# Name:Gaurav singh
# Roll.No:70
# PRACTICAL 6

**Create a BST and write a menu driven program to**:

1. Perform Pre order Traversal
2. Perform In order Traversal
3. Perform Post order Traversal
4. Count number of leaf nodes
5. Count number of non-leaf nodes
6. Count number of half-nodes

**Assignment:**

7. Find height of tree
8. Count number of nodes with ODD data in left sub tree of root
9. Check if the tree is Strictly Binary Tree or not.

***CODE:***

```
#include<stdio.h>
#include<stdlib.h>
struct bst{
struct bst *left;
struct bst *right;
int data;
};
struct bst *createNewNode(int data){
struct bst *ptr = (struct bst *)malloc(sizeof(struct bst));
ptr->data = data;
ptr->left = NULL;
ptr->right = NULL;
}
void preOrderTraversal(struct bst *root){
if(root!=NULL){
printf("%d ", root->data);
preOrderTraversal(root->left);
preOrderTraversal(root->right);
}
}
void postOrderTraversal(struct bst *root){
if(root != NULL){
postOrderTraversal(root->left);
postOrderTraversal(root->right);
printf("%d ", root->data);
}
}
void inOrderTraversal(struct bst *root){
if(root != NULL){
inOrderTraversal(root->left);
printf("%d ", root->data);
inOrderTraversal(root->right);
}
```

```
}
struct bst *createTree(struct bst *root){
int num, i;
int a[] = {8,3,10,14,13,6,7,4,1};
struct bst *ptr, *temp, *prev;
root = NULL;
for(i=0; i<8; i++){
temp = (struct bst *)malloc(sizeof(struct bst));
temp->data = a[i];
temp->left = NULL;
temp->right = NULL;
if(root == NULL){
root = temp;
}
else{
ptr = root;
while(ptr!=NULL){
prev = ptr;
if(ptr->data<temp->data){
ptr = ptr->right;
}
else{
ptr = ptr->left;
}
}
if(prev->data <temp->data){
prev->right = temp;
}
else{
prev->left = temp;
}
}
}
return root;
}
int count=0;
int LeafNodes(struct bst* root)
{
if(root != NULL)
{
LeafNodes(root->left);
if((root->left == NULL) && (root->right == NULL))
{
count++;
}
LeafNodes(root->right);
}
return count;
}
int count2=0;
int nonLeafNodes(struct bst *root){
if(root!=NULL){
nonLeafNodes(root->left);
if((root->left!=NULL)||(root->right!=NULL)){
count2++;
}
nonLeafNodes(root->right);
}
return count2;
```

```
}
int halfNode(struct bst* root){
int count3=0;
if (root == NULL)
return 0;
if ((root->left == NULL && root->right != NULL) || (root->left
!= NULL && root->right == NULL)){
count3++;
}
count3 += (halfNode(root->left) + halfNode(root->right));
return count3;
}
int count4=0;
int maxHeight(struct bst *root)
{
if (root == NULL)
return -1;
else
{
int lheight = maxHeight(root->left);
int rheight = maxHeight(root->right);
if (lheight > rheight)
return(lheight + 1);
else return(rheight + 1);
}
}
int isBST(struct bst* root)
{
if (root == NULL)
return 1;
if (root->left != NULL && root->left->data > root->data)
return 0;
if (root->right != NULL && root->right->data < root->data)
return 0;
if (!isBST(root->left) || !isBST(root->right))
return 0;
return 1;
}
int main()
{
struct bst *root;
root = createTree(root);
char ch='y';
int choice;
while(ch!='n'){
printf("Enter the Choice : \n");
printf("Enter 1 for Pre-Order Traversal.\n");
printf("Enter 2 for Post-Order Traversal.\n");
printf("Enter 3 for In-Order Traversal.\n");
printf("Enter 4 for Counting Leaf-Nodes.\n");
printf("Enter 5 for Counting Non-Leaf-Nodes.\n");
printf("Enter 6 for Counting Half-Nodes.\n");
printf("Enter 7 for checking if the given tree is Binary-SearchTree or Not.");
scanf("%d", &choice);
switch(choice){
case 1:{
printf("\nPre Order Traversal of given tree is : ");
preOrderTraversal(root);
}
```

```
break;
case 2:{
printf("\nPost Order Traversal of given tree is : ");
postOrderTraversal(root);
}
break;
case 3:{
printf("\nIn Order Traversal of given tree is : ");
inOrderTraversal(root);
}
break;
case 4:{
int n = LeafNodes(root);
printf("Leaf Nodes in the given tree is %d\n", n);
}
break;
case 5:{
int n = nonLeafNodes(root);
printf("Non Leaf Nodes in the given tree is %d\n", n);
}
break;
case 6:{
int n = halfNode(root);
printf("Half Leaf Nodes in the given tree is %d\n", n);
}
break;
case 7:{
int n = isBST(root);
if(n==1){
printf("The given tree is Binary Search Tree.\n");
}
else{
printf("The given tree is not a Binary Search.\n");
}
}
break;
default:{
printf("You Entered a wrong choice.\n");
}
break;
}
printf("\nDo you want to continue? Press y for yes and n for no.\n");
scanf(" %c", &ch);
}
return 0;
}
```

