Assignment #B: 图论和树算

Updated 1709 GMT+8 Apr 28, 2024

2024 spring, Complied 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

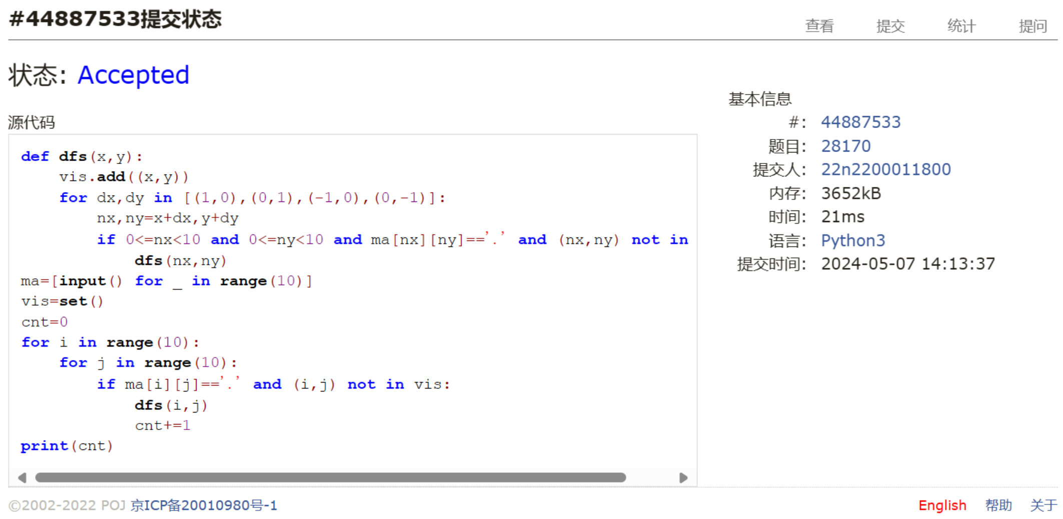
28170: 算鹰

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

代码

```
def dfs(x,y):
    vis.add((x,y))
    for dx,dy in [(1,0),(0,1),(-1,0),(0,-1)]:
        nx,ny=x+dx,y+dy
        if 0<=nx<10 and 0<=ny<10 and ma[nx][ny]=='.' and (nx,ny) not in vis:
            dfs(nx,ny)
ma=[input() for _ in range(10)]
vis=set()
cnt=0
for i in range(10):
    for j in range(10):
        if ma[i][j]=='.' and (i,j) not in vis:
            dfs(i,j)
            cnt+=1
print(cnt)
```

```
##先用dfs即可
```



#44887533提交状态

状态: Accepted

源代码

```
def dfs(x,y):
    vis.add((x,y))
    for dx,dy in [(1,0),(0,1),(-1,0),(0,-1)]:
        nx,ny=x+dx,y+dy
        if 0<=nx<10 and 0<=ny<10 and ma[nx][ny]=='.' and (nx,ny) not in
            dfs(nx,ny)
ma=[input() for _ in range(10)]
vis=set()
cnt=0
for i in range(10):
    for j in range(10):
        if ma[i][j]=='.' and (i,j) not in vis:
            dfs(i,j)
            cnt+=1
print(cnt)
```

基本信息

#: 44887533
题目: 28170
提交人: 22n2200011800
内存: 3652kB
时间: 21ms
语言: Python3
提交时间: 2024-05-07 14:13:37

