==========

Roll Number: SYCOC303 Division: C

PRN Number: 122B2B303 Batch: C4

Name: VINAYAK MADAN SHETE

==========

Problem Statement:

- **Write a C++ program to construct a binary search tree and perform insertion, deletion, searching of a node and its traversal.**

==========

INPUT:

```
/*
 *     ====================================
 *            Program Name: BST.cpp
 *     Created on: December 03, 2022
 *      Author: Vinayak Shete
 *  ====================================
 */

#include <iostream>
using namespace std;

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

class BST
{
	Node* root;
	public:
	BST()
```

```
        {
        root=NULL;
        }

        Node* getRoot()
        {
                return(this->root);
        }

        void insert(int);
        int delete_node(int);
        int search(int);
        void inorder(Node*);
        void preorder(Node*);
        void postorder(Node*);
};

//inserting an element into BST
void BST::insert(int element)
{
        Node* nn=new Node();
        Node* temp=root;
        Node* parent=NULL;
        nn->data=element;
        nn->right=nn->left=NULL;

        while(temp!=NULL)
        {
                parent=temp;
                if(element < temp->data)
                {
                        temp=temp->left;
                }
                else if(element > temp->data)
                {
                        temp=temp->right;
                }
```

```
		else
		{
			cout<<"\nThe enetered element is already inserted!Duplicate value canot be added!!";
			return;
		}
	}
	if(root==NULL)
	{
		root=nn;
		cout<<"\nYou have entered for the first time...So it is the ROOT of your BST";
	}
	else if(element < parent->data)
	{
		parent->left=nn;
		cout<<"\nThe element is enetered successfully as the left child!";
	}
	else if(element > parent->data)
	{
		parent->right=nn;
		cout<<"\nThe element is enetered successfully as the right child!!";
	}

}

//deleting an element from BST
int BST::delete_node(int del_ele)
{
	int flag=0;
	Node* temp=root;
	Node* parent=NULL;
	if(root==NULL)
	{
		cout<<"\nNo data is present in the BST. Deletion operation failed!";
		return 0;
	}
```

```
	//this loop to get the deleted node in temp and its parent
	while(temp!=NULL && temp->data!=del_ele)
	{
		parent=temp;
		if(del_ele<temp->data)
		{
			temp=temp->left;
		}
		else
		{
			temp=temp->right;
		}
	}

	//if the element to be deleted not found in the BST
	if(temp==NULL)
	{
		cout<<"\nThe element to be deleted is not present in BST. Deletion
operation failed!";
		return 0;
	}

	//if the node to be deleted has no children
	if(temp->left==NULL && temp->right==NULL)
	{
		if(parent!=NULL)
		{
			if(parent->left=temp)
			{
				parent->left=NULL;
			}
			else if(parent->right=temp)
			{
				parent->right=NULL;
			}
		}
```

```
        else
        {
                root=NULL;
        }
        delete(temp);
}
//if the node to be deleted is having only right child
else if(temp->left==NULL && temp->right!=NULL)
{
        if(parent!=NULL)
        {
                if(parent->left=temp)
                {
                        parent->left=temp->right;
                }
                else if(parent->right=temp)
                {
                        parent->right=temp->right;
                }
        }
        else
        {
                root=temp->right;
        }
        delete(temp);
}
//if the node to be deleted is having only left child
else if(temp->left!=NULL && temp->right==NULL)
{
        if(parent!=NULL)
        {
                if(parent->left=temp)
                {
                        parent->left=temp->left;
                }
                else if(parent->right=temp)
                {
```

```
                        parent->right=temp->left;
                }
        }
        else
        {
                root=temp->left;
        }
    }
    //if the node to be deleted is having both left and right children
    else if(temp->left!=NULL && temp->right!=NULL)
    {
        Node* inorder_suc=temp->right;
        while(inorder_suc->left!=NULL)
        {
                inorder_suc=inorder_suc->left;
        }
        int inorder_suc_val=inorder_suc->data;
        delete_node(inorder_suc_val);

