算鹰

dfs, http://cs101.openjudge.cn/practice/28170/

思路：dfs的模版题

代码

```
board = [list(map(str, input())) for _ in range(10)]
visited = [[0] * 10 for _ in range(10)]
dx = [0, 0, 1, -1]
dy = [1, -1, 0, 0]
cnt = 0
# print(*broad, sep='\n')
# ['-', '-', '-', '.', '-', '-', '.', '-', '.', '.']
# ['-', '.', '.', '-', '.', '-', '.', '.', '.', '.']
# ['.', '.', '.', '-', '-', '.', '.', '.', '.', '-']
```

```python
# ['-', '-', '-', '-', '.', '.', '.', '.', '.', '.']
# ['-', '-', '.', '-', '-', '-', '.', '.', '.', '.']
# ['-', '.', '-', '.', '.', '-', '.', '-', '-', '-']
# ['.', '.', '.', '.', '-', '.', '-', '.', '.', '-']
# ['-', '.', '.', '-', '-', '-', '-', '-', '.', '.']
# ['-', '.', '.', '.', '.', '.', '.', '.', '-', '.']
# ['.', '.', '.', '.', '.', '-', '-', '.', '-', '-']


def dfs(chessboard, x, y, visited):
    for i in range(4):
        new_x = x + dx[i]
        new_y = y + dy[i]
        if 0 <= new_x < 10 and 0 <= new_y < 10 and chessboard[new_x][new_y] == '.'
and visited[new_x][new_y] == 0:
            visited[new_x][new_y] = 1
            dfs(chessboard, new_x, new_y, visited)


for i in range(10):
    for j in range(10):
        if board[i][j] == '.' and visited[i][j] == 0:
            visited[i][j] = 1
            dfs(board, i, j, visited)
            cnt += 1

print(cnt)


```

代码运行截图 （至少包含有"Accepted"）

状态: Accepted

基本信息

#: 44878936
题目: 28170
提交人: 23n2300011030(陈奕好)
内存: 3656kB
时间: 19ms
语言: Python3
提交时间: 2024-05-06 14:21:27

源代码

```
board = [list(map(str, input())) for _ in range(10)]
visited = [[0] * 10 for _ in range(10)]
dx = [0, 0, 1, -1]
dy = [1, -1, 0, 0]
cnt = 0
# print(*broad, sep='\n')
# ['-', '-', '-', '.', '-', '-', '.', '-', '.', '.']
# ['-', '.', '.', '-', '.', '-', '.', '.', '.', '.']
# ['.', '.', '.', '-', '-', '.', '.', '.', '.', '-']
# ['-', '-', '-', '-', '.', '.', '.', '.', '.', '.']
# ['-', '-', '.', '-', '-', '-', '.', '.', '.', '.']
# ['-', '.', '-', '.', '.', '-', '.', '-', '-', '-']
# ['.', '.', '.', '.', '-', '.', '-', '.', '.', '-']
# ['-', '.', '.', '-', '-', '-', '-', '-', '.', '.']
# ['-', '.', '.', '.', '.', '.', '.', '.', '-', '.']
# ['.', '.', '.', '.', '.', '-', '-', '.', '-', '-']


def dfs(chessboard, x, y, visited):
    for i in range(4):
        new_x = x + dx[i]
        new_y = y + dy[i]
        if 0 <= new_x < 10 and 0 <= new_y < 10 and chessboard[new_x][new
            visited[new_x][new_y] = 1
            dfs(chessboard, new_x, new_y, visited)


for i in range(10):
    for j in range(10):
        if board[i][j] == '.' and visited[i][j] == 0:
            visited[i][j] = 1
            dfs(board, i, j, visited)
```

02754: 八皇后

dfs, http://cs101.openjudge.cn/practice/02754/

思路：换用bfs，从835ms下降至35ms

代码

```
from collections import deque
n = int(input())
ans = []


def isVaild(string, x):
    for i in string:
        if abs(int(i)-int(x)) == abs(string.index(i)-len(string)):
            return False
    return True


def bfs(size):
```

```
    queue = deque()
    for i in range(1, size+1):
        queue.append(str(i))

    while queue:
        string = queue.popleft()
        if len(string) == size:
            ans.append(string)
        else:
            for i in range(1, size+1):
                if str(i) not in string and isVaild(string, str(i)):
                    queue.append(string+str(i))


bfs(8)
for i in range(n):
    print(ans[int(input()) - 1])

```

代码运行截图 （至少包含有"Accepted"）

状态: Accepted

源代码

```
from collections import deque
n = int(input())
ans = []


def isVaild(string, x):
    for i in string:
        if abs(int(i)-int(x)) == abs(string.index(i)-len(string)):
            return False
    return True


def bfs(size):
    queue = deque()
    for i in range(1, size+1):
        queue.append(str(i))

    while queue:
        string = queue.popleft()
        if len(string) == size:
            ans.append(string)
        else:
            for i in range(1, size+1):
                if str(i) not in string and isVaild(string, str(i)):
                    queue.append(string+str(i))


bfs(8)
for i in range(n):
    print(ans[int(input()) - 1])
```

