Assignment #B: 图论和树算

Updated 1709 GMT+8 Apr 28, 2024

2024 spring, Complied by 数学科学学院 王镜廷 2300010724

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

操作系统：Windows11 专业版

Python编程环境：VSCode 1.86.2, with extension Python and python version 3.12.2

1. 题目

28170: 算鹰

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

用时：约20分钟

思路：依题意求出连通块即可

代码

```
class Graph:
    def __init__(self) :
        self.edges = {}
    def addv(self, u) :
        self.edges[u] = []
    def adde(self, u, v) :
        self.edges[u].append(v)
    def get_connected_component(self) :
        res = 0
        vis = {}
        for u in self.edges :
            vis[u] = 0
        for u in self.edges :
            if vis[u] == 1 :
                continue
            res += 1
            q = [u]
            while len(q) != 0 :
                v = q[-1]
                q.pop()
                if vis[v] == 1 :
                    continue
                vis[v] = 1
                for w in self.edges[v] :
                    if vis[w] == 0 :
                        q.append(w)
        return res

q = []
for i in range(10) :
    q.append(input())
G = Graph()
for i in range(10) :
    for j in range(10) :
        if q[i][j] == "." :
            G.addv((i, j))
for i in range(10) :
    for j in range(10) :
        for dx in [(1, 0), (-1, 0), (0, 1), (0, -1)] :
            if (i, j) in G.edges and (i + dx[0], j + dx[1]) in G.edges :
                G.adde((i, j), (i + dx[0], j + dx[1]))
print(G.get_connected_component())
```

代码运行截图 （至少包含有"Accepted"）

#44836942提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
class Graph:
    def __init__(self) :
        self.edges = {}
    def addv(self, u) :
        self.edges[u] = []
    def adde(self, u, v) :
        self.edges[u].append(v)
    def get_connected_component(self) :
        res = 0
        vis = {}
        for u in self.edges :
            vis[u] = 0
        for u in self.edges :
            if vis[u] == 1 :
                continue
            res += 1
```

基本信息

#: 44836942
题目: 28170
提交人: 23n2300010724
内存: 3676kB
时间: 19ms
语言: Python3
提交时间: 2024-04-30 17:55:28

02754: 八皇后

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

用时：约10分钟

思路：直接dfs即可

代码

```
def getres(now) :
    n = len(now)
    if n == 8 :
        return [now]
    else :
        res = []
        for j in range(8) :
            flag = True
            for k in range(n) :
                if j == now[k] or j - now[k] == n - k or j - now[k] == k - n :
                    flag = False
            if flag :
                res.extend(getres(now + [j]))
        return res

T = int(input())
res = getres([])
for i in range(T) :
    u = int(input())
    print("".join(str(j + 1) for j in res[u - 1]))
```

代码运行截图 （至少包含有"Accepted"）

#44836977提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def getres(now) :
    n = len(now)
    if n == 8 :
        return [now]
    else :
        res = []
        for j in range(8) :
            flag = True
            for k in range(n) :
                if j == now[k] or j - now[k] == n - k or j - now[k] == 
                    flag = False
            if flag :
                res.extend(getres(now + [j]))
        return res

T = int(input())
res = getres([])
for i in range(T) :
    u = int(input())
    print("".join(str(j + 1) for j in res[u - 1]))
```

基本信息

#: 44836977
题目: 02754
提交人: 23n2300010724
内存: 3628kB
时间: 35ms
语言: Python3
提交时间: 2024-04-30 18:01:56