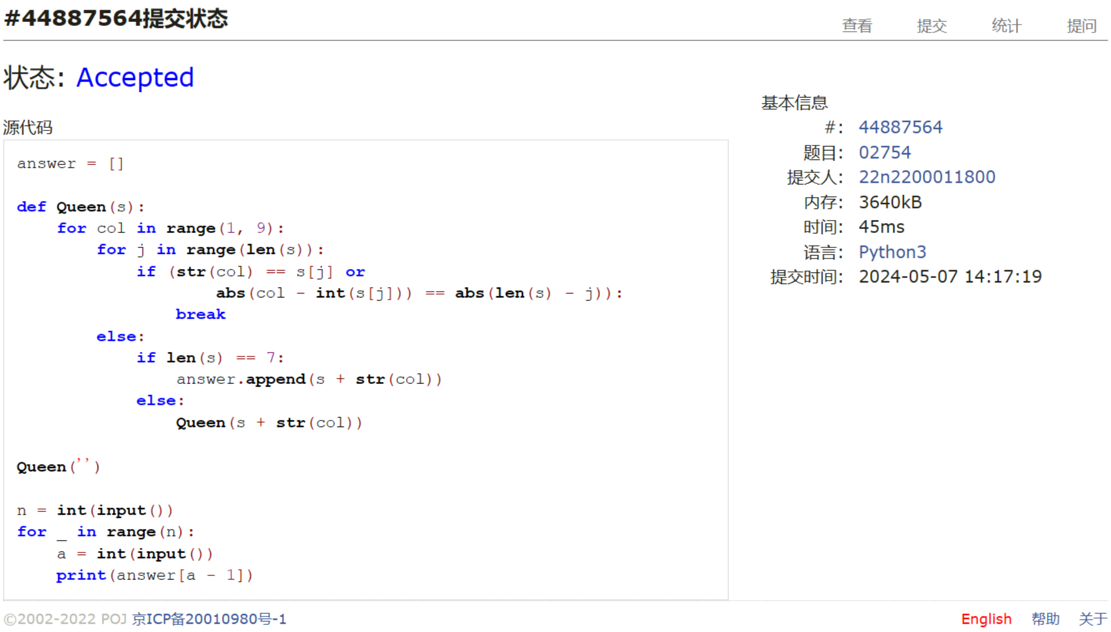
©2002-2022 POJ 京ICP备20010980号-1

02754: 八皇后

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

代码

```
answer = []

def Queen(s):
    for col in range(1, 9):
        for j in range(len(s)):
            if (str(col) == s[j] or abs(col - int(s[j])) == abs(len(s) - j)):
                break
        else:
            if len(s) == 7:
                answer.append(s + str(col))
            else:
                Queen(s + str(col))

Queen('')

n = int(input())
for _ in range(n):
    a = int(input())
    print(answer[a - 1])
##学会了如何用dfs写八皇后，之前都是硬写
```



#44887564提交状态

状态: Accepted

源代码

```
answer = []

def Queen(s):
    for col in range(1, 9):
        for j in range(len(s)):
            if (str(col) == s[j] or
                    abs(col - int(s[j])) == abs(len(s) - j)):
                break
        else:
            if len(s) == 7:
                answer.append(s + str(col))
            else:
                Queen(s + str(col))

Queen('')

n = int(input())
for _ in range(n):
    a = int(input())
    print(answer[a - 1])
```

基本信息

#:	44887564
题目:	02754
提交人:	22n2200011800
内存:	3640kB
时间:	45ms
语言:	Python3
提交时间:	2024-05-07 14:17:19

