14/10/2020

ADS LAB

Lab-5

Triveni.y

1BM19CS411

Write a program to perform insertion and deletion operations on AVL trees

Algorithm Insertion (Node * node, int Key)

// Insertion of a new node to the tree and balance the tree, if it is unbalance

{

if node = NULL

return newNode(Key)

if Key < node → Key

node → left = Insertion (node → left, Key)

elseif Key > node → Key

node → right = Insertion (node → right, Key)

else

return node

node → height = 1 + max (height (node → left), height (node → right))

// To check after inserting node whether tree is unbalanced.

if balance > 1 and key < node → left → key

}

Y. Triveni

Triveni.y
1BM19CS411

x → right = y
y → left = T2

y → height = max(height(y→left), height(y→right)) + 1

x → height = max(height(x→left), height(x→right)) + 1

end if

if balance < -1 and key > node → right → key

y → left = x
x → right = T2

x → height = max(height(x→left), height(x→right)) + 1

y → height = max(height(x→left), height(x→right)) + 1

end if

if balance > 1 and key > node → left → key

node → left = leftRotate(node → left)
return RightRotate(node)

end if

if balance < -1 and key < node → right → key

node → right = rightRotate(node → right)
return leftRotate(node)

y. Triveni

```
//delete

Node* delete (Node* node, int Key)

  if (node ==NULL)
      return root
  if   Key < node ->Key
      node ->left =delete (node ->left->Key)
else if  Key > node ->Key
    node ->right = delete(node->right->Key)

  else
     if  node ->left ==NULL and
        node-> right == NULL

       temp = node ->left ? node->left :
                  node->right
       if temp==NULL
           temp= node
           node =NULL
       else
           free (temp)
     end if
     else
        temp = minvaluenode (node->right)
        node ->Key  = temp ->Key
        node ->right = delete (node->right,
                          temp->Key)
```