#include <stdio.h>

#include <stdlib.h>

// Define the structure for a node in the binary search tree

struct Node {

int data;

struct Node\* left;

struct Node\* right;

};

// Function to create a new node with given data

struct Node\* createNode(int data) {

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

newNode->data = data;

newNode->left = NULL;

newNode->right = NULL;

return newNode;

}

// Function to insert a new node with given data in the BST

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

if (root == NULL) {

return createNode(data);

}

if (data < root->data) {

root->left = insert(root->left, data);

} else if (data > root->data) {

root->right = insert(root->right, data);

}

return root;

}

// Function to perform inorder traversal of the BST

void inorderTraversal(struct Node\* root) {

if (root != NULL) {

inorderTraversal(root->left);

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

inorderTraversal(root->right);

}

}

// Function to perform preorder traversal of the BST

void preorderTraversal(struct Node\* root) {

if (root != NULL) {

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

preorderTraversal(root->left);

preorderTraversal(root->right);

}

}

// Function to perform postorder traversal of the BST

void postorderTraversal(struct Node\* root) {

if (root != NULL) {

postorderTraversal(root->left);

postorderTraversal(root->right);

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

}

}

// Function to search for a node with given data in the BST

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

if (root == NULL || root->data == data) {

return root;

}

if (data < root->data) {

return search(root->left, data);

} else {

return search(root->right, data);

}

}

// Main function to demonstrate the BST operations

int main() {

struct Node\* root = NULL;

root = insert(root, 50);

insert(root, 30);

insert(root, 20);

insert(root, 40);

insert(root, 70);

insert(root, 60);

insert(root, 80);

printf("Inorder traversal of the binary search tree:\n");

inorderTraversal(root);

printf("\n");

printf("Preorder traversal of the binary search tree:\n");

preorderTraversal(root);

printf("\n");

printf("Postorder traversal of the binary search tree:\n");

postorderTraversal(root);

printf("\n");

int key = 40;

if (search(root, key) != NULL) {

printf("Node with data %d found in the BST.\n", key);

} else {

printf("Node with data %d not found in the BST.\n", key);

}

return 0;

}

