**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **01-07-2020** | | | | **Name:** | | | **Ashika** | |
| **Sem & Sec** | **6th  Sem ‘A’ Sec** | | | | **USN:** | | | **4AL17CS016** | |
| **Online Test Summary** | | | | | | | | | |
| **Subject** | | **-** | | | | | | | |
| **Max. Marks** | | **-** | | | | | **Score** | **-** | |
| **Pre-Placement Training Summary** | | | | | | | | | |
| **Course** | **-** | | | | | | | | |
| **Faculty** | | | **-** | | | **Duration** | | | **-** |
| **Coding Challenges** | | | | | | | | | |
| **Problem Statement:** 1. Write a program to find given two trees are mirror or not. | | | | | | | | | |
| **Status: done** | | | | | | | | | |
| **Uploaded the report in Github** | | | | **Yes** | | | | | |
| **If yes Repository name** | | | | Daily Report | | | | | |
| **Uploaded the report in slack** | | | | **Yes** | | | | | |

**Coding Challenge:**

1. Write a program to find given two trees are mirror or not.  
Description:  
Here are the steps to find out mirrored binary trees:

If both given trees root node values are same.  
Left subtree of root of first tree is mirror of right subtree of root of second tree.  
Right subtree of root of first tree is mirror of left subtree of root of second tree.

import java.util.\*;

class Main {

static class Node

{

int data;

Node left, right;

}

static Node newNode(int data)

{

Node temp = new Node();

temp.data = data;

temp.left = null;

temp.right = null;

return temp;

}

static String areMirrors(Node root1, Node root2)

{

Stack<Node> st1 = new Stack<Node> ();

Stack<Node> st2 = new Stack<Node> ();

while (true)

{

while (root1 != null && root2 != null)

{

if (root1.data != root2.data)

return "No";

st1.push(root1);

st2.push(root2);

root1 = root1.left;

root2 = root2.right;

}

if (!(root1 == null && root2 == null))

return "No";

if (!st1.isEmpty() && !st2.isEmpty())

{

root1 = st1.peek();

root2 = st2.peek();

st1.pop();

st2.pop();

root1 = root1.right;

root2 = root2.left;

}

else

break;

}

return "Yes";

}

public static void main(String[] args)

{

Node root1 = newNode(1);

root1.left = newNode(3);

root1.right = newNode(2);

root1.right.left = newNode(5);

root1.right.right = newNode(4);

Node root2 = newNode(1);

root2.left = newNode(2);

root2.right = newNode(3);

root2.left.left = newNode(4);

root2.left.right = newNode(5);

System.out.println(areMirrors(root1, root2));

}

}

**Output:**

