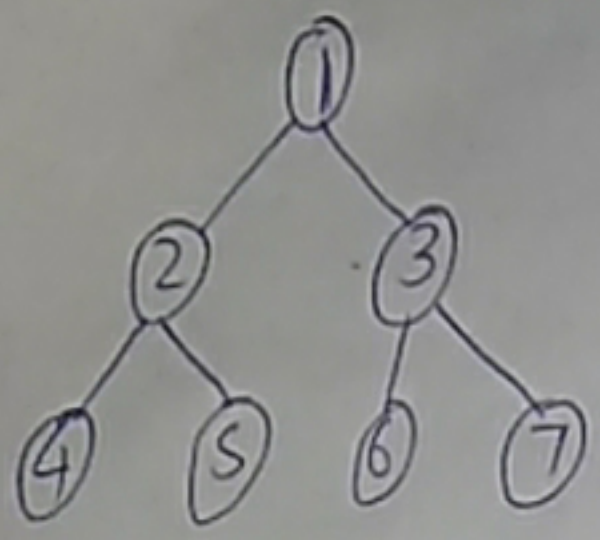
Day 21

Lowest Common Ancestor in a binary tree



Examples

⇒ LCA(4,5) = 2

LCA(4,6) = 1

LCA(3,4) = 1

LCA(2,4) = 2

* find path from root to n1 and store it in vector or array
* find path from root to n2 and store it in another vecor or array
* Traverse both paths till the values in array are same. Return the common element just before the mismatch

Implementation:

```
Class Node:
    def __init__(self, key)
        self.key = key
        self.right = None
        self.left = None
```

```
def findlca(root, n1, n2):
    If root is None:
        return None
    if root.key == n1 or root.key == n2:
        return root
    left_lca = findlca(root.left, n1, n2)
    right_lca = findlca(root.right, n1, n2)
    if left_lca and right_lca:
        return root
    return left_lca if lca is not None
                    else right_lca
```

Input:

```
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)
root.right.left = Node(6)
root.right.right = Node(7)
print("LCA(4,5) = ", findlca(root,4,5).key)
print("LCA(4,6) = ", findlca(root,4,6).key)
```

Output:

LCA(4,5) = 2

LCA(4,6) = 1

Diameter of Binary tree

The diameter of a tree is the number of nodes on the longest path between two end nodes.

* the diameter of T's left subtree
* the diameter of T's right subtree
* The longest path between leaves that goes thorough root t.

Implementation:

```
Class Node:

    def __init__(self,data):
        self.data = data
        self.left = self.right = None

class Height():

    def __init__(self):
        self.h = 0
```

```
def diameter0(root, height):
    lh = Height()
    rh = Height()
    If root is None:
        height.h = 0
        return 0

    ldiameter = diameter0(root.left, lh)
    rdiameter = diameter0(root.right, rh)
    height.h = max(lh.h, rh.h) + 1
    return max( lh.h + rh.h +1, max(ldiameter0,
                                    rdiameter0))

def diameter (root):
    height = Height()
    return diameter 0(root, height)
```

Input :

```
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)
print (diameter(root))
```

output :-

5 4