# Is Symmetric in C++

```cpp
#include <iostream>
#include <vector>
#include <stack>

using namespace std;

// Node class definition
class Node {
public:
    int data;
    vector<Node*> children;

    Node(int val) {
        data = val;
    }
};

// Function to construct the tree
from the given array
Node* construct(vector<int>&
arr) {
    Node* root = nullptr;
    stack<Node*> st;

    for (int i = 0; i < arr.size(); ++i) {
        if (arr[i] == -1) {
            st.pop();
        } else {
            Node* t = new
Node(arr[i]);

            if (!st.empty()) {
                st.top()-
>children.push_back(t);
            } else {
                root = t;
            }

            st.push(t);
        }
    }

    return root;
}

// Function to check if two trees
are mirrors of each other
bool areMirror(Node* n1, Node*
n2) {
    if (n1->children.size() != n2-
>children.size()) {
        return false;
    }

    for (int i = 0; i < n1-
>children.size(); ++i) {
        int j = n1->children.size() - 1
- i;
        Node* c1 = n1->children[i];
        Node* c2 = n2->children[j];
```

**Tree Structure from Input**

```
10
├── 20
│   ├── 50
│   └── 60
├── 30
│   ├── 70
│   ├── 80
│   └── 90
└── 40
    ├── 100
    └── 110
```

## 📋 Tabular Dry Run of are Mirror (node1, node2)

| Step | node1->data | node2->data | Children Count Match | Comparing Child Pair | Recursive Call | Result |
|---|---|---|---|---|---|---|
| 1 | 10 | 10 | ✓ Yes (3 children) | Compare 20 & 40 | areMirror(20, 40) | proceeds |
| 2 | 20 | 40 | ✓ Yes (2 children) | Compare 50 & 110 | areMirror(50, 110) | ✓ true |
| 3 | 50 | 110 | ✓ Yes (0 children) | - | leaf nodes | ✓ true |
| 4 | 20 | 40 | - | Compare 60 & 100 | areMirror(60, 100) | ✓ true |
| 5 | 60 | 100 | ✓ Yes (0 children) | - | leaf nodes | ✓ true |
| 6 | 20 & 40 | done | All children matched | - | return to previous | ✓ true |
| 7 | 10 | 10 | - | Compare 30 & 30 (middle node) | areMirror(30, 30) | proceeds |
| 8 | 30 | 30 | ✓ Yes (3 children) | Compare 70 & 90 | areMirror(70, 90) | ✓ true |
| 9 | 70 | 90 | ✓ Yes (0 | - | leaf nodes | ✓ true |

```
        if (!areMirror(c1, c2)) {
            return false;
        }
    }

    return true;
}

// Function to check if a tree is
symmetric
bool IsSymmetric(Node* node) {
    return areMirror(node, node);
}

// Main function
int main() {
    vector<int> arr = {10, 20, 50, -1,
60, -1, -1, 30, 70, -1, 80, -1, 90, -1,
-1, 40, 100, -1, 110, -1, -1, -1};

    Node* root = construct(arr);
    bool sym = IsSymmetric(root);
    cout << boolalpha << sym <<
endl;

    return 0;
}
```

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | children) | | | |
| 10 | 30 | 30 | - | Compare 80 & 80 | areMirror(80, 80) | ✅ true |
| 11 | 80 | 80 | ✅ Yes (0 children) | - | leaf nodes | ✅ true |
| 12 | 30 | 30 | - | Compare 90 & 70 | areMirror(90, 70) | ✅ true |
| 13 | 90 | 70 | ✅ Yes (0 children) | - | leaf nodes | ✅ true |
| 14 | 30 & 30 | done | All children matched | - | return to previous | ✅ true |
| 15 | 10 | 10 | - | Compare 40 & 20 | already compared in step 1 | ✅ true |
| 16 | 10 & 10 | done | All pairs matched | - | final result | ✅ true |

✅ **Final Result:**

true

true