Rajalakshmi Engineering College

Name: kamali rj
Email: 240701225@rajalakshmi.edu.in
Roll no: 240701225
Phone: 9344843996
Branch: REC
Department: I CSE AH
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are required to implement basic operations on a Binary Search Tree (BST), like insertion and searching.

Insertion: Given a list of integers, construct a Binary Search Tree by repeatedly inserting each integer into the tree according to the rules of a BST.

Searching: Given an integer, search for its presence in the constructed Binary Search Tree. Print whether the integer is found or not.

Write a program to calculate this efficiently.

Input Format

The first line of input consists of an integer n, representing the number of nodes in the binary search tree.

The second line consists of the values of the nodes, separated by space as integers.

The third line consists of an integer representing, the value that is to be searched.

Output Format

The output prints, "Value <value> is found in the tree." if the given value is present, otherwise it prints: "Value <value> is not found in the tree."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
8 3 10 1 6 14 23
6
Output: Value 6 is found in the tree.

Answer

```
// You are using GCC
#include <stdio.h>
#include <stdlib.h>

// 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
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;
}
```

240701225

```
// Function to insert a node into 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 search for a value in the BST
struct Node* search(struct Node* root, int key) {
    if (root == NULL || root->data == key) {
        return root;
    }

    if (key < root->data) {
        return search(root->left, key);
    }

    return search(root->right, key);
}

int main() {
    int n, valueToSearch;

    // Read the number of nodes
    scanf("%d", &n);

    // Read the node values and construct the BST
    struct Node* root = NULL;
    for (int i = 0; i < n; i++) {
        int nodeValue;
        scanf("%d", &nodeValue);
        root = insert(root, nodeValue);
    }
```

240701225

```
    // Read the value to be searched
    scanf("%d", &valueToSearch);

    // Search for the value
    if (search(root, valueToSearch) != NULL) {
        printf("Value %d is found in the tree.\n", valueToSearch);
    } else {
        printf("Value %d is not found in the tree.\n", valueToSearch);
    }

    return 0;
}
```

Status : Correct ***Marks : 10/10***

240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225 240701225