#include <stdio.h>

#include <stdlib.h>

Struct node {

Int item;

Struct node\* left;

Struct node\* right;

};

// Inorder traversal

Void inorderTraversal(struct node\* root) {

If (root == NULL) return;

inorderTraversal(root->left);

printf(“%d ->”, root->item);

inorderTraversal(root->right);

}

// Preorder traversal

Void preorderTraversal(struct node\* root) {

If (root == NULL) return;

Printf(“%d ->”, root->item);

preorderTraversal(root->left);

preorderTraversal(root->right);

}

// Postorder traversal

Void postorderTraversal(struct node\* root) {

If (root == NULL) return;

postorderTraversal(root->left);

postorderTraversal(root->right);

printf(“%d ->”, root->item);

}

// Create a new Node

Struct node\* createNode(value) {

Struct node\* newNode = malloc(sizeof(struct node));

newNode->item = value;

newNode->left = NULL;

newNode->right = NULL;

return newNode;

}

// Insert on the left of the node

Struct node\* insertLeft(struct node\* root, int value) {

Root->left = createNode(value);

Return root->left;

}

// Insert on the right of the node

Struct node\* insertRight(struct node\* root, int value) {

Root->right = createNode(value);

Return root->right;

}

Int main() {

Struct node\* root = createNode(1);

insertLeft(root, 2);

insertRight(root, 3);

insertLeft(root->left, 4);

printf(“Inorder traversal \n”);

inorderTraversal(root);

printf(“\nPreorder traversal \n”);

preorderTraversal(root);

printf(“\nPostorder traversal \n”);

postorderTraversal(root);

}