        temp->data=inorder_suc_val;
    }
    return 1;
}

int BST::search(int ser_ele)
{
    Node* temp=root;
    Node* parent=NULL;
    if(root==NULL)
    {
        cout<<"\nNo data is present in the BST. Search operation failed!";
        return 0;
    }

    //this loop to get the deleted node in temp and its parent
    while(temp!=NULL)
    {
```

```
		if(ser_ele==temp->data)
		{
			return 1;
		}
		if(ser_ele<temp->data)
		{
			temp=temp->left;
			if(ser_ele==temp->data)
			{
				return 1;
			}
		}
		else
		{
			temp=temp->right;
			if(ser_ele==temp->data)
			{
				return 1;
			}
		}
	}

	return 0;

}


//inorder traversal
void BST::inorder(Node* temp)
{
	if(temp!=NULL)
	{
		inorder(temp->left);
		cout<<temp->data<<"  ";
		inorder(temp->right);
	}
}
```

```
//preorder traversal
void BST::preorder(Node* temp)
{
	if(temp!=NULL)
	{
		cout<<temp->data<<"  ";
		preorder(temp->left);
		preorder(temp->right);
	}
}

//postorder traversal
void BST::postorder(Node* temp)
{
	if(temp!=NULL)
	{
		postorder(temp->left);
		postorder(temp->right);
		cout<<temp->data<<"  ";
	}
}

int main()
{
	int ch,doch,ins_ele,insch;
	int del_ele,is_deleted,ser_ele,is_found;
	BST obj;
	Node* troot;
	cout<<"\n===============WELCOME===============";
	do
	{
		cout<<"\n1.Add element into BST\n2.Delete element from BST\n3.Search
element into BST\n4.Display BST using Inorder Traversal\n5.Display BST using Preoder
Traversal\n6.Display BST using Postorder Traversal\n7.EXIT";
		cout<<"\nEnter you proper choice:";
		cin>>ch;
```

```
		switch(ch)
		{
			case 1:
				do
				{
					cout<<"\nEnter the element you want to insert:";
					cin>>ins_ele;
					obj.insert(ins_ele);
					cout<<"\n====================\nDo you want to
continue Insertion?[Press 1 for YES|| 0 for NO]-->";
					cin>>insch;
				}while(insch==1);
				break;
			case 2:
				cout<<"\nEnter the element you want to delete:";
				cin>>del_ele;
				is_deleted=obj.delete_node(del_ele);
				if(is_deleted)
				{
					cout<<"\nThe node with element "<<del_ele<<" is
deleted successfully!";
					cout<<"\n========================================";
					cout<<"\nThe inorder traversal of the BST after
deletion operation is:\n";
					troot=obj.getRoot();
					obj.inorder(troot);
					cout<<"\n========================================";
				}
				break;
			case 3:
				cout<<"\nEnter the element you want to search for:";
				cin>>ser_ele;
				is_found=obj.search(ser_ele);
				if(is_found)
				{
					cout<<"\nThe node with element "<<ser_ele<<" is found
successfully!";
					cout<<"\n========================================";
```

```
                        }
                        else
                        {
                                cout<<"\nThe node with element "<<ser_ele<<" is not
found";
                                cout<<"\n=======================================";
                        }
                break;
                case 4:
                        cout<<"\n=======================================";
                        cout<<"\nThe inorder traversal of the BST is:\n";
                        troot=obj.getRoot();
                        obj.inorder(troot);
                        cout<<"\n=======================================";
                        break;
                case 5:
                        cout<<"\n=======================================";
                        cout<<"\nThe preorder traversal of the BST is:\n";
                        troot=obj.getRoot();
                        obj.preorder(troot);
                        cout<<"\n=======================================";
                        break;
                case 6:
                        cout<<"\n=======================================";
                        cout<<"\nThe postorder traversal of the BST is:\n";
                        troot=obj.getRoot();
                        obj.postorder(troot);
                        cout<<"\n=======================================";
                        break;
                case 7:
                        goto exit;
                        break;
                default:
                        cout<<"\nPlease enter Correct Choice!!";
                        break;
        }
```