03151: Pots

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

用时：约30分钟

思路：将两个壶中的水量状态作为点建图，求最短路即可，使用BFS，同时记录最短路上前一个结点。

代码

```
from collections import deque

class Graph:
    def __init__ (self) :
        self.edges = {}
    def addv(self, u) :
        self.edges[u] = []
    def adde(self, u, v, w) :
        self.edges[u].append((v, w))
    def bfs_for_shortest_path(self, start, end) :
        dist = {}
        dist[start] = 0
        vis = set()
        q = deque()
        q.append(start)
        prev = {}
        while q :
            u = q.popleft()
            if u in vis :
                continue
            vis.add(u)
            for e in self.edges[u] :
                v, w = e
                if v not in vis and v not in dist:
                    dist[v] = dist[u] + 1
                    prev[v] = (u, w)
                    q.append(v)
        if end not in dist :
            return -1
        else :
            u = end
            path = []
            while u in prev and u != start :
                e = prev[u]
                path.append(e[1])
                u = e[0]
            path.reverse()
            return path

A, B, C = map(int, input().split())
G = Graph()
for i in range(A + 1) :
    for j in range(B + 1) :
        G.addv((i, j))
G.addv("end")
for i in range(A + 1) :
```

```
    if (i, C) in G.edges :
        G.adde((i, C), "end", "")
for i in range(B + 1) :
    if (C, i) in G.edges :
        G.adde((C, i), "end", "")
for i in range(A + 1) :
    for j in range(B + 1) :
        G.adde((i, j), (0, j), "DROP(1)")
        G.adde((i, j), (i, 0), "DROP(2)")
        G.adde((i, j), (A, j), "FILL(1)")
        G.adde((i, j), (i, B), "FILL(2)")
        G.adde((i, j), (max(i + j - B, 0), min(i + j, B)), "POUR(1,2)")
        G.adde((i, j), (min(i + j, A), max(i + j - A, 0)), "POUR(2,1)")
res = G.bfs_for_shortest_path((0, 0), "end")
if res == -1 :
    print("impossible")
else :
    print(len(res) - 1)
    for i in range(len(res) - 1) :
        print(res[i])
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44837109提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from collections import deque

class Graph:
    def __init__ (self) :
        self.edges = {}
    def addv(self, u) :
        self.edges[u] = []
    def adde(self, u, v, w) :
        self.edges[u].append((v, w))
    def bfs_for_shortest_path(self, start, end) :
        dist = {}
        dist[start] = 0
        vis = set()
        q = deque()
        q.append(start)
        prev = {}
        while q :
            u = q.popleft()
            if u in vis :
                continue
            vis.add(u)
            for e in self.edges[u] :
                v, w = e
                if v not in vis and v not in dist:
```

基本信息

#: 44837109
题目: 03151
提交人: 23n2300010724
内存: 8452kB
时间: 40ms
语言: Python3
提交时间: 2024-04-30 18:31:08

05907: 二叉树的操作

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

用时：约60分钟（中间有个地方一开始写漏了，debug用时比较长）

思路：注意本题并不需要建立二叉树，直接交换对应指向的节点即可

代码

```
t = int(input())
for Case in range(t) :
    n, m = map(int, input().split())
    son = []
    for i in range(n) :
        son.append([0, 0])
    parent = []
    for i in range(n) :
        parent.append([0, 0])
    for j in range(n) :
        i, u, v = map(int, input().split())
        son[i] = [u, v]
        parent[u] = [i, 0]
        parent[v] = [i, 1]
    for i in range(m) :
        s = input().split()
        if s[0] == "1" :
            u = int(s[1])
            v = int(s[2])
            fu, diru = parent[u]
            fv, dirv = parent[v]
            son[fu][diru] = v
            son[fv][dirv] = u
            parent[v] = [fu, diru]
            parent[u] = [fv, dirv]
        if s[0] == "2" :
            u = int(s[1])
            while son[u][0] != -1 :
                u = son[u][0]
            print(u)
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44838448提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
t = int(input())
for Case in range(t) :
    n, m = map(int, input().split())
    son = []
    for i in range(n) :
        son.append([0, 0])
    parent = []
    for i in range(n) :
        parent.append([0, 0])
    for j in range(n) :
        i, u, v = map(int, input().split())
        son[i] = [u, v]
        parent[u] = [i, 0]
        parent[v] = [i, 1]
    for i in range(m) :
        s = input().split()
        if s[0] == "1" :
            u = int(s[1])
```

基本信息

#: 44838448
题目: 05907
提交人: 23n2300010724
内存: 3684kB
时间: 77ms
语言: Python3
提交时间: 2024-04-30 22:32:15

