LAB-10

Name: K V Jaya Harsha

Roll no: CS23B1034 Date: 09-10-2024

Q1. Binary Search Tree with in-order traversal (in cpp)

```cpp
// CS23B1034
// K V Jaya Harsha
#include <iostream>
using namespace std;
struct Node{
    int data;
    Node *left;
    Node *right;
};
struct Node *nn(int item){
    struct Node *temp = new Node;
    temp->data = item;
    temp->left = NULL;
    temp->right = NULL;
    return temp;
};
struct Node *insert(struct Node *&node, int val){
    if (node == NULL){
        return nn(val);
    }
    if (val < node->data){
        node->left = insert(node->left, val);
    }
    if (val > node->data){
        node->right = insert(node->right, val);
    }
    return node;
};
void inorderTraversal(Node *root){
    if (root != NULL){
        inorderTraversal(root->left);
        cout << root->data << " ";
        inorderTraversal(root->right);
    }
}
int main(){
    struct Node *root = NULL;
    root = insert(root, 54);
    insert(root, 76);
    insert(root, 64);
    insert(root, 91);
    insert(root, 32);
    insert(root, 47);
    insert(root, 72);
    insert(root, 21);
    insert(root, 9);
    cout << "Displaying BST: ";
    inorderTraversal(root);
    cout << endl;
    return 0;
}
```

Lk6

```
PS C:\Users\harsh> cd "c:\Users\harsh\OneDrive\Doc
-inorder-traversal.cpp -o bst-inorder-traversa
Displaying BST: 9 21 32 47 54 64 72 76 91
PS C:\Users\harsh\OneDrive\Documents\Desktop\c
```

Q3. Delete a node

```cpp
// CS23B1034
// K V Jaya Harsha
#include <iostream>
using namespace std;

struct Node{
    int data;
    Node *left;
    Node *right;
};

struct Node *nn(int item){
    struct Node *temp = new Node;
    temp->data = item;
    temp->left = NULL;
    temp->right = NULL;
    return temp;
}

struct Node *insert(struct Node *&node, int val){
    if (node == NULL){
        return nn(val);
    }
    if (val < node->data){
        node->left = insert(node->left, val);
    }
    else if (val > node->data){
        node->right = insert(node->right, val);
    }
    return node;
}

void inorderTraversal(Node *root){
    if (root != NULL){
        inorderTraversal(root->left);
        cout << root->data << " ";
        inorderTraversal(root->right);
    }
}

Node *minValueNode(Node *node){
    Node *current = node;
    while (current && current->left != NULL){
        current = current->left;
    }
    return current;
}
```

```cpp
Node *deletenode(Node *&root, int key){
    if (root == NULL)
        return root;

    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){
            Node *temp = root->right;
            delete root;
            return temp;
        }
        else if (root->right == NULL){
            Node *temp = root->left;
            delete root;
            return temp;
        }
        Node *temp = minValueNode(root->right);
        root->data = temp->data;
        root->right = deletenode(root->right, temp->data);
    }
    return root;
}
int main(){
    struct Node *root = NULL;
    root = insert(root, 54);
    insert(root, 76);
    insert(root, 64);
    insert(root, 91);
    insert(root, 32);
    insert(root, 47);
    insert(root, 72);
    insert(root, 21);
    insert(root, 9);
    cout << "BST before deletion: ";
    inorderTraversal(root);
    cout << endl;
    root = deletenode(root, 64);
    cout << "BST after deleting 64: ";
    inorderTraversal(root);
    cout << endl;
    return 0;
}
```

```
ing-node } ; if ($?) { .\deleting-node }
BST before deletion: 9 21 32 47 54 64 72 76 91
BST after deleting 64: 9 21 32 47 54 72 76 91
PS C:\Users\harsh\OneDrive\Documents\Desktop\chall
```