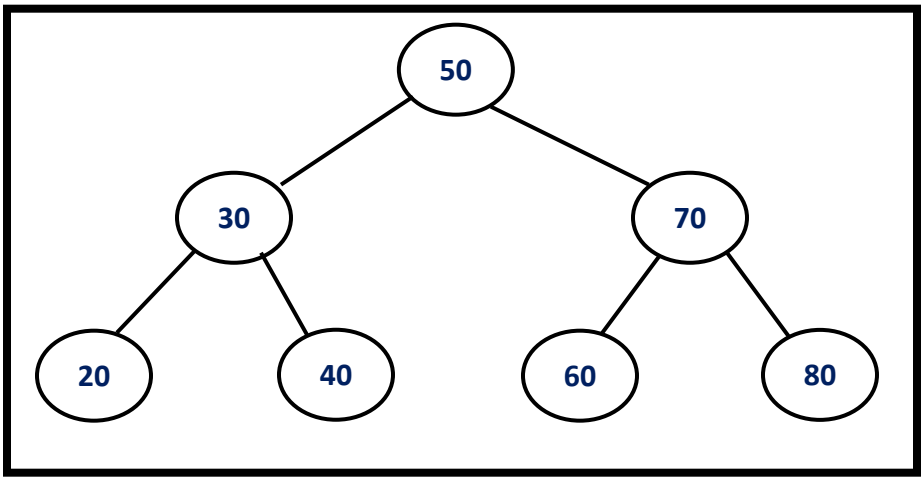
```
            cout<<"\n====================\nDo you want to continue with main
menu?[Press 1 for YES|| 0 for NO]-->";
            cin>>doch;
      }while(doch==1);
      exit:
      cout<<"\n=================THANK YOU============";
      return 0;
}
```

==========

OUTPUT:

```
===============WELCOME================
1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:
```

Let's Create the Binary Search Tree as shown in the following figue:



Adding Nodes into the BST:

```
Enter you proper choice:1

Enter the element you want to insert:50

You have entered for the first time...So it is the ROOT of your BST
====================
Do you want to continue Insertion?[Press 1 for YES|| 0 for NO]-->1

Enter the element you want to insert:30

The element is enetered successfully as the left child!
====================
Do you want to continue Insertion?[Press 1 for YES|| 0 for NO]-->1

Enter the element you want to insert:70

The element is enetered successfully as the right child!!
====================
Do you want to continue Insertion?[Press 1 for YES|| 0 for NO]-->1

Enter the element you want to insert:20

The element is enetered successfully as the left child!
====================
Do you want to continue Insertion?[Press 1 for YES|| 0 for NO]-->1

Enter the element you want to insert:40

The element is enetered successfully as the right child!!
====================
Do you want to continue Insertion?[Press 1 for YES|| 0 for NO]-->1

Enter the element you want to insert:60

The element is enetered successfully as the left child!
====================
Do you want to continue Insertion?[Press 1 for YES|| 0 for NO]-->1

Enter the element you want to insert:80

The element is enetered successfully as the right child!!
====================
Do you want to continue Insertion?[Press 1 for YES|| 0 for NO]-->0
```

Displaying with all types of Traversal:

```
1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:4

========================================
The inorder traversal of the BST is:
20  30  40  50  60  70  80
========================================
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->1
```

