```cpp
#include <bits/stdc++.h>

using namespace std;

class treeNode
{

public:
    int data;
    treeNode *leftchild;
    treeNode *rightchild;

    treeNode(int value)
    {

        data = value;
        leftchild = NULL;
        rightchild = NULL;
    }
};

void spacePrint(int level)
{
    for (int i = 0; i < level; i++)
    {
        cout << "   ";
    }
}

void printTree(treeNode *root, int level)
{
    if (root == NULL)
    {
        return;
    }
    if (root->leftchild == NULL && root->rightchild == NULL)
    {
        cout << root->data << endl;
    }
    else
    {
        cout << endl;
        spacePrint(level);
        cout << " Root: " << root->data << endl;
    }

    if (root->leftchild != NULL)
    {
        spacePrint(level);
        cout << " Left: ";
        printTree(root->leftchild, level + 1);
    }

    if (root->rightchild != NULL)
    {
        spacePrint(level);
        cout << " Right: ";
        printTree(root->rightchild, level + 1);
    }
}

void inOrder(treeNode *root, string &chk)
{
    if (root == NULL)
        return;

    inOrder(root->leftchild, chk);
    chk += to_string(root->data);
    inOrder(root->rightchild, chk);
}

void PreOrderTravarsal(treeNode *root, string &chk)
{
    if (root == NULL)
        return;

    chk += to_string(root->data);
    PreOrderTravarsal(root->leftchild, chk);
    PreOrderTravarsal(root->rightchild, chk);
```

```
}

void PostOrder(treeNode *root, string &chk)
{
    if (root == NULL)
        return;

    PostOrder(root->leftchild, chk);
    PostOrder(root->rightchild, chk);
    chk += to_string(root->data);
}

int searchInOrder(int inOrder[], int current, int start, int end)
{
    for (int i = start; i <= end; i++)
    {
        if (inOrder[i] == current)
        {
            return i;
        }
    }

    return -1;
}

treeNode *buildTreePreIn(int preOrder[], int inOrder[], int start, int
end)
{
    static int id = 0;
    int current = preOrder[id];
    id++;
    treeNode *newNode = new treeNode(current);

    if (start == end)
    {
        return newNode;
    }
    int pos = searchInOrder(inOrder, current, start, end);
    newNode->leftchild = buildTreePreIn(preOrder, inOrder, start, pos -
1);
    newNode->rightchild = buildTreePreIn(preOrder, inOrder, pos + 1, end);
    return newNode;
}

int main()
{


    int n;
    cin >> n;
    int preOrder[n], inOrder[n];

    for (int i = 0; i < n; i++)
    {
        cin >> preOrder[i];
    }

    for (int i = 0; i < n; i++)
    {
        cin >> inOrder[i];
    }
    treeNode *root = buildTreePreIn(preOrder, inOrder, 0, n - 1);
    string chkpre = "";
    PreOrderTravarsal(root, chkpre);
    cout << chkpre << endl
         << endl;
    return 0;
}
```