Output

```
/tmp/LTjHDgCfHl.o
Enter the Choice :
Enter 1 for Pre-Order Traversal.
Enter 2 for Post-Order Traversal.
Enter 3 for In-Order Traversal.
Enter 4 for Counting Leaf-Nodes.
Enter 5 for Counting Non-Leaf-Nodes.
Enter 6 for Counting Half-Nodes.
Enter 7 for checking if the given tree is Binary-SearchTree or Not.1
Pre Order Traversal of given tree is : 8 3 6 4 7 10 14 13
```

```
Enter the Choice :
Enter 1 for Pre-Order Traversal.
Enter 2 for Post-Order Traversal.
Enter 3 for In-Order Traversal.
Enter 4 for Counting Leaf-Nodes.
Enter 5 for Counting Non-Leaf-Nodes.
Enter 6 for Counting Half-Nodes.
Enter 7 for checking if the given tree is Binary-SearchTree or Not.2
Post Order Traversal of given tree is : 4 7 6 3 13 14 10 8
```

```
Enter the Choice :
Enter 1 for Pre-Order Traversal.
Enter 2 for Post-Order Traversal.
Enter 3 for In-Order Traversal.
Enter 4 for Counting Leaf-Nodes.
Enter 5 for Counting Non-Leaf-Nodes.
Enter 6 for Counting Half-Nodes.
Enter 7 for checking if the given tree is Binary-SearchTree or Not.3
In Order Traversal of given tree is : 3 4 6 7 8 10 13 14
Do you want to continue? Press y for yes and n for no.
y
```

```
Enter the Choice :
Enter 1 for Pre-Order Traversal.
Enter 2 for Post-Order Traversal.
Enter 3 for In-Order Traversal.
Enter 4 for Counting Leaf-Nodes.
Enter 5 for Counting Non-Leaf-Nodes.
Enter 6 for Counting Half-Nodes.
Enter 7 for checking if the given tree is Binary-SearchTree or Not.4
Leaf Nodes in the given tree is 3
```

```
Enter the Choice :
Enter 1 for Pre-Order Traversal.
Enter 2 for Post-Order Traversal.
Enter 3 for In-Order Traversal.
Enter 4 for Counting Leaf-Nodes.
Enter 5 for Counting Non-Leaf-Nodes.
Enter 6 for Counting Half-Nodes.
Enter 7 for checking if the given tree is Binary-SearchTree or Not.5
Non Leaf Nodes in the given tree is 5

Do you want to continue? Press y for yes and n for no.
```

```
Enter the Choice :
Enter 1 for Pre-Order Traversal.
Enter 2 for Post-Order Traversal.
Enter 3 for In-Order Traversal.
Enter 4 for Counting Leaf-Nodes.
Enter 5 for Counting Non-Leaf-Nodes.
Enter 6 for Counting Half-Nodes.
Enter 7 for checking if the given tree is Binary-SearchTree or Not.6
Half Leaf Nodes in the given tree is 3
```

```
Enter the Choice :
Enter 1 for Pre-Order Traversal.
Enter 2 for Post-Order Traversal.
Enter 3 for In-Order Traversal.
Enter 4 for Counting Leaf-Nodes.
Enter 5 for Counting Non-Leaf-Nodes.
Enter 6 for Counting Half-Nodes.
Enter 7 for checking if the given tree is Binary-SearchTree or Not.7
The given tree is Binary Search Tree.
```