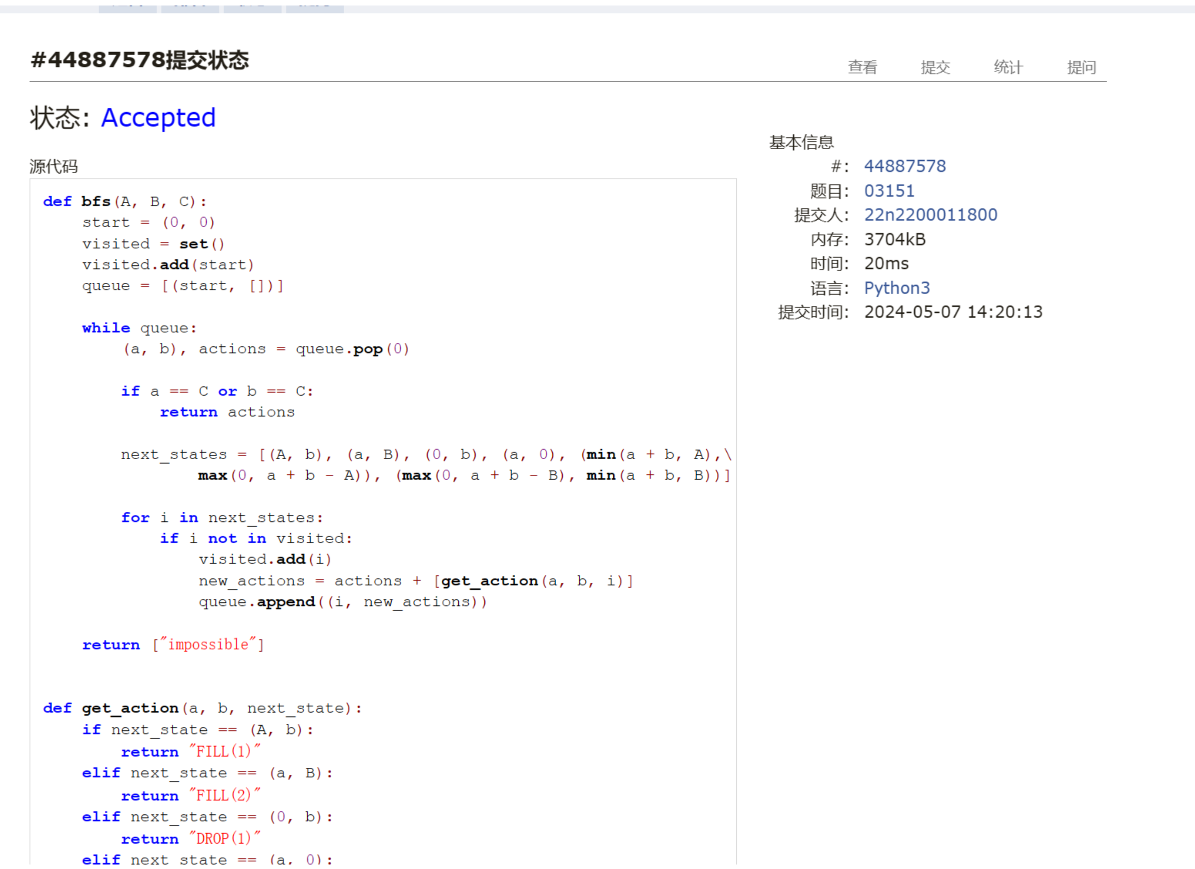
©2002-2022 POJ 京ICP备20010980号-1

03151: Pots

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

代码

```
def bfs(A, B, C):
    start = (0, 0)
    visited = set()
    visited.add(start)
    queue = [(start, [])]

    while queue:
        (a, b), actions = queue.pop(0)

        if a == C or b == C:
            return actions

        next_states = [(A, b), (a, B), (0, b), (a, 0), (min(a + b, A),\
                max(0, a + b - A)), (max(0, a + b - B), min(a + b, B))]

        for i in next_states:
            if i not in visited:
                visited.add(i)
                new_actions = actions + [get_action(a, b, i)]
                queue.append((i, new_actions))

    return ["impossible"]


def get_action(a, b, next_state):
    if next_state == (A, b):
        return "FILL(1)"
    elif next_state == (a, B):
        return "FILL(2)"
    elif next_state == (0, b):
        return "DROP(1)"
    elif next_state == (a, 0):
        return "DROP(2)"
    elif next_state == (min(a + b, A), max(0, a + b - A)):
        return "POUR(2,1)"
    else:
        return "POUR(1,2)"


A, B, C = map(int, input().split())
solution = bfs(A, B, C)

if solution == ["impossible"]:
    print(solution[0])
else:
    print(len(solution))
    for i in solution:
        print(i)
##其实就是直接的bfs
```



#44887578提交状态

状态: Accepted

源代码

```
def bfs(A, B, C):
    start = (0, 0)
    visited = set()
    visited.add(start)
    queue = [(start, [])]

    while queue:
        (a, b), actions = queue.pop(0)

        if a == C or b == C:
            return actions

        next_states = [(A, b), (a, B), (0, b), (a, 0), (min(a + b, A),\
                max(0, a + b - A)), (max(0, a + b - B), min(a + b, B))]

        for i in next_states:
            if i not in visited:
                visited.add(i)
                new_actions = actions + [get_action(a, b, i)]
                queue.append((i, new_actions))

    return ["impossible"]


def get_action(a, b, next_state):
    if next_state == (A, b):
        return "FILL(1)"
    elif next_state == (a, B):
        return "FILL(2)"
    elif next_state == (0, b):
        return "DROP(1)"
    elif next_state == (a, 0):
```

基本信息

#: 44887578
题目: 03151
提交人: 22n2200011800
内存: 3704kB
时间: 20ms
语言: Python3
提交时间: 2024-05-07 14:20:13

