14/10/2020

M. S. SATHWICK KIRAN
IBM18CS050
5-B

AVL TREE

```
Node * insert (Node * node, int key)
{
    if (Node == NULL)
    Return (newNode (key))
  if key = node → key
     node → left = insert (node → left, key);
  else if key > node → key
     node → right = insert (node → right, key);
  else
     return node;
  node → height = 1 + max( height (node → left), height
     int balance = get Balance (node);          (node→right));
  if balance < -1 && key > node → right → key
     return leftRotate (node);
  if balance > 1 && key < node → left → key
     return rightRotate (node);
  if balance > 1 && key > node → left → key
     node → left = left Rotate (node → left);
     return Right Rotate (node);
  if balance < -1 && key < node → right → key
     node → right = right Rotate (node → right);
     return left Rotate (node);
  return node;
}
```

```
Node * deleteNode (Node *root, int key)
{
  if root == NULL
    return root;
    if key < root -> key
      root -> left = deleteNode (root -> left, key);
    else if key > root -> key
      root -> right = deleteNode (root -> right, key)
  else
    {
      if root -> left == NULL || root -> right == NULL
        Node * temp = root -> left ?
                      root -> left :
                      root -> right;
  if temp == NULL
    temp = root;
    root = NULL;
  else
    root = * temp;
    free (temp);
  else
    Node * temp = minValueNode (root -> right);
    root -> key = temp -> key;
    root -> right = deleteNode (root -> right, temp -> key);
    if root == NULL
      return root;
      root -> height = 1 + max (height (root -> left),
                                height (root -> right));
  int balance = getBalance (root);
    if balance > 1 && getBalance (root -> left) >= 0)
    return RightRotate (root);
    if balance > 1 && getBalance (root -> left) < 0
      root -> left = LeftRotate (root -> left);
    return RightRotate
```

```
if balance < -1 && getBalance(root->right) <= 0
    return leftRotate(root);
if balance < -1 && getBalance(root->right) > 0
    root->right = rightRotate(root->right);
    return leftRotate(root);
return root;
}
```