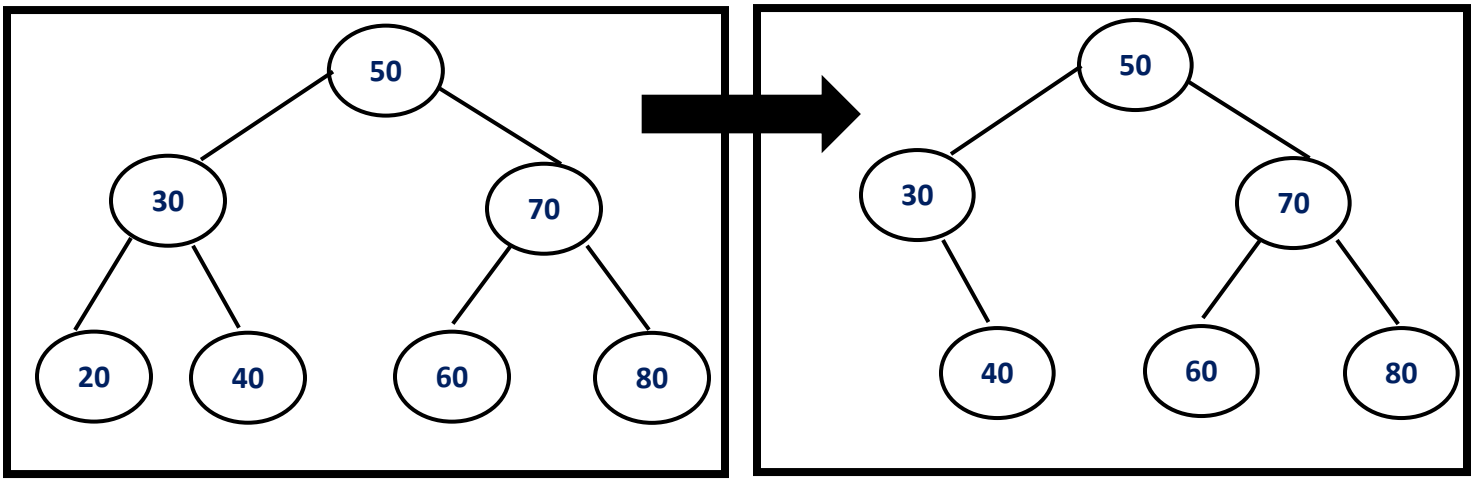
```
1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:5

========================================
The preorder traversal of the BST is:
50  30  20  40  70  60  80
========================================
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->1

1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:6

========================================
The postorder traversal of the BST is:
20  40  30  60  80  70  50
========================================
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->
```

Searching an Element into BST:

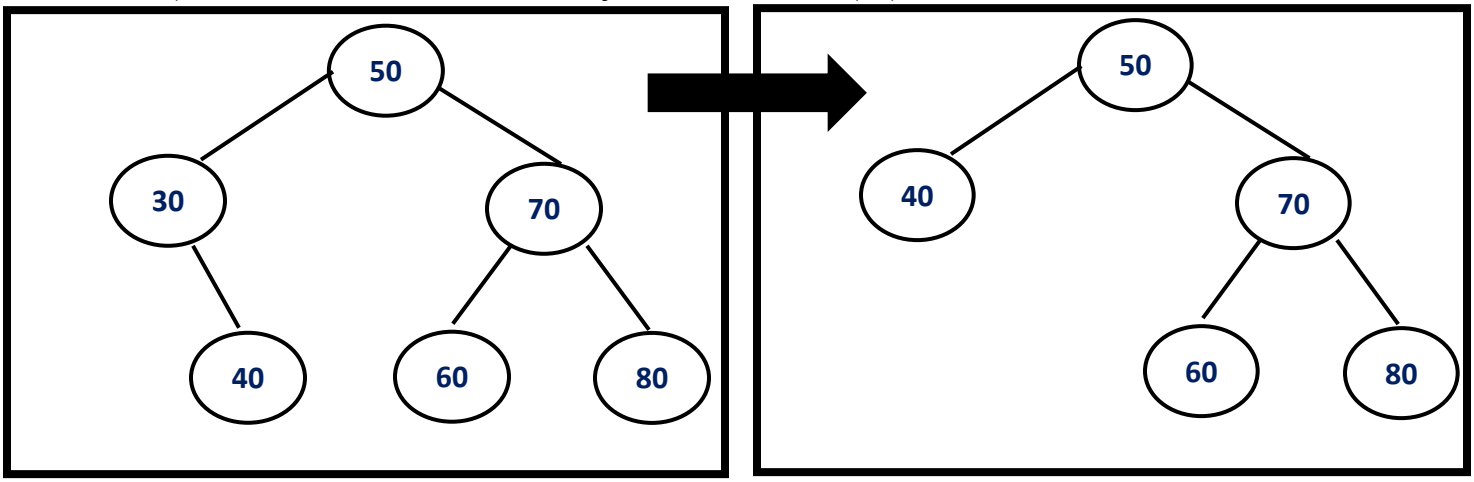
```
1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:3

Enter the element you want to search for:60

The node with element 60 is found successfully!
========================================
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->
```