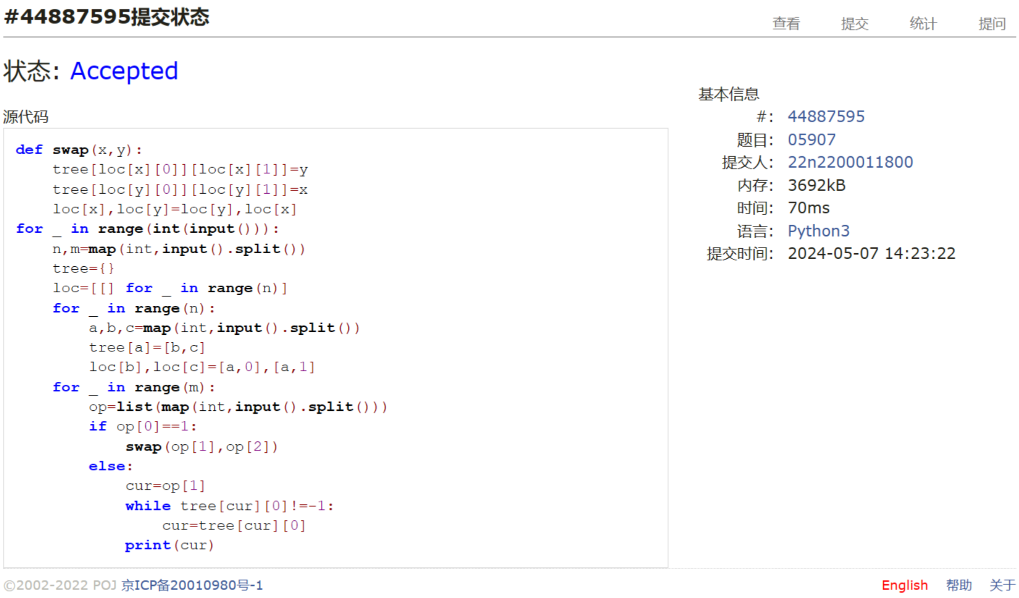
05907: 二叉树的操作

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

代码

```
def swap(x,y):
    tree[loc[x][0]][loc[x][1]]=y
    tree[loc[y][0]][loc[y][1]]=x
```

```
        loc[x],loc[y]=loc[y],loc[x]
for _ in range(int(input())):
    n,m=map(int,input().split())
    tree={}
    loc=[[] for _ in range(n)]
    for _ in range(n):
        a,b,c=map(int,input().split())
        tree[a]=[b,c]
        loc[b],loc[c]=[a,0],[a,1]
    for _ in range(m):
        op=list(map(int,input().split()))
        if op[0]==1:
            swap(op[1],op[2])
        else:
            cur=op[1]
            while tree[cur][0]!=-1:
                cur=tree[cur][0]
            print(cur)
##学习了同学用列表和字典的写法
```



#44887595提交状态

状态: Accepted

源代码

```
def swap(x,y):
    tree[loc[x][0]][loc[x][1]]=y
    tree[loc[y][0]][loc[y][1]]=x
    loc[x],loc[y]=loc[y],loc[x]
for _ in range(int(input())):
    n,m=map(int,input().split())
    tree={}
    loc=[[] for _ in range(n)]
    for _ in range(n):
        a,b,c=map(int,input().split())
        tree[a]=[b,c]
        loc[b],loc[c]=[a,0],[a,1]
    for _ in range(m):
        op=list(map(int,input().split()))
        if op[0]==1:
            swap(op[1],op[2])
        else:
            cur=op[1]
            while tree[cur][0]!=-1:
                cur=tree[cur][0]
            print(cur)
```

基本信息

#:	44887595
题目:	05907
提交人:	22n2200011800
内存:	3692kB
时间:	70ms
语言:	Python3
提交时间:	2024-05-07 14:23:22

