Q1

class Solution:

def isSumProperty(self, root):

# code here

x,y=0,0

l1,r1=1,1

if root==None:

return 1

if root.left==None and root.right==None:

return 1

if root.left:

x=root.left.data

l1 = self.isSumProperty(root.left)

if root.right:

y=root.right.data

r1 = self.isSumProperty(root.right)

if l1==1 and r1==1 and root.data==x+y:

return 1

else:

return 0

Q2

class Solution:

def lca(self,root,a,b):

if root == None or root.data == a or root.data==b:

return root

leftlca = self.lca(root.left,a,b)

rightlca = self.lca(root.right,a,b)

if not leftlca:

return rightlca

if not rightlca:

return leftlca

return root

def distance(self,root,a):

if root == None:

return -1

if root.data == a:

return 0

leftDist = self.distance(root.left,a)

rightDist = self.distance(root.right,a)

if leftDist == -1 and rightDist == -1:

return -1

elif rightDist == -1:

return leftDist+1

else:

return rightDist+1

def findDist(self,root,a,b):

lca = self.lca(root,a,b)

return self.distance(lca,a)+self.distance(lca,b)

Q3

class Node:

def \_\_init\_\_(self, val):

self.right = None

self.data = val

self.left = None

prev = None

def BinaryTree2DoubleLinkedList(root):

if root is None:

return root

head = BinaryTree2DoubleLinkedList(root.left);

global prev

if prev is None :

head = root

else:

root.left = prev

prev.right = root

prev = root;

BinaryTree2DoubleLinkedList(root.right);

return head

def print\_dll(head):

while head is not None:

print(head.data, end=" ")

head = head.right

head = BinaryTree2DoubleLinkedList(root)

print\_dll(head)