Deleting Nodes from the BST:

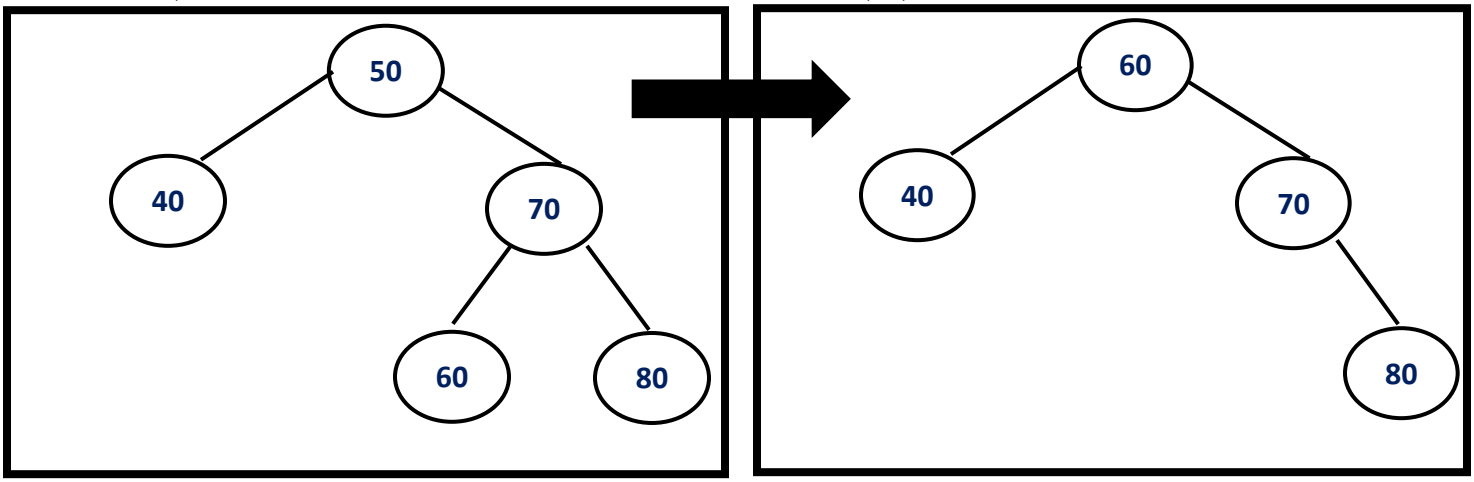
1) **The node to be deleted is a leaf node: delete(20)**



2) **The node to be delete has only one child: delete(30)**



3) **The node to be deleted has both children: delete(50)**



```
1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:2

Enter the element you want to delete:20

The node with element 20 is deleted successfully!
========================================
The inorder traversal of the BST after deletion operation is:
30  40  50  60  70  80
========================================
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->1
```

```
1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:2

Enter the element you want to delete:30

The node with element 30 is deleted successfully!
========================================
The inorder traversal of the BST after deletion operation is:
40  50  60  70  80
========================================
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->1
```

```
1.Add element into BST
2.Delete element from BST
3.Search element into BST
4.Display BST using Inorder Traversal
5.Display BST using Preoder Traversal
6.Display BST using Postorder Traversal
7.EXIT
Enter you proper choice:2

Enter the element you want to delete:50

The node with element 50 is deleted successfully!
========================================
The inorder traversal of the BST after deletion operation is:
40  60  70  80
========================================
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->
```

```
====================
Do you want to continue with main menu?[Press 1 for YES|| 0 for NO]-->0

=================THANK YOU=============
--------------------------------
Process exited after 845 seconds with return value 0
Press any key to continue . . .
```

==========