Assignment 5

Program:

```
#include <iostream>

using namespace std;


class Node

{

public:

   int iData;

   Node *pLeft;

   Node *pRight;

   int iLThread;

   int iRThread;


   Node(int value)

   {

      iData = value;

      pLeft = NULL;

      pRight = NULL;

      iLThread = 1;

      iRThread = 1;

   }

};

Node *TbtInsert(Node *ppRoot, int iData)

{

   Node *pTemp = ppRoot;
Node *pSearch = NULL;
Node *pNewNode = new Node(iData);


while (pTemp != NULL)
{
    pSearch = pTemp;
    if (ppRoot->iData == iData)
    {
        return ppRoot;
    }


    if (iData < pTemp->iData)
    {
        if (pTemp->iLThread == 0)
        {
            pTemp = pTemp->pLeft;
        }
        else
            break;
    }
    else
    {
        if (pTemp->iRThread == 0)
        {
            pTemp = pTemp->pRight;
        }
```

```
            else
                break;
        }
    }
    if (ppRoot == NULL)
    {
        ppRoot = pNewNode;
        return ppRoot;
    }
    else if (iData < pSearch->iData)
    {
        pNewNode->pLeft = pSearch->pLeft;
        pNewNode->pRight = pSearch;
        pSearch->pLeft = pNewNode;
        pSearch->iLThread = 0;
    }
    else
    {
        pNewNode->pLeft = pSearch;
        pNewNode->pRight = pSearch->pRight;
        pSearch->pRight = pNewNode;
        pSearch->iRThread = 0;
    }
    return ppRoot;
}
```

```
void Input(Node *&pRoot)
{
    int iData;
    cout << "Give -1 to Stop";
    cout << "\nEnter the data : ";
    cin >> iData;
    while (iData != -1)
    {
        pRoot = TbtInsert(pRoot, iData);
        cin >> iData;
    }
}
Node *InOrderSuccessor(Node *pTemp)
{
    if (pTemp->iRThread == 1)
    {
        return pTemp->pRight;
    }

    pTemp = pTemp->pRight;

    while (pTemp->iLThread == 0)
    {
        pTemp = pTemp->pLeft;
    }
    return pTemp;
```

```
}

void InOrder(Node *pRoot)

{

    Node *pTemp = pRoot;


    while (pTemp->iLThread == 0)

    {

        pTemp = pTemp->pLeft;

    }

    while (pTemp != NULL)

    {

        cout << pTemp->iData << "\t";

        pTemp = InOrderSuccessor(pTemp);

    }

}

int main()

{

    int iChoice;

    Node *pRoot = NULL;

    while (1)

    {

        cout << "Enter the choice : \n1.Input Data \n2.InOrder Traversal";

        cin >> iChoice;


        switch (iChoice)
        {
        case 1:
            Input(pRoot);
            break;
        case 2:
            InOrder(pRoot);
            break;
case 3:
            return 0;
        }
    }
}
```

Output:

```
Enter the choice :
1.Input Data
2.InOrder Traversal
3.Exit1
Give -1 to Stop
Enter the data : 50
20
40
50
68
72
-1
Enter the choice :
```

```
1.Input Data

2.InOrder Traversal

3.Exit2

20   40   50   68   72    Enter the choice :

1.Input Data

2.InOrder Traversal

3.Exit3
```