Our Solution(s) Run Code

Solution 1

```python
# Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class AncestralTree:
    def __init__(self, name):
        self.name = name
        self.ancestor = None


# O(d) time | O(1) space - where d is the depth (height) of the ancestral tree
def getYoungestCommonAncestor(topAncestor, descendantOne, descendantTwo):
    depthOne = getDescendantDepth(descendantOne, topAncestor)
    depthTwo = getDescendantDepth(descendantTwo, topAncestor)
    if depthOne > depthTwo:
        return backtrackAncestralTree(descendantOne, descendantTwo, depthOne - de
    else:
        return backtrackAncestralTree(descendantTwo, descendantOne, depthTwo - de


def getDescendantDepth(descendant, topAncestor):
    depth = 0
    while descendant != topAncestor:
        depth += 1
        descendant = descendant.ancestor
    return depth


def backtrackAncestralTree(lowerDescendant, higherDescendant, diff):
    while diff > 0:
        lowerDescendant = lowerDescendant.ancestor
        diff -= 1
    while lowerDescendant != higherDescendant:
        lowerDescendant = lowerDescendant.ancestor
        higherDescendant = higherDescendant.ancestor
    return lowerDescendant

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```python
# This is an input class. Do not edit.
class AncestralTree:
    def __init__(self, name):
        self.name = name
        self.ancestor = None


def getYoungestCommonAncestor(topAncestor, descendantOne, descendantTwo):
    # Write your code here.
    pass

```

Custom Output Raw Output Submit Code

Run or submit code when you're ready.