18250: 冰阔落 I

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

用时：约15分钟

思路：并查集

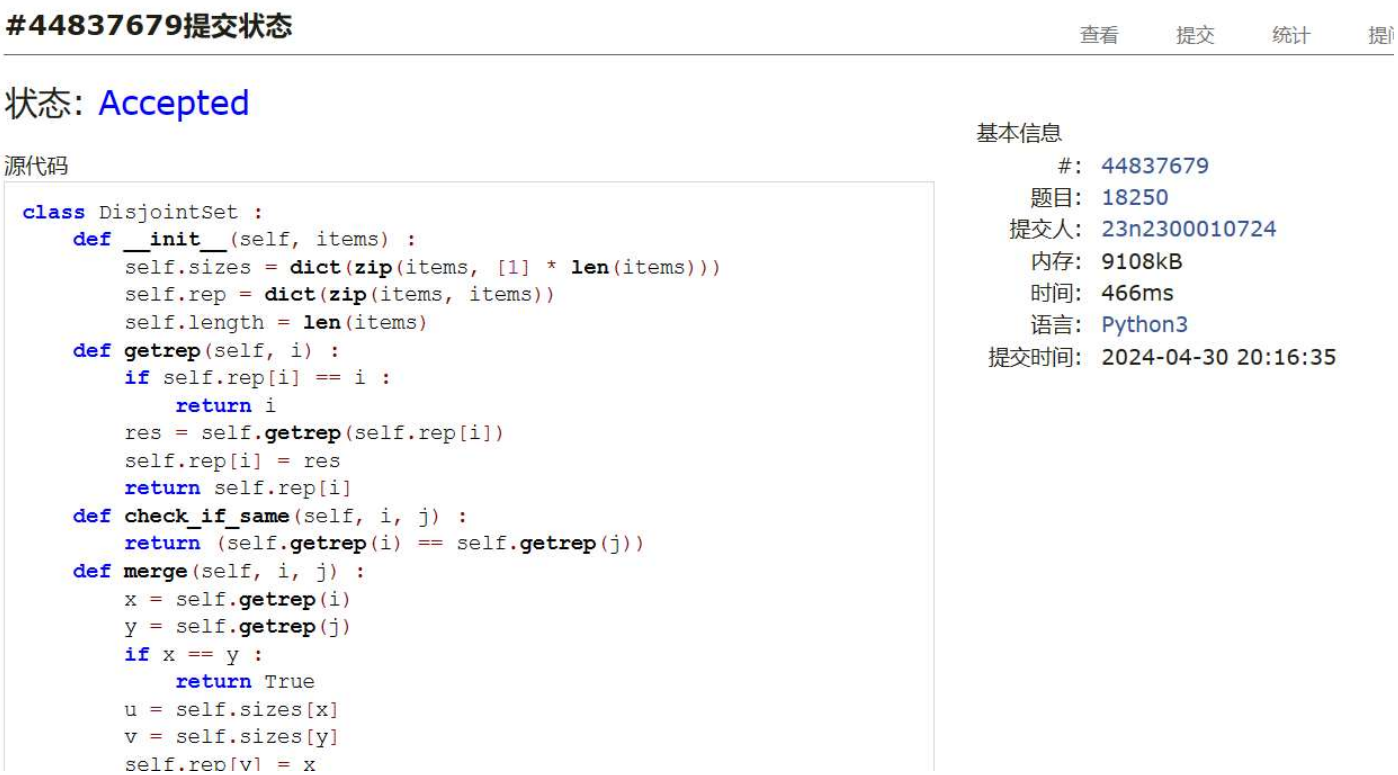
代码

```
class DisjointSet :
    def __init__(self, items) :
        self.sizes = dict(zip(items, [1] * len(items)))
        self.rep = dict(zip(items, items))
        self.length = len(items)
    def getrep(self, i) :
        if self.rep[i] == i :
            return i
        res = self.getrep(self.rep[i])
        self.rep[i] = res
        return self.rep[i]
    def check_if_same(self, i, j) :
        return (self.getrep(i) == self.getrep(j))
    def merge(self, i, j) :
        x = self.getrep(i)
        y = self.getrep(j)
        if x == y :
            return True
        u = self.sizes[x]
        v = self.sizes[y]
        self.rep[y] = x
        self.sizes[x] = u + v
        return False
    def count(self) :
        res = 0
        for item in self.rep :
            if self.getrep(item) == item :
                res += 1
        return res

while True :
    try :
        n, m = map(int, input().split())
        S = DisjointSet(list(i for i in range(1, n + 1)))
        for i in range(m) :
            u, v = map(int, input().split())
            flag = S.merge(u, v)
            if flag :
                print("Yes")
            else :
                print("No")
        print(S.count())
        for i in S.rep :
            if S.rep[i] == i :
                print(i, end = " ")
        print()
```

```
    except:
        break
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



#44837679提交状态

查看 提交 统计 提

状态: Accepted

源代码

