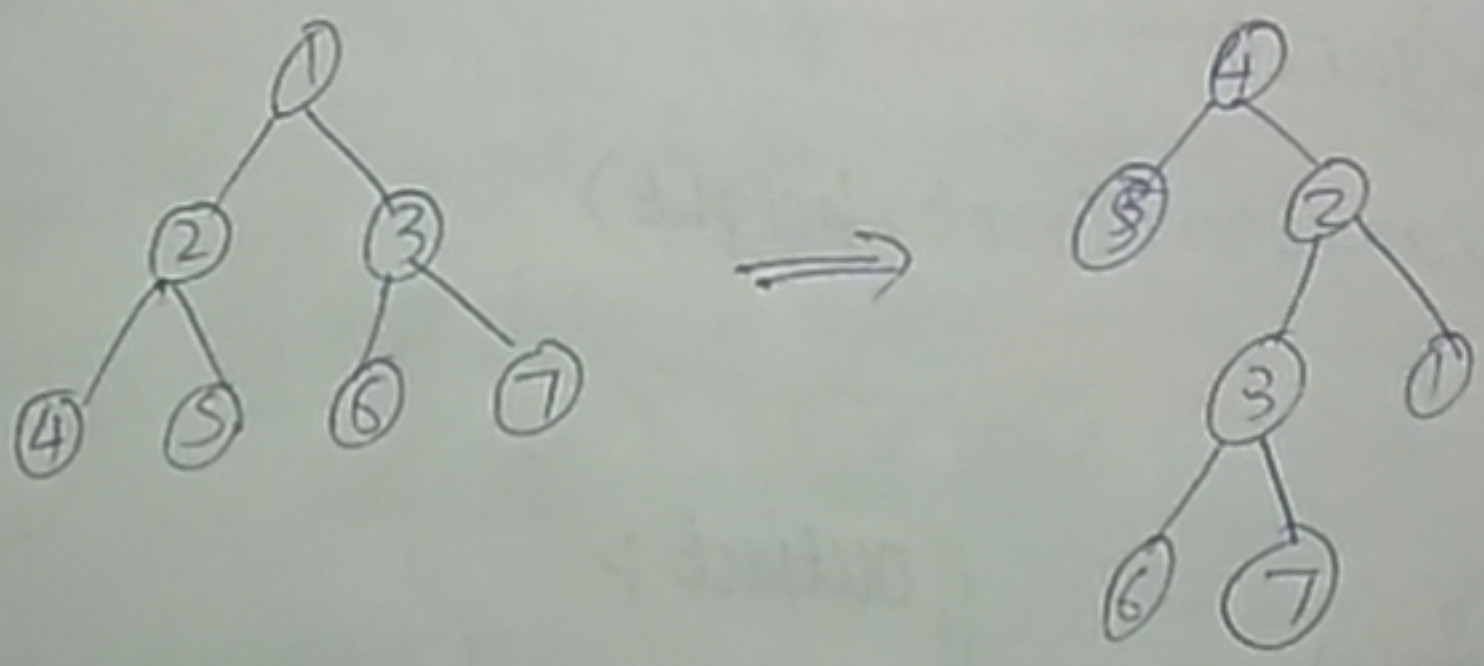
Day 22

Flip a binary tree

Given a binary tree, the task is to flip the binary tree towards right.

In a flip operation, left most node becomes the root of fliped tree and it parents becomes its right child and the right sibilings becomes left child and same should be done for all left most node recursively.



root → left → left = root → right

root → left → right = root

root → left = Null;

root → right = Null

```
class Node:

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

def flipbinarytree(root):
    if root is None:
        return root
    if root.left == None and root.right == None:
        return root

~~flipbinary~~ tree(
    flippedroot = flipbinarytree(root.left)
    root.left.left = root.right
    root.left.right = root
    root.left = root.right = None
    return flipped
```

```
def printlevelorder(root)
    if root is None:
        return
    from Queue import queue
    q = Queue()
    q.put(root)
    while(True):
        nodeCount = q.qsize()
        if nodecount == 0:
            break
        while node count > 0:
            node = q.get()
            print(node.data)
            if node.left is not None:
                q.put(node.left)
            if node.right is not None:
                q.put(node.right)
            node count -= 1
```

Input :

```
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.right.left = Node(4)
root.right.right = Node(5)
print("level order traversal of given tree")
printLevelOrder(root)
root = flipbinary tree(root)
print("level order traversal of fliped tree")
print levelorder(root)
```

