EXPERIMENT – 3.2

Name: Himanshu Raj
UID: 21BCS9318
Branch: CSE
Section/Group: **902** (A)
Semester: 3rd
Date of Performance: 10th Nov
Subject Name: DS
Subject Code: 21CSH-211

Aim of the practical: Write a program to implement of different operation on a binary search tree

Algorithm:

Insertion-
1. Create a new BST node and assign values to it.
2. insert(node, key)
if root == NULL,
return the new node to the calling function.
if root=>data < key
call the insert function with root=>right and assign the return value in root=>right.
root->right = insert(root=>right,key)
if root=>data > key
call the insert function with root->left and assign the return value in root=>left.
root=>left = insert(root=>left,key)
3. Finally, return the original root pointer to the calling function.

Deletion-
1.Leaf Node
If the node is leaf (both left and right will be NULL), remove the node directly and free its memory.
2. Node with Right Child
If the node has only right child (left will be NULL), make the node points to the right node and free the node.
3. Node with Left Child
If the node has only left child (right will be NULL), make the node points to the left node and free the node.
4. Node has both left and right child
If the node has both left and right child,
find the smallest node in the right subtree. say min
make node->data = min
Again delete the min node.

Program code:

```
#include <bits/stdc++.h>
using namespace std;
#define COUNT 10

struct Node{
    int val;
    Node* left; //smaller
    Node* right; //Greater
    Node(int data){
        val=data;
        left=NULL;
        right=NULL;
    }
};

void inorder(Node* root){
    if(!root) return;

    inorder(root->left);
    cout<<root->val<<" ";
    inorder(root->right);
}

int successor(Node* root){ // for node 'x' successor call successor(x->right);
    if(!root->left)
        return root->val;
    return successor(root->left);
}

Node* insert(int val, Node* root){
    // Base case
    if(!root){
        return new Node(val);
    }

    if(val<root->val)
        root->left = insert(val,root->left);
    else
        root->right = insert(val, root->right);

    return root;
}

Node* del(int target, Node* root){
    if(!root) return root;
```

```
    if(root->val>target){
        root->left = del(target,root->left);
    }
    else if(root->val<target){
        root->right = del(target, root->right);
    }
    else{
        if(!root->left && !root->right){
            free(root);
            return NULL;
        }
        else if(!root->left){
            Node* tmp = root->right;
            free(root);
            return tmp;
        }
        else if(!root->right){
            Node* tmp = root->left;
            free(root);
            return tmp;
        }
        int imm_successor = successor(root->right);
        del(imm_successor,root);
        root->val = imm_successor;
        return root;
    }
    return root;
}

bool search(int val, Node* root){
    // Base case
    if(!root){
        return false;
    }
    if(root->val==val)
        return true;

    if(val<root->val)
        return search(val, root->left);
    else
        return search(val, root->right);
}
int main(){
    cout<<"Enter the no. of elements you want in BST: ";
    int sz;cin>>sz;
    cout<<"Enter "<<sz<<" elements: ";
    Node* root = NULL;
```

```
for(int i=0;i<sz;++i){
    int ele;cin>>ele;
    if(i==0)
        root = insert(ele, root);
    else
        insert(ele, root);
}
cout<<"\n";

bool flag=true;
while(flag){
    cout<<setw(19)<<"BST Menu\n";
    cout<<"-------------------------\n";
    cout<<"(1) Insert"<<setw(18)<<"(2) Delete\n";
    cout<<"(3) Search"<<setw(19)<<"(4) Display\n";
    cout<<setw(19)<<"(5) Exit\n";

    cout<<"\nWhat do u want to do? : ";
    string choice;cin>>choice;
    if(choice.size()>1) // for tackling when input is alphabet and strings.
        choice[0]='6';

    int num;
    switch(choice[0]){
        case '1':
            cout<<"Enter the element you want to Insert: ";
            cin>>num;
            if(!root)
                root= insert(num,root);
            else
                insert(num, root);
            break;

        case '2':
            cout<<"Enter the element you want to Delete: ";
            cin>>num;
            root = del(num,root);
            break;

        case '3':
            cout<<"Enter the element you want to Search: ";
            cin>>num;
            if(search(num,root))
                cout<<"Found";
            else
                cout<<"Not Found";
            break;

        case '4':
```

```
                cout<<"Inorder Traversal: ";
                inorder(root);
                break;

            case '5':
                flag = false;
                cout<<"Exiting.......";
                break;

            default:
                cout<<"Invalid Choice.....Try again!";
                break;
        }
        cout<<"\n\n";
        system("pause");
        cout<<"\033[2J\033[1;1H";
    }
    cout<<"Program Stopped!!";
}
```

Output:

```
Enter the no. of elements you want in BST: 5
Enter 5 elements:  8 2 12 17 4

          BST Menu
--------------------------
(1) Insert       (2) Delete
(3) Search       (4) Display
          (5) Exit

What do u want to do? : 1
Enter the element you want to Insert: 11

Press any key to continue . . .
```

```
          BST Menu
--------------------------
(1) Insert       (2) Delete
(3) Search       (4) Display
          (5) Exit

What do u want to do? : 2
Enter the element you want to Delete: 4

Press any key to continue . . .
```

```
          BST Menu
--------------------------
(1) Insert       (2) Delete
(3) Search       (4) Display
          (5) Exit

What do u want to do? : 3
Enter the element you want to Search: 17
Found

Press any key to continue . . .
```

```
          BST Menu
--------------------------
(1) Insert       (2) Delete
(3) Search       (4) Display
          (5) Exit

What do u want to do? : 4
Inorder Traversal: 2 8 11 12 17

Press any key to continue . . .
```