ASSIGNMENT-5.9

M.Paramesh

HNo: 2303A53021

Batch – 46

Q)Community Detection in a Social Networking Platform

You are developing a social networking platform where users can form friendships. If two users are directly or indirectly connected through friendships, they belong to the same community. Initially, each user is independent. As friendships are formed, communities merge.

Your task is to determine whether two users belong to the same community after processing friendship relationships. To efficiently manage and query user connections, use the Disjoint Set Union (Union–Find) data structure.

Task

1. Initialize each user as a separate set.

2. Process friendship connections using Union operations.

3. Answer queries to check if two users are in the same community.

Operations

-Union(u, v): Create a friendship between users u and v

- Find(u): Identify the community representative of user u

- Connected(u, v): Check if users u and v belong to the same community

Example Test Case 1

Input

- Number of users: 7

- Number of friendships: 5

Friendships:

(0,1)

(1,3)

(2,4)

(5,6)

(3,4)

Queries:

(0,4)

(2,6)

(5,6)

Output

YES

NO

YES

CODE :

```
class DisjointSet:
    def __init__(self, n):
        self.parent = list(range(n))
        self.rank = [0] * n

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

    def union(self, a, b):
        ra = self.find(a)
        rb = self.find(b)
        if ra != rb:
```

```
        if self.rank[ra] < self.rank[rb]:
            self.parent[ra] = rb
        elif self.rank[ra] > self.rank[rb]:
            self.parent[rb] = ra
        else:
            self.parent[rb] = ra
            self.rank[ra] += 1
    def connected(self, a, b):
        return self.find(a) == self.find(b)
n = int(input().strip())
f = int(input().strip())
dsu = DisjointSet(n)
for _ in range(f):
    u, v = map(int, input().split())
    dsu.union(u, v)
q = int(input().strip())
for _ in range(q):
    u, v = map(int, input().split())
    print("YES" if dsu.connected(u, v) else "NO")
```

output:


election View Go Run Terminal Help
Untitled-1.py
C: > Users > param > OneDrive > Documents > Untitled-1.py > ...
class DisjointSet:
def __init__(self, n):
self.parent = list(range(n))
self.rank = [0] * n
def find(self, x):
if self.parent[x] != x:
self.parent[x] = self.find(self.parent[x])
return self.parent[x]
def union(self, a, b):
ra = self.find(a)
rb = self.find(b)
if ra != rb:
if self.rank[ra] < self.rank[rb]:
self.parent[ra] = rb
elif self.rank[ra] > self.rank[rb]:
self.parent[rb] = ra
else:
self.parent[rb] = ra
self.rank[ra] += 1
def connected(self, a, b):
return self.find(a) == self.find(b)
n = int(input().strip())
f = int(input().strip())
dsu = DisjointSet(n)
for _ in range(f):
u, v = map(int, input().split())
dsu.union(u, v)
q = int(input().strip())
for _ in range(q):
u, v = map(int, input().split())
print("YES" if dsu.connected(u, v) else "NO")
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\Users\param> & C:/Users/param/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/param/OneDrive/Documents/Untitled-1.py
7
5
0 1
1 2
2 4
5 6
3 4
3
0 4
YES
2 6
NO
5 6
YES
PS C:\Users\param>
pleted.