基本信息

#:	44879044
题目:	02754
提交人:	23n2300011030(陈奕好)
内存:	3632kB
时间:	35ms
语言:	Python3
提交时间:	2024-05-06 14:41:09

03151: Pots

bfs, http://cs101.openjudge.cn/practice/03151/

思路：

代码

```python
from collections import deque

def bfs(A, B, C):
    queue = deque([((0, 0), [])])
    visited = set([(0, 0)])
    while queue:
        (a, b), path = queue.popleft()
        if a == C or b == C:
            return path
        states = [((A, b), path + ['FILL(1)']),
                  ((a, B), path + ['FILL(2)']),
                  ((0, b), path + ['DROP(1)']),
                  ((a, 0), path + ['DROP(2)']),
                  ((a-min(a, B-b), b+min(a, B-b)), path + ['POUR(1,2)']),
                  ((a+min(b, A-a), b-min(b, A-a)), path + ['POUR(2,1)'])]
        for state, new_path in states:
            if state not in visited:
                queue.append((state, new_path))
                visited.add(state)
    return None

def solve(A, B, C):
    if A < C and B < C:
        return "impossible"
    path = bfs(A, B, C)
    if path is None:
        return "impossible"
    return f"{len(path)}\n" + "\n".join(path)

A, B, C = map(int, input().split())
print(solve(A, B, C))


```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44883430提交状态 查看 提交 统计 提问

状态: Accepted

基本信息

#: 44883430
题目: 03151
提交人: 23n2300011030(陈奕好)
内存: 3676kB
时间: 22ms
语言: Python3
提交时间: 2024-05-06 21:44:12

源代码

```
from collections import deque

def bfs(A, B, C):
    queue = deque([((0, 0), [])])
    visited = {(0, 0)}
    while queue:
        (a, b), path = queue.popleft()
        if a == C or b == C:
            return path
        states = [((A, b), path + ['FILL(1)']),
                  ((a, B), path + ['FILL(2)']),
                  ((0, b), path + ['DROP(1)']),
                  ((a, 0), path + ['DROP(2)']),
                  ((a-min(a, B-b), b+min(a, B-b)), path + ['POUR(1,2)']),
                  ((a+min(b, A-a), b-min(b, A-a)), path + ['POUR(2,1)'])]
        for state, new_path in states:
            if state not in visited:
                queue.append((state, new_path))
                visited.add(state)
    return None

def solve(A, B, C):
    if A < C and B < C:
        return "impossible"
    path = bfs(A, B, C)
    if path is None:
        return "impossible"
    return f"{len(path)}\n" + "\n".join(path)
```

05907: 二叉树的操作

http://cs101.openjudge.cn/practice/05907/

思路：用字典模拟树

代码

```
for turn in range(int(input())):
    n, m = map(int, input().split())
    nodes = {i: (-1, -1) for i in range(n)}

    for node in range(n):
        X, Y, Z = map(int, input().split())
        nodes[X] = (Y, Z)

    for operation in range(m):
        op = list(map(int, input().split()))
        if op[0] == 1:
            x, y = op[1], op[2]
            for node in nodes:
                if nodes[node][0] == x:
                    nodes[node] = (y, nodes[node][1])
```

```
                elif nodes[node][0] == y:
                    nodes[node] = (x, nodes[node][1])

                if nodes[node][1] == x:
                    nodes[node] = (nodes[node][0], y)
                elif nodes[node][1] == y:
                    nodes[node] = (nodes[node][0], x)

        elif op[0] == 2:
            x = op[1]
            while nodes[x][0] != -1:
                x = nodes[x][0]
            print(x)

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44883630提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
for turn in range(int(input())):
    n, m = map(int, input().split())
    nodes = {i: (-1, -1) for i in range(n)}

    for node in range(n):
        X, Y, Z = map(int, input().split())
        nodes[X] = (Y, Z)

    for operation in range(m):
        op = list(map(int, input().split()))
        if op[0] == 1:
            x, y = op[1], op[2]
            for node in nodes:
                if nodes[node][0] == x:
                    nodes[node] = (y, nodes[node][1])
                elif nodes[node][0] == y:
                    nodes[node] = (x, nodes[node][1])

                if nodes[node][1] == x:
                    nodes[node] = (nodes[node][0], y)
                elif nodes[node][1] == y:
                    nodes[node] = (nodes[node][0], x)

        elif op[0] == 2:
            x = op[1]
            while nodes[x][0] != -1:
                x = nodes[x][0]
```

基本信息

#: 44883630
题目: 05907
提交人: 23n2300011030(陈奕好)
内存: 3728kB
时间: 288ms
语言: Python3
提交时间: 2024-05-06 22:05:26