©2002-2022 POJ 京ICP备20010980号-1

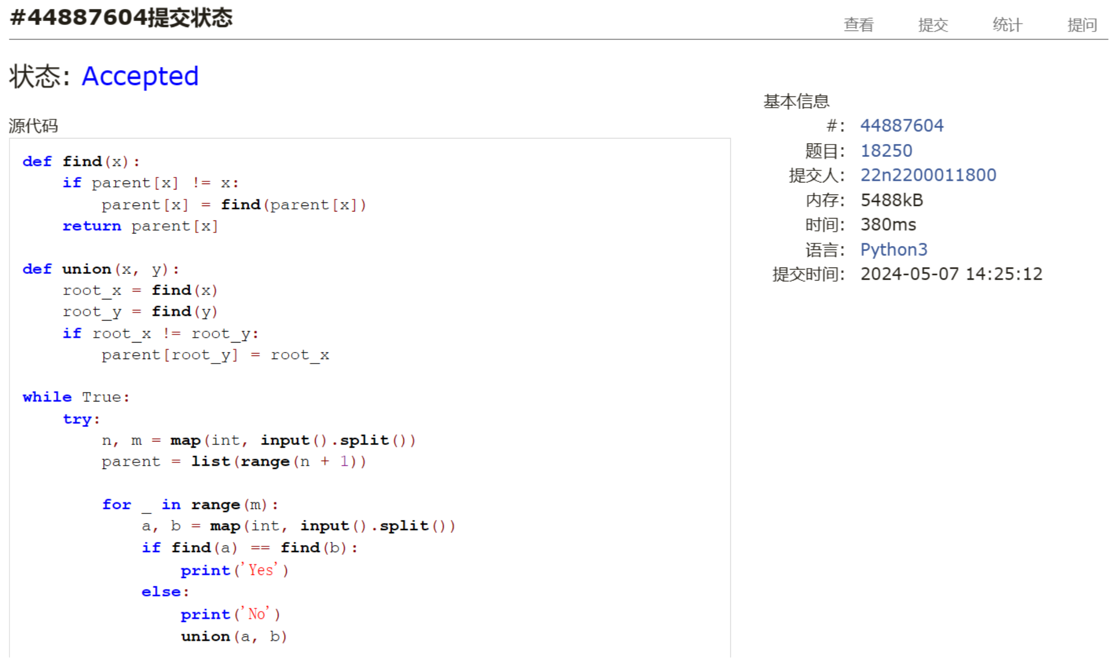
18250: 冰阔落 I

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

代码

```
def find(x):
    if parent[x] != x:
        parent[x] = find(parent[x])
    return parent[x]

def union(x, y):
    root_x = find(x)
    root_y = find(y)
    if root_x != root_y:
        parent[root_y] = root_x
```

```
while True:
    try:
        n, m = map(int, input().split())
        parent = list(range(n + 1))

        for _ in range(m):
            a, b = map(int, input().split())
            if find(a) == find(b):
                print('Yes')
            else:
                print('No')
                union(a, b)

        unique_parents = set(find(x) for x in range(1, n + 1))
        ans = sorted(unique_parents)
        print(len(ans))
        print(*ans)

    except EOFError:
        break
##学习了并查集
```



#44887604提交状态

状态: Accepted

源代码

```
def find(x):
    if parent[x] != x:
        parent[x] = find(parent[x])
    return parent[x]

def union(x, y):
    root_x = find(x)
    root_y = find(y)
    if root_x != root_y:
        parent[root_y] = root_x

while True:
    try:
        n, m = map(int, input().split())
        parent = list(range(n + 1))

        for _ in range(m):
            a, b = map(int, input().split())
            if find(a) == find(b):
                print('Yes')
            else:
                print('No')
                union(a, b)
```

