**1.Right Side View Of Binary Tree ( Input:** root = [1,2,3,null,5,null,4]

**Output:** [1,3,4]**)**

***BFS***

**LOGIC C#**

public IList<int> RightSideView(Node root)

{

List<int> rightSideValues = new List<int>();

Queue<Node> queue = new Queue<Node>();

queue.Enqueue(root);

while (queue.Count > 0)

{

int size = queue.Count;

for(int i = 0; i < size; i++)

{

var currentVal = queue.Dequeue();

if (i == 0)

rightSideValues.Add(currentVal.val);

if (currentVal.right != null)

queue.Enqueue(currentVal.right);

if (currentVal.left != null)

queue.Enqueue(currentVal.left);

}

}

return rightSideValues;

}

**LOGIC JS**

var rightSideView = function(root) {

let result = [];

let queue = [];

if(root == null)

return result;

queue.push(root);

while(queue.length>0){

let size = queue.length;

for(let i = 0 ; i<size; i++){

let currentVal = queue.shift();

if(i == 0){

result.push(currentVal.val);

}

if(currentVal.right)

queue.push(currentVal.right);

if(currentVal.left )

queue.push(currentVal.left);

}

}

return result;

};

**2. Subtree of another tree ( Input:** root = [3,4,5,1,2], subRoot = [4,1,2]

**Output:** true**)**

***DFS***

**LOGIC C#**

public bool IsSubtree(TreeNode root, TreeNode subRoot) {

if(root == null)

return false;

if(root.val == subRoot.val){

if(IsSame(root,subRoot)

return true;

}

Return IsSubtree(root.left,subRoot) || IsSubtree(root.right,subRoot);

}

private bool IsSame(TreeNode s, TreeNode t)

{

If(s == null && t == null)

Return true;

If(s!= null && t == null || s== null && t!= null || s.val != t.val)

Return false;

Return IsSame(s.left,t.left) &&IsSame(s.right,t.right);

}

**LOGIC JS**

var isSubtree = function(root, subRoot) {

if(root == null)

return false;

if(root.val == subRoot.val)

{

if(IsSame(root,subRoot))

return true;

}

return isSubtree(root.left,subRoot) || isSubtree(root.right,subRoot);

};

function IsSame(s,t){

if(s == null && t == null)

return true;

if(s!=null && t==null || t!=null && s==null || s.val != t.val)

return false;

return (IsSame(s.left, t.left) && IsSame(s.right, t.right));

}