18250: 冰阔落 I

Disjoint set, http://cs101.openjudge.cn/practice/18250/

思路：优化了union，使其成为了判断条件

代码

```
class DisjSet:
    def __init__(self, n):
        self.rank = [1] * (n+1)
        self.parent = [(i) for i in range(n+1)]

    def find(self, x):
        if self.parent[x] != x:
            self.parent[x] = self.find(self.parent[x])
        return self.parent[x]

    def union(self, x, y):
        x_root, y_root = self.find(x), self.find(y)
        if x_root != y_root:
            self.parent[y_root] = x_root
            self.rank[x_root] += 1
            return False
        return True


while True:
    try:
        n, m = map(int, input().split())
        ds = DisjSet(n)
        for i in range(m):
            x, y = map(int, input().split())
            if ds.union(x, y):
                print("Yes")
            else:
                print("No")

        ds.parent[1:] = [ds.find(i) for i in ds.parent[1:]]
        print(len(set(ds.parent[1:])))
        print(" ".join(str(i) for i in sorted(list(set(ds.parent[1:])))))

    except EOFError:
        break

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44679615提交状态

查看 提交 统计 提问

状态: Accepted

基本信息

#: 44679615
题目: 18250
提交人: 23n2300011030(陈奕好)
内存: 6804kB
时间: 386ms
语言: Python3
提交时间: 2024-04-16 23:06:35

源代码

```
class DisjSet:
    def __init__(self, n):
        self.rank = [1] * (n+1)
        self.parent = [(i) for i in range(n+1)]

    def find(self, x):
        if self.parent[x] != x:
            self.parent[x] = self.find(self.parent[x])
        return self.parent[x]

    def union(self, x, y):
        x_root, y_root = self.find(x), self.find(y)
        if x_root != y_root:
            self.parent[y_root] = x_root
            self.rank[x_root] += 1
            return False
        return True


while True:
    try:
        n, m = map(int, input().split())
        ds = DisjSet(n)
        for i in range(m):
            x, y = map(int, input().split())
            if ds.union(x, y):
                print("Yes")
```

05443: 兔子与樱花

http://cs101.openjudge.cn/practice/05443/

思路：dijkstra

代码

```python
import heapq

def dijkstra(graph, start):
    distances = {node: (float('infinity'), []) for node in graph}
    distances[start] = (0, [start])
    queue = [(0, start, [start])]
    while queue:
        current_distance, current_node, path = heapq.heappop(queue)
        if current_distance > distances[current_node][0]:
            continue
        for neighbor, weight in graph[current_node].items():
            distance = current_distance + weight
            if distance < distances[neighbor][0]:
                distances[neighbor] = (distance, path + [neighbor])
                heapq.heappush(queue, (distance, neighbor, path + [neighbor]))
    return distances
```

```python

P = int(input())
places = {input(): i for i in range(P)}
graph = {i: {} for i in range(P)}

Q = int(input())
for _ in range(Q):
    place1, place2, distance = input().split()
    distance = int(distance)
    graph[places[place1]][places[place2]] = distance
    graph[places[place2]][places[place1]] = distance

R = int(input())
for _ in range(R):
    start, end = input().split()
    distances = dijkstra(graph, places[start])
    path = distances[places[end]][1]
    result = ""
    for i in range(len(path) - 1):
        result += f"{list(places.keys())[list(places.values()).index(path[i])]}->
({graph[path[i]][path[i+1]]})->"
    result += list(places.keys())[list(places.values()).index(path[-1])]
    print(result)

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44884256提交状态 查看 提交 统计 提问

状态: Accepted

源代码

```python
import heapq

def dijkstra(graph, start):
    distances = {node: (float('infinity'), []) for node in graph}
    distances[start] = (0, [start])
    queue = [(0, start, [start])]
    while queue:
        current_distance, current_node, path = heapq.heappop(queue)
        if current_distance > distances[current_node][0]:
            continue
        for neighbor, weight in graph[current_node].items():
            distance = current_distance + weight
            if distance < distances[neighbor][0]:
                distances[neighbor] = (distance, path + [neighbor])
                heapq.heappush(queue, (distance, neighbor, path + [neigh
    return distances


P = int(input())
places = {input(): i for i in range(P)}
graph = {i: {} for i in range(P)}

Q = int(input())
for _ in range(Q):
    place1, place2, distance = input().split()
    distance = int(distance)
    graph[places[place1]][places[place2]] = distance
```

基本信息

#:	44884256
题目:	05443
提交人:	23n2300011030(陈奕好)
内存:	3700kB
时间:	20ms
语言:	Python3
提交时间:	2024-05-06 23:21:26

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。

五一鸽了😭，不知道该怎么活过期末。

太焦虑了，但题目又很好玩，图的知识学的又不是太懂，做的时候很快乐，但真的不知道怎么挤出时间了。。。