基本信息

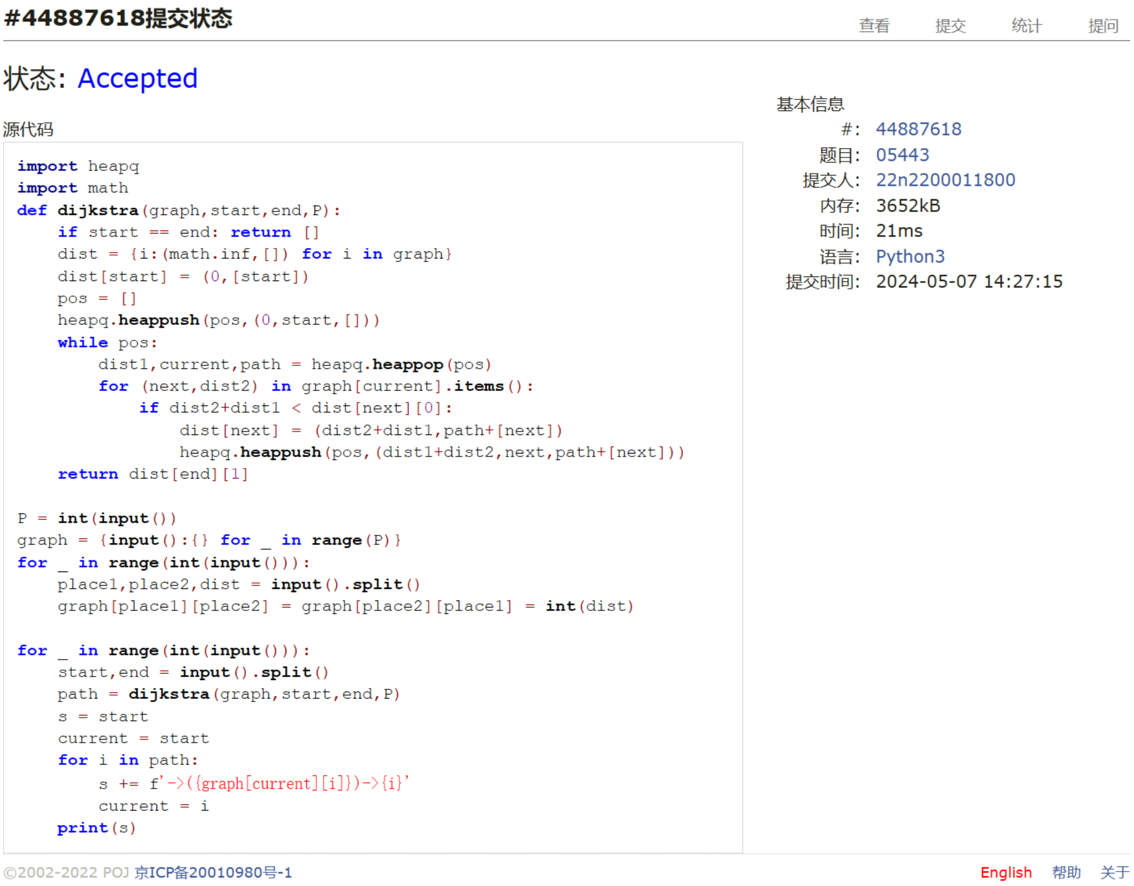
#:	44887604
题目:	18250
提交人:	22n2200011800
内存:	5488kB
时间:	380ms
语言:	Python3
提交时间:	2024-05-07 14:25:12

05443: 兔子与樱花

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

代码

```
import heapq
import math
def dijkstra(graph,start,end,P):
    if start == end: return []
    dist = {i:(math.inf,[]) for i in graph}
    dist[start] = (0,[start])
    pos = []
    heapq.heappush(pos,(0,start,[]))
    while pos:
        dist1,current,path = heapq.heappop(pos)
        for (next,dist2) in graph[current].items():
            if dist2+dist1 < dist[next][0]:
                dist[next] = (dist2+dist1,path+[next])
                heapq.heappush(pos,(dist1+dist2,next,path+[next]))
    return dist[end][1]

P = int(input())
graph = {input():{} for _ in range(P)}
for _ in range(int(input())):
    place1,place2,dist = input().split()
    graph[place1][place2] = graph[place2][place1] = int(dist)

for _ in range(int(input())):
    start,end = input().split()
    path = dijkstra(graph,start,end,P)
    s = start
    current = start
    for i in path:
        s += f'->({graph[current][i]})->{i}'
        current = i
    print(s)
##添加多一条路径即可
```



#44887618提交状态

状态: Accepted

源代码

```
import heapq
import math
def dijkstra(graph,start,end,P):
    if start == end: return []
    dist = {i:(math.inf,[]) for i in graph}
    dist[start] = (0,[start])
    pos = []
    heapq.heappush(pos,(0,start,[]))
    while pos:
        dist1,current,path = heapq.heappop(pos)
        for (next,dist2) in graph[current].items():
            if dist2+dist1 < dist[next][0]:
                dist[next] = (dist2+dist1,path+[next])
                heapq.heappush(pos,(dist1+dist2,next,path+[next]))
    return dist[end][1]

P = int(input())
graph = {input():{} for _ in range(P)}
for _ in range(int(input())):
    place1,place2,dist = input().split()
    graph[place1][place2] = graph[place2][place1] = int(dist)

for _ in range(int(input())):
    start,end = input().split()
    path = dijkstra(graph,start,end,P)
    s = start
    current = start
    for i in path:
        s += f'->({graph[current][i]})->{i}'
        current = i
    print(s)
```

基本信息

#:	44887618
题目:	05443
提交人:	22n2200011800
内存:	3652kB
时间:	21ms
语言:	Python3
提交时间:	2024-05-07 14:27:15

©2002-2022 POJ 京ICP备20010980号-1

2. 学习总结和收获

在恶补数算之前落下来的东西，现在每天翻阅几周前的群消息“回溯”，学到了很多。