**Ex 8: Tree Traversal**

**PROGRAM :**

#include <stdio.h>

#include <stdlib.h>

struct node {

int element;

struct node\* left;

struct node\* right;

};

struct node\* createNode(int val)

{

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

Node->element = val;

Node->left = NULL;

Node->right = NULL;

return (Node);

}

void traversePreorder(struct node\* root)

{

if (root == NULL)

return;

printf(" %d ", root->element);

traversePreorder(root->left);

traversePreorder(root->right);

}

void traverseInorder(struct node\* root)

{

if (root == NULL)

return;

traverseInorder(root->left);

printf(" %d ", root->element);

traverseInorder(root->right);

}

void traversePostorder(struct node\* root)

{

if (root == NULL)

return;

traversePostorder(root->left);

traversePostorder(root->right);

printf(" %d ", root->element);

}

int main()

{

struct node\* root = createNode(36);

root->left = createNode(26);

root->right = createNode(46);

root->left->left = createNode(21);

root->left->right = createNode(31);

root->left->left->left = createNode(11);

root->left->left->right = createNode(24);

root->right->left = createNode(41);

root->right->right = createNode(56);

root->right->right->left = createNode(51);

root->right->right->right = createNode(66);

printf("\n The Preorder traversal of given binary tree is -\n");

traversePreorder(root);

printf("\n The Inorder traversal of given binary tree is -\n");

traverseInorder(root);

printf("\n The Postorder traversal of given binary tree is -\n");

traversePostorder(root);

return 0;

}

**OUTPUT:**

