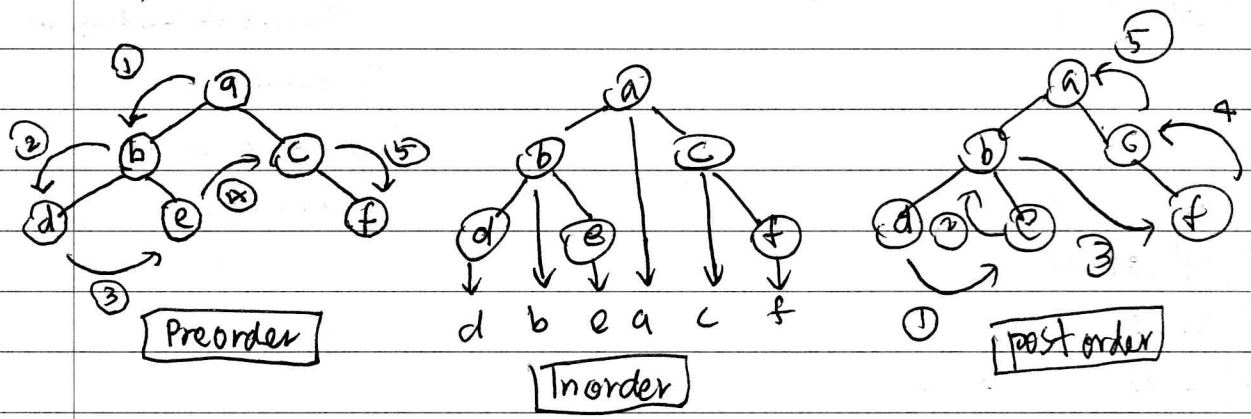
No.: Date:

Tut 6

Q1a) Preorder: a, b, d, e, c, f

b) Inorder: d, b, e, a, c, f

c) Postorder: d, e, b, f, c, a left right root



Q2) BSTinsert_recurs(root, temp) {

if (temp.data ≤ root.data) {

if (root.left == null)

root.left = temp

else BSTinsert_recurs (root.left, temp)

} else {

if (root.right == null)

root.right = temp

else BSTinsert_recurs (root.right, temp);

}

}

Q3) Algorithm LeafCounter(T);

if (T.Left != null && T.Right != null) {

Leaves = Leaf.counter(T.Left) + Leaf.counter(T.Right)

return Leaves

else

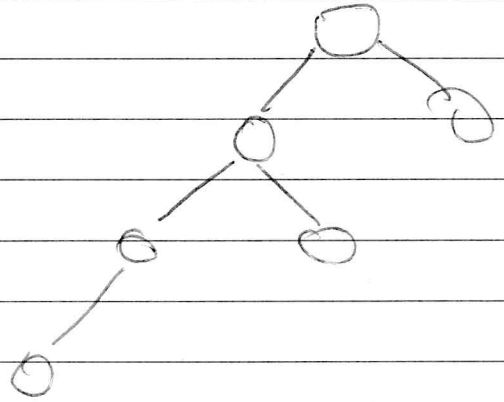
else if (T.right == null & T.left == null)

return 1.

No.: Date:

04 count_height(root)

if (root != null) {



height → number of ancestor

h = max(right height, Left height) + 1

height (T)

if T == null return -1

else

Leftheight (T.left)

right height (T.right)

if left height > right height

return left height + 1

else

return right height + 1

depth (v) = 1 + depth (parent(v))

05 // Return whether the tree T is empty

boolean isEmpty(T);

depth (T, v)

if Empty ( )

return -1

//

if is Root (v)

return 0

return 1 + depth (T, parameter)