Rajalakshmi Engineering College

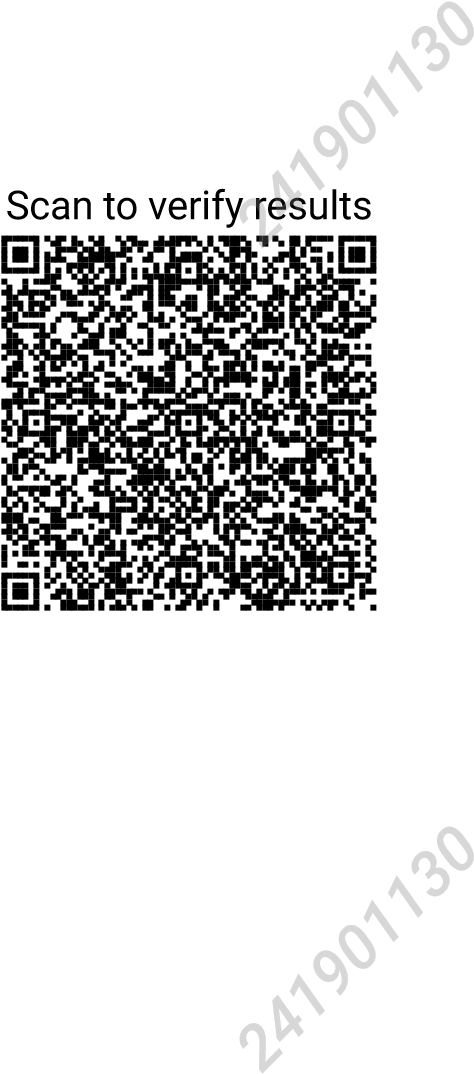
*241901092*

*241901092*

Name: H.Sadhaa sivam

*241901092*

*241901092*

Email: 241901092@rajalakshmi.edu.in

Roll no: 241901092

Phone: 6383796684

Branch: REC

Department: l CSE (CS) FB

Batch: 2028

Degree: B.E - CSE (CS)

NeoColab\_REC\_CS23231\_DATA STRUCTURES

*241901092*

*241901092*

REC\_DS using C\_Week 5\_COD\_Question 1

Attempt : 1

Total Mark : 10

Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

John is learning about Binary Search Trees (BST) in his computer science class. He wants to create a program that allows users to delete a node with a given value from a BST and print the remaining nodes using an inorder traversal.

*241901092*

*241901092*

*241901092*

*241901092*

Implement a function to help him delete a node with a given value from a BST.

*Input Format*

The first line of input consists of an integer N, representing the number of nodes in the BST.

*241901092*

*241901092*

*241901092*

*241901092*

The second line consists of N space-separated integers, representing the values of the BST nodes.

The third line consists of an integer V, which is the value to delete from the BST.

*241901092*

*Output Format*

The output prints the space-separated values in the BST in an in-order traversal, after the deletion of the specified value.

*241901092*

*241901092*

*241901092*

If the specified value is not available in the tree, print the given input values inorder traversal.

Refer to the sample output for formatting specifications.

Input: 5

10

5 15 2

7

15

*241901092*

*241901092*

*241901092*

*241901092*

*Sample Test Case*

Output: 2 5 7 10

*Answer*

#include <stdio.h>

#include <stdlib.h>

struct TreeNode {

*241901092*

int data;

struct TreeNode\* left;

*241901092*

*241901092*

*241901092*

struct TreeNode\* right;

};

struct TreeNode\* createNode(int key) {

struct TreeNode\* newNode = (struct TreeNode\*)malloc(sizeof(struct TreeNode)); newNode->data = key;

newNode->left = newNode->right = NULL; return newNode;

struct TreeNode\* insert(struct TreeNode\* root, int key) { if (root == NULL) {

}

*241901092*

*241901092*

*241901092*

*241901092*

return createNode(key);

*241901092*

*241901092*

}

if (key < root->data)

root->left = insert(root->left, key); else if (key > root->data) root->right = insert(root->right, key); return root;

*241901092*

*241901092*

}

struct TreeNode\* findMin(struct TreeNode\* root) { while (root && root->left != NULL) root = root->left; return root;

}

struct TreeNode\* deleteNode(struct TreeNode\* root, int key) { if (root == NULL) return NULL;

*241901092*

*241901092*

*241901092*

*241901092*

if (key < root->data)

root->left = deleteNode(root->left, key); else if (key > root->data)

root->right = deleteNode(root->right, key); else {

if (root->left == NULL) { struct TreeNode\* temp = root->right; free(root); return temp;

*241901092*

*241901092*

*241901092*

*241901092*

} else if (root->right == NULL) {

struct TreeNode\* temp = root->left; free(root); return temp;

} else {

struct TreeNode\* temp = findMin(root->right); root->data = temp->data;

root->right = deleteNode(root->right, temp->data); void inorderTraversal(struct TreeNode\* root) { if (root != NULL) {

*241901092*

*241901092*

*241901092*

*241901092*

*241901092*

}

}

return root;

}

*241901092*

*241901092*

*241901092*

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

inorderTraversal(root->right);

}

}

int main()

{

int N, rootValue, V; scanf("%d", &N);

*241901092*

*241901092*

struct TreeNode\* root = NULL;

for (int i = 0; i < N; i++) { int key; scanf("%d", &key); if (i == 0) rootValue = key;

*241901092*

*241901092*

root = insert(root, key);

}

scanf("%d", &V); root = deleteNode(root, V); inorderTraversal(root); return 0;

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

}

*241901092*

*241901092*

*241901092*

*241901092*

*241901092*

*241901092*

*241901092*

*241901092*