cenkay ★ 3413 Last Edit: November 3, 2019 1:53 AM 1.7K VIEWS

21

- BFS

```
class Solution:
    def treeDiameter(self, edges: List[List[int]], move: int = 0) -> int:
        graph = collections.defaultdict(set)
        for a, b in edges:
            graph[a].add(b)
            graph[b].add(a)
        bfs = {(u, None) for u, nex in graph.items() if len(nex) == 1}
        while bfs:
            bfs, move = {(v, u) for u, pre in bfs for v in graph[u] if v != pre}, move + 1
        return max(move - 1, 0)
```

- DFS

```
class Solution:
    diameter = 0
    def treeDiameter(self, edges: List[List[int]], move: int = 0) -> int:
        def dfs(node, pre):
            d1 = d2 = 0
            for nex in graph[node]:
                if nex != pre:
                    d = dfs(nex, node)
                    if d > d1:
                        d1, d2 = d, d1
                    elif d > d2:
                        d2 = d
            self.diameter = max(self.diameter, d1 + d2)
            return d1 + 1
        graph = collections.defaultdict(set)
        for a, b in edges:
            graph[a].add(b)
            graph[b].add(a)
        dfs(0, None)
        return self.diameter
```


Type comment here... (Markdown is supported)

Post

eric496 ★ 114 Last Edit: March 22, 2020 5:00 AM

Here is a more straightforward and readable BFS solution:

```
from collections import deque

class Solution:
    def treeDiameter(self, edges: List[List[int]]) -> int:
        if not edges:
            return 0

        adjacent = {}
```

Read More

1 Reply

shikzheng ★ 38 November 11, 2019 5:45 AM

What's the run time and space of both solutions?

0 Reply

darrenfu42 ★ 15 Last Edit: November 2, 2019 11:57 PM

Can you explain the idea behind this line?

```
bfs = {(v, u) for u, pre in bfs for v in graph[u] if v != pre}
```

0 Show 5 replies Reply