**Assignment 5**

**Name :-** Dhruv Thakkar

Branch :- CS

Division :- A

Roll no. :- 69

GR Number :- 12111512

**Problem Statement**

Write a Program to create a Binary Search Tree and perform

following nonrecursive operations on it. a. Preorder Traversal b.

Inorder Traversal c. Display Number of Leaf Nodes d. Mirror

Image

**Solution**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

struct Node {

    int data;

    struct Node\* left;

    struct Node\* right;

};

struct Node\* insertNode(struct Node\* root, int data) {

   struct Node \*new = (struct Node \*)(malloc(sizeof(struct Node)));

    if (root == NULL)

    {

        new->data = data;

        new->left = NULL;

        new->right = NULL;

        root = new;

        return root;

    }

    else

    {

        struct Node \*temp = root;

        while (1)

        {

            if (temp->data > data)

            {

                if (temp->left != NULL)

                {

                    temp = temp->left;

                }

                else

                {

                    new->data = data;

                    new->left = NULL;

                    new->right = NULL;

                    temp->left = new;

                    return root;

                }

            }

            else

            {

                if (temp->right != NULL)

                {

                    temp = temp->right;

                }

                else

                {

                    new->data = data;

                    new->left = NULL;

                    new->right = NULL;

                    temp->right = new;

                    return root;

                }

            }

        }

    }

}

void preorderTraversal(struct Node\* root) {

    if (root == NULL){

        printf("Tree is empty!\n");

        return;

    }

    struct Node\* stack[100];

    int top = -1;

    stack[++top] = root;

    while (top >= 0) {

        struct Node\* currentNode = stack[top--];

        printf("%d ", currentNode->data);

        if (currentNode->right != NULL){

            stack[++top] = currentNode->right;

        }

        if (currentNode->left != NULL){

            stack[++top] = currentNode->left;

        }

    }

}

void inorderTraversal(struct Node\* root) {

    if (root == NULL){

        return;

    }

    struct Node\* stack[100];

    int top = -1;

    struct Node\* currentNode = root;

    while (currentNode != NULL || top >= 0) {

        while (currentNode != NULL) {

            stack[++top] = currentNode;

            currentNode = currentNode->left;

        }

        currentNode = stack[top--];

        printf("%d ", currentNode->data);

        currentNode = currentNode->right;

    }

}

int countLeafNodes(struct Node\* root) {

    if (root == NULL){

        return 0;

    }

    struct Node\* queue[100];

    int front = 0;

    int rear = -1;

    queue[++rear] = root;

    int count = 0;

    while (front <= rear) {

        struct Node\* currentNode = queue[front++];

        if (currentNode->left == NULL && currentNode->right == NULL){

            count++;

        }

        if (currentNode->left != NULL){

            queue[++rear] = currentNode->left;

        }

        if (currentNode->right != NULL){

            queue[++rear] = currentNode->right;

        }

    }

    return count;

}

void mirrorImage(struct Node\* root) {

    if (root == NULL){

        return;

    }

    struct Node\* queue[100];

    int front = 0;

    int rear = -1;

    queue[++rear] = root;

    while (front <= rear) {

        struct Node\* currentNode = queue[front++];

        struct Node\* temp = currentNode->left;

        currentNode->left = currentNode->right;

        currentNode->right = temp;

        if (currentNode->left != NULL){

            queue[++rear] = currentNode->left;

        }

        if (currentNode->right != NULL){

            queue[++rear] = currentNode->right;

        }

    }

}

int main() {

    struct Node\* root = NULL;

    int n;

    printf("Enter the number of nodes in the BST:");

    scanf("%d", &n);

    printf("Enter the data of nodes:");

    for (int i = 0; i < n; i++)

    {

        int data;

        scanf("%d", &data);

        root = insertNode(root, data);

    }

    printf("Preorder Traversal: \n");

    preorderTraversal(root);

    printf("\n");

    printf("Inorder Traversal\n");

    inorderTraversal(root);

    printf("\n");

    printf("Number of Leaf Nodes:  %d\n",countLeafNodes(root));

    printf("\n");

    printf("Mirror Image Done\n");

    mirrorImage(root);

    printf("\n");

    printf("Preorder Traversal: \n");

    preorderTraversal(root);

    printf("\n");

    printf("Inorder Traversal\n");

    inorderTraversal(root);

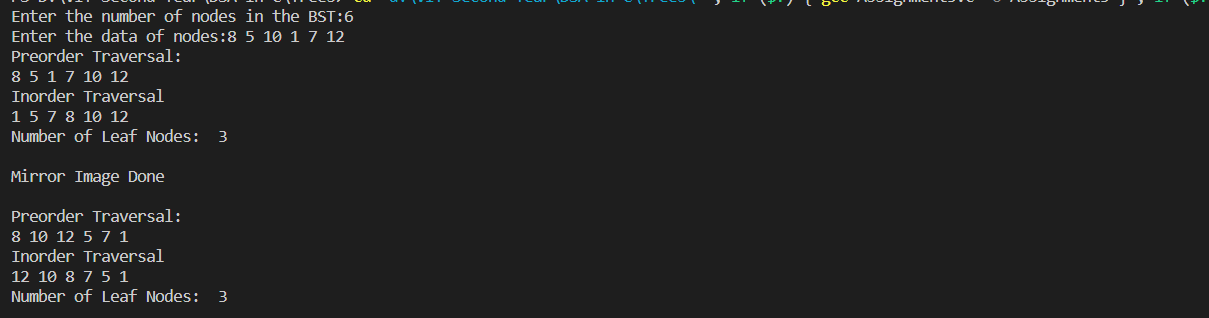
    printf("\n");

    printf("Number of Leaf Nodes:  %d\n",countLeafNodes(root));

    printf("\n");

}

**OUTPUT**

****