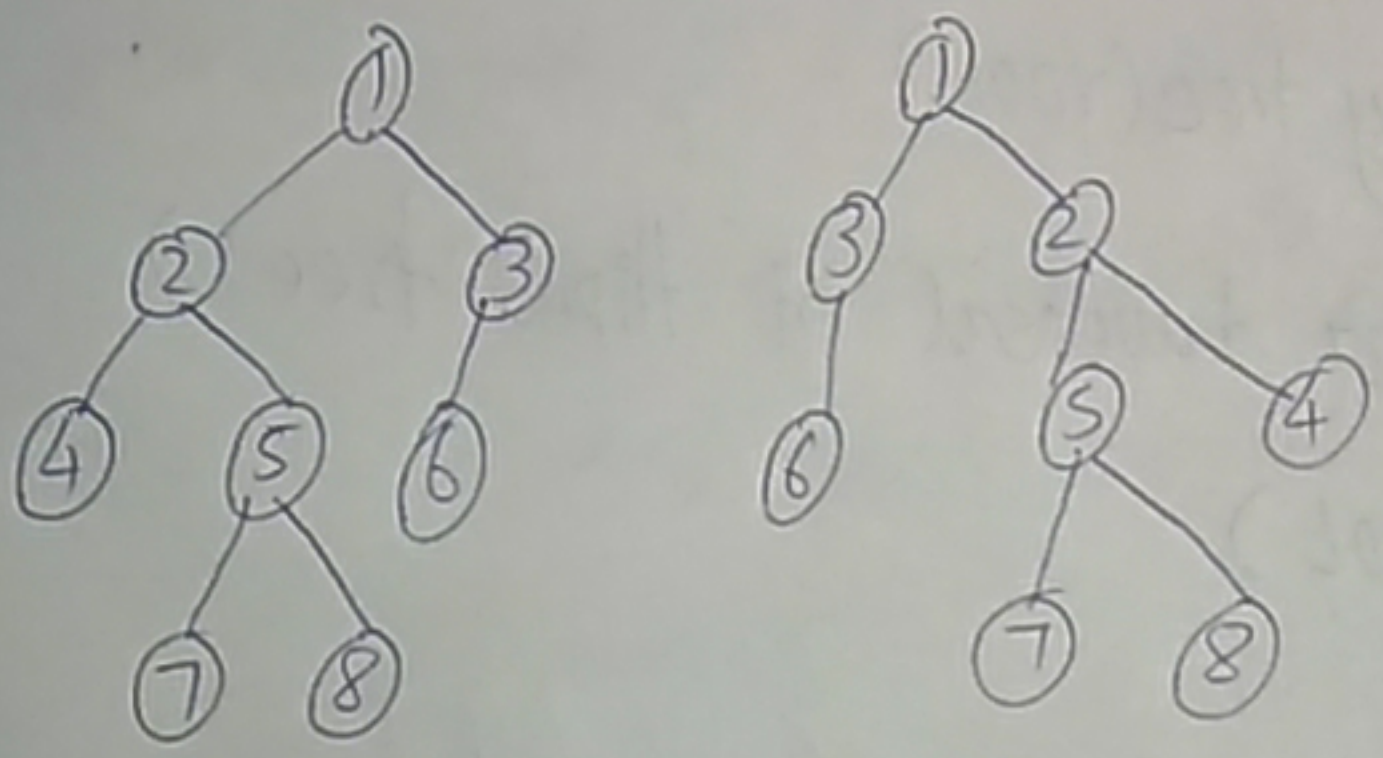
Output :

```
level order traversal of given tree
1
2 3
4 5
level order traversal of fliped tree
2
3 1
4 5
```

Tree Isomorphism Problem

~~Writ~~ If two trees are isomorpic. Two trees are called isomorpic if one of them can be obtained from other by a series of flips.

by swapping left and right children of a number of nodes.



Class Node:

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

```
def isomorphic(n1, n2):
    If n1 is None and n2 is None:
        return True
    If n1 is None or n2 is None:
        return False
    if n1.data != n2.data:
        return false
return ((isomorphic(n1.left, n2.right) and
        isomorphic(n1.right, n2.right)) or
        isomorphic(n1.left, n2.right) and
        isomorphic(n1.right, n2.left))
```