```
class DisjointSet :
    def __init__(self, items) :
        self.sizes = dict(zip(items, [1] * len(items)))
        self.rep = dict(zip(items, items))
        self.length = len(items)
    def getrep(self, i) :
        if self.rep[i] == i :
            return i
        res = self.getrep(self.rep[i])
        self.rep[i] = res
        return self.rep[i]
    def check_if_same(self, i, j) :
        return (self.getrep(i) == self.getrep(j))
    def merge(self, i, j) :
        x = self.getrep(i)
        y = self.getrep(j)
        if x == y :
            return True
        u = self.sizes[x]
        v = self.sizes[y]
        self.rep[v] = x
```

基本信息

#: 44837679
题目: 18250
提交人: 23n2300010724
内存: 9108kB
时间: 466ms
语言: Python3
提交时间: 2024-04-30 20:16:35

05443: 兔子与樱花

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

用时：约40分钟

思路：用Dijkstra算法求最短路并记录路径上的上一个节点

代码

```
import heapq

class Graph :
    def __init__(self) :
        self.edges = {}
    def addv(self, name) :
        self.edges[name] = []
    def addedge(self, u, v, w) :
        if u in self.edges and v in self.edges:
            self.edges[u].append((v, w))
    def dijkstra(self, start, end) :
        dist = {}
        prev = {}
        for i in self.edges :
            dist[i] = -1
        for i in self.edges :
            prev[i] = None
        dist[start] = 0
        vis = set()
        q = [(0, start)]
        heapq.heapify(q)
        while q :
            d, u = heapq.heappop(q)
            #print(d, u)
            if u == end :
                break
            if u in vis :
                continue
            dist[u] = d
            vis.add(u)
            for e in self.edges[u] :
                v, w = e
                if (dist[u] + w < dist[v] or dist[v] == -1) and v not in vis :
                    dist[v] = dist[u] + w
                    prev[v] = (u, v, w)
                heapq.heappush(q, (dist[v], v))
        return dist, prev
n = int(input())
G = Graph()
for i in range(n) :
    G.addv(input().strip())
m = int(input())
for i in range(m) :
    s = input().split()
    G.addedge(s[0], s[1], int(s[2]))
    G.addedge(s[1], s[0], int(s[2]))
```

```
T = int(input())
for i in range(T) :
    s = input().split()
    dist, prev = G.dijkstra(s[0], s[1])
    u = s[1]
    path = []
    while u != s[0] :
        path.append(prev[u])
        u = prev[u][0]
    path.reverse()
    print(s[0], end="")
    for p in path :
        print("->(", p[2], ")->", p[1], sep="", end="")
    print()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44837902提交状态

查看 提交 统计 提问

状态: Accepted

基本信息

#: 44837902
题目: 05443
提交人: 23n2300010724
内存: 3712kB
时间: 21ms
语言: Python3
提交时间: 2024-04-30 20:54:55

源代码

```
import heapq

class Graph :
    def __init__(self) :
        self.edges = {}
    def addv(self, name) :
        self.edges[name] = []
    def addedge(self, u, v, w) :
        if u in self.edges and v in self.edges:
            self.edges[u].append((v, w))
    def dijkstra(self, start, end) :
        dist = {}
        prev = {}
        for i in self.edges :
            dist[i] = -1
        for i in self.edges :
            prev[i] = None
        dist[start] = 0
        vis = set()
        q = [(0, start)]
        heapq.heapify(q)
        while q :
            d, u = heapq.heappop(q)
            #print(d, u)
```

2. 学习总结和收获

感觉做这次作业的过程中发现图论的很多题目细节都比较多，包括经典算法实现的时候，如果不小心写错了debug也比较困难，所以以后还是要多熟悉一下各种算法的标准实现。