Tree: Preorder Traversal

```
#include <iostream>
#include <cstddef>

class Node {
    public:
        int data;
        Node *left;
        Node *right;
        Node(int d) {
            data = d;
            left = NULL;
            right = NULL;
        }
};

class Solution {
    public:
        Node* insert(Node* root, int data) {
            if(root == NULL) {
                return new Node(data);
            } else {
                Node* cur;
                if(data <= root->data) {
                    cur = insert(root->left, data);
                    root->left = cur;
                } else {
                    cur = insert(root->right, data);
                    root->right = cur;
                }

               return root;
           }
        }
#include <iostream>
// using namespace std;
/* you only have to complete the function given below.
Node is defined as

class Node {
    public:
        int data;
        Node *left;
        Node *right;
        Node(int d) {
            data = d;
            left = NULL;
            right = NULL;
        }
};

*/

    void preOrder(Node *root) {
    if(!root) return;
    std::cout<<root->data<<" ";
    preOrder(root->left);
    preOrder(root->right);
    }
}; //End of Solution

int main() {

    Solution myTree;
    Node* root = NULL;

    int t;
    int data;

    std::cin >> t;

    while(t-- > 0) {
        std::cin >> data;
        root = myTree.insert(root, data);
    }

    myTree.preOrder(root);

    return 0;
}
```