```
/*Implement In-order Threaded Binary Tree and traverse it in In-order and Pre-order. */

#include <iostream>

struct Node {
   int data;
   Node* left;
   Node* right;
   bool isThreaded;  // Flag to indicate if the right pointer is a thread
};

// Helper function to create a new node
Node* createNode(int data) {
   Node* newNode = new Node;
   newNode->data = data;
   newNode->left = newNode->right = nullptr;
   newNode->isThreaded = false;
   return newNode;
}

// Function to insert a node into the threaded binary tree
Node* insert(Node* root, int data) {
   if (!root) {
      return createNode(data);
   }

   if (data < root->data) {
      root->left = insert(root->left, data);
   } else if (data > root->data) {
      if (root->isThreaded) {
         Node* temp = root->right;
```

```
            root->right = createNode(data);
            root->isThreaded = false;
            root->right->right = temp;
        } else {
            root->right = insert(root->right, data);
        }
    }

    return root;
}

// Function to perform an in-order traversal using threads
void inOrderThreadedTraversal(Node* root) {
    Node* current = root;
    while (current) {
        while (current->left) {
            current = current->left;
        }

        std::cout << current->data << " ";

        if (current->isThreaded) {
            current = current->right;
        } else {
            current = current->right;
        }
    }
}

// Function to perform a pre-order traversal
void preOrderTraversal(Node* root) {
```

```
    if (!root) {
        return;
    }

    std::cout << root->data << " ";
    preOrderTraversal(root->left);
    preOrderTraversal(root->right);
}

int main() {
    Node* root = nullptr;
    int n, data;

    std::cout << "Enter the number of nodes: ";
    std::cin >> n;

    std::cout << "Enter the data for each node: ";
    for (int i = 0; i < n; ++i) {
        std::cin >> data;
        root = insert(root, data);
    }

    std::cout << "In-order Traversal: ";
    inOrderThreadedTraversal(root);
    std::cout << "\nPre-order Traversal: ";
    preOrderTraversal(root);

    return 0;
}
```