Assignment6.cpp

```cpp
#include <iostream>
#include <cstdlib>
#define MAX_VALUE 65536
using namespace std;


struct N {   //node declaration
    int k;
   N *l, *r;
   bool leftTh,  rightTh;
};

//global root
N *root;

  //creating the inorder threaded BT
void insert(int key)
{
N *ptr = NULL;
   if(root==NULL)
   {
     root= new N();
         root->r= root->l= root;
         root->leftTh = true;
         root->k = MAX_VALUE;
   }
   /*else
   {
     ptr= new N();
         ptr->r= root->l= root;
         ptr->leftTh = true;
         ptr->k = MAX_VALUE;
   }*/
    N *p = root;
    for (;;) {
        if (p->k< key) { //move to right thread
            if (p->rightTh)
            break;
            p = p->r;
        } else if (p->k > key) { // move to left thread
            if (p->leftTh)
            break;
            p = p->l;
        } else {
            return;
        }
    }
    N *temp = new N();
    temp->k = key;
    temp->rightTh= temp->leftTh= true;
```

```
51      if (p->k < key) {
52          temp->r = p->r;
53          temp->l= p;
54          p->r = temp;
55          p->rightTh= false;
56      } else {
57          temp->r = p;
58          temp->l = p->l;
59          p->l = temp;
60          p->leftTh = false;
61      }
62     }
63
64
65  void displayTree() { //print the tree
66     N *temp = root, *p;
67     for (;;) {
68      p = temp;
69      temp = temp->r;
70      if (!p->rightTh) {
71          while (!temp->leftTh) {
72              temp = temp->l;
73          }
74      }
75      if (temp == root)
76          break;
77      cout<<temp->k<<" ";
78     }
79     cout<<endl;
80  }
81
82  void preorderDisplayTree(){
83      N *temp = root, *p;
84     for (;;) {
85      p = temp;
86
87      // Visit the current node (pre-order)
88      if (temp != root) {
89          cout << temp->k << " ";
90      }
91
92      // If there is a left child, move to it
93      if (!temp->leftTh) {
94          temp = temp->l;
95      }
96      // Else move to the right threaded node
97      else {
98          while (temp->rightTh && temp->r != root) {
99              temp = temp->r;
100         }
101         temp = temp->r;
102         if (temp == root) {
103             break;
```

```
        }
     }
    }
    cout<<endl;
}


int main() {

    cout<<"Inorder ThreadedBinaryTree\n";
    char ch;
    int c, v;
    while(1) {
     cout<<"1. Insert "<<endl;
     cout<<"2. Inorder Traversal"<<endl;
     cout<<"3. Preorder Traversal"<<endl;
     cout<<"6. Exit"<<endl;
     cout<<"Enter Your Choice: ";
     cin>>c;
     //perform switch operation
     switch (c) {
         case 1 :
             cout<<"Enter integer element to insert: ";
             cin>>v;
             insert(v);
             break;
         case 2:
             cout<<"In-order Display tree: \n ";
             displayTree();
             break;
         case 3:
             cout<<"Pre-order Display tree: \n";
             preorderDisplayTree();
             cout<<endl;
             break;
         case 6:
             exit(1);
         default:
             cout<<"\nInvalid type! \n";
     }
    }
    cout<<"\n";
    return 0;
}
/*
Sample output
Inorder ThreadedBinaryTree
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 50
```

```
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 25
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 75
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 20
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 40
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 60
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 80
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 3
Pre-order Display tree:
50 25 20 40 75 60 80

1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 1
Enter integer element to insert: 85
1. Insert
2. Inorder Traversal
3. Preorder Traversal
```

```
6. Exit
Enter Your Choice: 2
In-order Display tree:
 20 25 40 50 60 75 80 85
1. Insert
2. Inorder Traversal
3. Preorder Traversal
6. Exit
Enter Your Choice: 6
*/
```