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

struct BST
{
    int data;
    BST *left_child, *right_child;
    BST(int value)
    {
        data = value;
        left_child = NULL;
        right_child = NULL;
    }
};

BST* InsertNODE(BST* root, int
inserting_value)
{
    if (root == NULL)
        return new
BST(inserting_value);
   else if (inserting_value <
root->data)
        root->left_child =
InsertNODE(root->left_child,
inserting_value);
    else
        root->right_child =
InsertNODE(root->right_child,
inserting_value);
    return root;
}

bool Search(BST* root, int value)
{
    if (root == NULL)
        return false;
  else  if (root->data == value)
        return true;
  else  if (value < root->data)
        return Search(root-
>left_child, value);
    else
        return Search(root-
>right_child, value);
}
```

```
void PrintPreorder(BST* root)
{
    if (root == NULL)
        return;
    cout << root->data << " ";
    PrintPreorder(root-
>left_child);
    PrintPreorder(root-
>right_child);
}
void PrintPostorder(BST* root)
{
    if (root == NULL)
        return;
    PrintPostorder(root-
>left_child);
    PrintPostorder(root-
>right_child);
    cout << root->data << " ";
}

int main()
{
    BST* root = NULL;
    cout << "Enter the number of
inserting values: ";
    int n;
    cin >> n;
    while (n--)
    {
        int value;
        cin >> value;
        root = InsertNODE(root,
value);
    }

    cout << "In-order traversal of
the BST: ";
    PrintInorder(root);
    cout << endl;

    cout << "Pre-order traversal
of the BST: ";
    PrintPreorder(root);
    cout << endl;
```

```
void PrintInorder(BST* root)
{
    if (root == NULL)
        return;
    PrintInorder(root->left_child);
    cout << root->data << " ";
    PrintInorder(root->right_child);
}
        cout << searchValue << " is found in the BST." << endl;
    else
```

```
    cout << "Post-order traversal of the BST: ";
    PrintPostorder(root);
    cout << endl;

    cout << "\nEnter a value to search for: ";
    int searchValue;
    cin >> searchValue;

    if (Search(root, searchValue))
        cout << searchValue << " is not found in the BST." << endl;

    return 0;
}
```