Crack-A-Hack

Even Trees

Course: Algorithmic Problem Solving

Course Code: 17ECSE309

Anup Patil

USN: 01FE15BCS031

Introduction

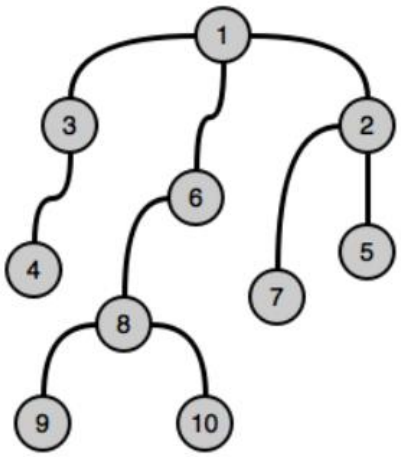
*A **tree** is a connected graph with no cycles and there's only one way to get from one node to another, but this isn't true in general graphs.*

*A **forest** is a bunch of trees.*

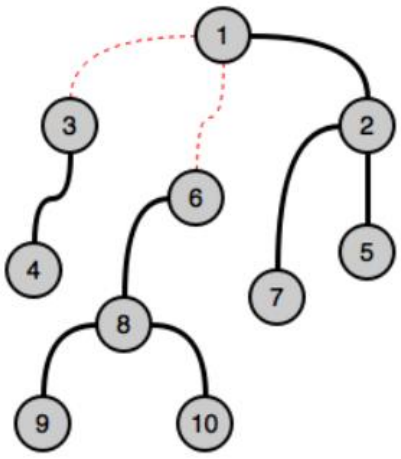
Problem

Given a tree, find the maximum number of edges you can remove from the tree to get a forest such that each connected component of the forest contains an even number of nodes.

Original tree:



Decomposed tree:



Solution

Try counting the children. If the subtree has even number of nodes then the edge leading to this subtree can be removed. Otherwise, you have to keep on searching until you find a suitable edge or the entire tree exhausted. As it always can be decomposed into forests of even number of nodes, you will always end up with an answer greater than 1.

```
edges = []
delete = 0
in1 =  input().split()

for j in range(int(in1[1])):
    i=input().split()
    edges.append((i[0],i[1]))
adj_list = dict()

for start, end in edges:
    if end in adj_list:
        adj_list[end].append(start)
    else:
        adj_list[end]=[start]

def dfs_iter(graph, root):
    visited = []
    stack = [root, ]
    while len(stack)!=0:
        node = stack.pop()
        if node not in visited:
            visited.append(node)
            if node not in graph:
                graph[node]=[]
            stack.extend([x for x in graph[node] if x not in visited])
    return visited

b = []
for a in adj_list:
   b.append(a)

for a in b:
   if  len(dfs_iter(adj_list, a)) % 2 == 0:
      delete += 1

print (delete-1)
```

References

- Problem
  - https://www.hackerrank.com/challenges/even-tree/problem

- Trees Explanation
  - https://en.wikipedia.org/wiki/Tree_(graph_theory)