Experiment No.9: Binary Search Tree Operations
An online directory system uses a BST to keep names in a sorted manner and support fast searching.Create a binary search tree and implement recursive traversals (inorder, preorder, postorder) and search for a specific name in the directory.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Node {
    char name[50];
    struct Node* left;
    struct Node* right;
};

// Create a new node
struct Node* createNode(char name[]) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    strcpy(newNode->name, name);
    newNode->left = newNode->right = NULL;
    return newNode;
}

// Insert into BST
struct Node* insert(struct Node* root, char name[]) {
    if (root == NULL) return createNode(name);

    if (strcmp(name, root->name) < 0)
        root->left = insert(root->left, name);
    else if (strcmp(name, root->name) > 0)
        root->right = insert(root->right, name);

    return root;
}

// Search name
int search(struct Node* root, char name[]) {
    if (root == NULL) return 0;
    if (strcmp(name, root->name) == 0) return 1;
    else if (strcmp(name, root->name) < 0)
        return search(root->left, name);
    else
        return search(root->right, name);
}

// Traversals
void inorder(struct Node* root) {
    if (root) {
        inorder(root->left);
```

```
        printf("%s ", root->name);
        inorder(root->right);
    }
}
void preorder(struct Node* root) {
    if (root) {
        printf("%s ", root->name);
        preorder(root->left);
        preorder(root->right);
    }
}
void postorder(struct Node* root) {
    if (root) {
        postorder(root->left);
        postorder(root->right);
        printf("%s ", root->name);
    }
}

int main() {
    struct Node* root = NULL;
    int n;
    char name[50];

    printf("Enter number of names to insert: ");
    scanf("%d", &n);
    getchar(); // clear newline

    for (int i = 0; i < n; i++) {
        printf("Enter name %d: ", i + 1);
        gets(name);  // For simplicity; can use fgets too
        root = insert(root, name);
    }

    printf("\nInorder: ");
    inorder(root);
    printf("\nPreorder: ");
    preorder(root);
    printf("\nPostorder: ");
    postorder(root);

    printf("\n\nEnter name to search: ");
    gets(name);
    if (search(root, name))
        printf("Name found in directory.\n");
    else
        printf("Name not found.\n");

    return 0;
}
```