

COMPETITIVE PROGRAMMING

Assignment-8

2303A51463

B-07

Implementing Disjoint Set Union -DSU for Connectivity Problems

Electrical Grid Connectivity Monitoring:

Algorithm:

1. Start.
2. Read the number of power stations n .
3. Initialize an array `parent` of size n .
4. For each station i from 0 to $n-1$, set `parent[i] = i`.
5. Define function `find(x)`:
 - a. If `parent[x] == x`, return x .
 - b. Else set `parent[x] = find(parent[x])` and return `parent[x]`.
6. Define function `union(x, y)`:
 - a. Find root of x using `find(x)`.
 - b. Find root of y using `find(y)`.
 - c. If roots are different, set `parent[rootY] = rootX`.
7. Read the number of power lines.
8. For each power line connection (u, v) :
 - a. Call `union(u, v)` to merge the zones.
9. Read the number of connectivity queries.
10. For each query (u, v) :
 - a. If `find(u) == find(v)`, print YES.
 - b. Else, print NO.
11. Stop.

```

ass8.4.py > ...
1  class DSU:
2      def __init__(self,n):
3          self.parent=[i for i in range(n)]
4      def find(self,x):
5          if self.parent[x]!=x:
6              self.parent[x]=self.find(self.parent[x])
7          return self.parent[x]
8      def union(self,x,y):
9          px=self.find(x)
10         py=self.find(y)
11         if px!=py:
12             self.parent[py]=px
13
14     n=7
15     power_lines=[(0,1),(1,2),(3,4),(5,6)]
16     queries=[(0,2),(2,4),(5,6)]
17
18     dsu=DSU(n)
19
20     for u,v in power_lines:
21         dsu.union(u,v)
22
23     for u,v in queries:
24         print("YES" if dsu.find(u)==dsu.find(v) else "NO")
25

```

Output:

```

PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/harsh/
thon.exe c:/Users/harsh/OneDrive/Desktop/CP/ass8.4.py
● YES
  NO
  YES

```

Campus Wi-Fi Network Connectivity:

Algorithm:

1. Start.
2. Read number of buildings n .
3. Initialize array `parent` of size n .
4. Set `parent[i] = i` for all buildings.
5. Define `find(x)`:
 - a. If `parent[x] == x`, return x .
 - b. Else set `parent[x] = find(parent[x])` and return it.
6. Define `union(x, y)`:
 - a. Find roots of x and y .
 - b. If different, make one root parent of the other.
7. Read number of network connections m .
8. For each connection (u, v) , perform `union(u, v)`.
9. Read number of queries q .
10. For each query (u, v) :
 - a. If `find(u) == find(v)`, print YES.
 - b. Else print NO.
11. Stop.

```

ass8.5.py > ...
1  class DSU:
2      def __init__(self,n):
3          self.parent=[i for i in range(n)]
4      def find(self,x):
5          if self.parent[x]!=x:
6              self.parent[x]=self.find(self.parent[x])
7          return self.parent[x]
8      def union(self,x,y):
9          px=self.find(x)
10         py=self.find(y)
11         if px!=py:
12             self.parent[py]=px
13
14     n=int(input())
15     dsu=DSU(n)
16
17     m=int(input())
18     for _ in range(m):
19         u,v=map(int,input().split())
20         dsu.union(u,v)
21
22     q=int(input())
23     for _ in range(q):
24         u,v=map(int,input().split())
25         print("YES" if dsu.find(u)==dsu.find(v) else "NO")
26

```

Output:

```

PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/harsh/AppData/L
thon.exe c:/Users/harsh/OneDrive/Desktop/CP/ass8.5.py
9
5
0 1
1 2
3 4
5 6
6 7
3
0 2
YES
2 4
NO
5 7
YES

```

