231501058 CS23231 – D a t a S t r u c t u r e s

**Ex. No.: 9**

**Implementation of Binary Search tree**

**Date: 17/5/24**

**Write a C prog ram to imple ment a Binary Search Tree and perform the followin g ope ration s .**

1. **Insert**
2. **Delete**
3. **Search**
4. **Display**

**Algorithm:**

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node\* left;

struct node\* right;

};

struct node\* createNode(int data) {

struct node\* newNode = (struct node\*)malloc(sizeof(struct node)); if (newNode != NULL) {

newNode->data = data;

newNode->left = NULL;

newNode->right = NULL;

}

return newNode;

}

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;

}

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);

}

return search(root->right, data);

}



**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 34

231501058 CS23231 – D a t a S t r u c t u r e s

struct node\* findMin(struct node\* root) {

while (root->left != NULL) {

root = root->left;

}

return root;

}

struct node\* deleteNode(struct node\* root, int data) { if (root == NULL) {

return root;

}

if (data < root->data) {

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

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

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

} else {

if (root->left == NULL) {

struct node\* temp = root->right;

free(root);

return temp;

} else if (root->right == NULL) { struct node\* temp = root->left; free(root);

return temp;

}

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

root->data = temp->data;

root->right = deleteNode(root->right, temp->data);

}

return root;

}

void inorder(struct node\* root) {

if (root != NULL) {

inorder(root->left);

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

inorder(root->right);

}

}

void preorder(struct node\* root) {

if (root != NULL) {

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

preorder(root->left);

preorder(root->right);

}

}

void postorder(struct node\* root) {

if (root != NULL) {

postorder(root->left);

postorder(root->right);

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

}



**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 35

231501058 CS23231 – D a t a S t r u c t u r e s

}

int main() {

struct node\* root = NULL;

root = insert(root, 50);

insert(root, 30);

insert(root, 70);

insert(root, 20);

insert(root, 40);

insert(root, 60);

insert(root, 80);

printf("Inorder traversal: ");

inorder(root);

printf("\n");

printf("Preorder traversal: ");

preorder(root);

printf("\n");

printf("Postorder traversal: ");

postorder(root);

printf("\n");

int key = 40;

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

printf("Element %d found in the BST\n", key);

} else {

printf("Element %d not found in the BST\n", key);

}

printf("Deleting 20\n");

root = deleteNode(root, 20);

printf("Inorder traversal after deletion: ");

inorder(root);

printf("\n");

printf("Deleting 30\n");

root = deleteNode(root, 30);

printf("Inorder traversal after deletion: ");

inorder(root);

printf("\n");

printf("Deleting 50\n");

root = deleteNode(root, 50);

printf("Inorder traversal after deletion: ");

inorder(root);

printf("\n");

return 0;

}

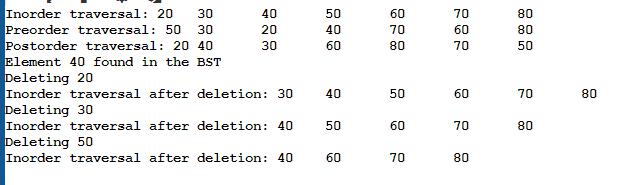


**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 36

231501058 CS23231 – D a t a S t r u c t u r e s

**OUTPUT**



**Dept of Artificial Intelligence and Machine Learning** | **Rajalakshmi Engineering College**

. 37