

Binary Tree Insertion

```
#include <iostream>
using namespace std;
```

```
class Node {
public:
    int data;
    Node* left;
    Node* right;

    Node(int val) {
        data = val;
        left = right = NULL;
    }
};
```

// Recursive Insertion in Binary Tree

```
Node* insert(Node* root, int key) {
    if (root == NULL) {
        return new Node(key); // If tree is empty, new node becomes root
    }

    // Insert in the first available position
    if (root->left == NULL) {
        root->left = new Node(key);
    }
    else if (root->right == NULL) {
        root->right = new Node(key);
    }
    else {
        // Keep inserting in left subtree (simple recursive choice)
        root->left = insert(root->left, key);
    }
    return root;
}
```

// Inorder Traversal

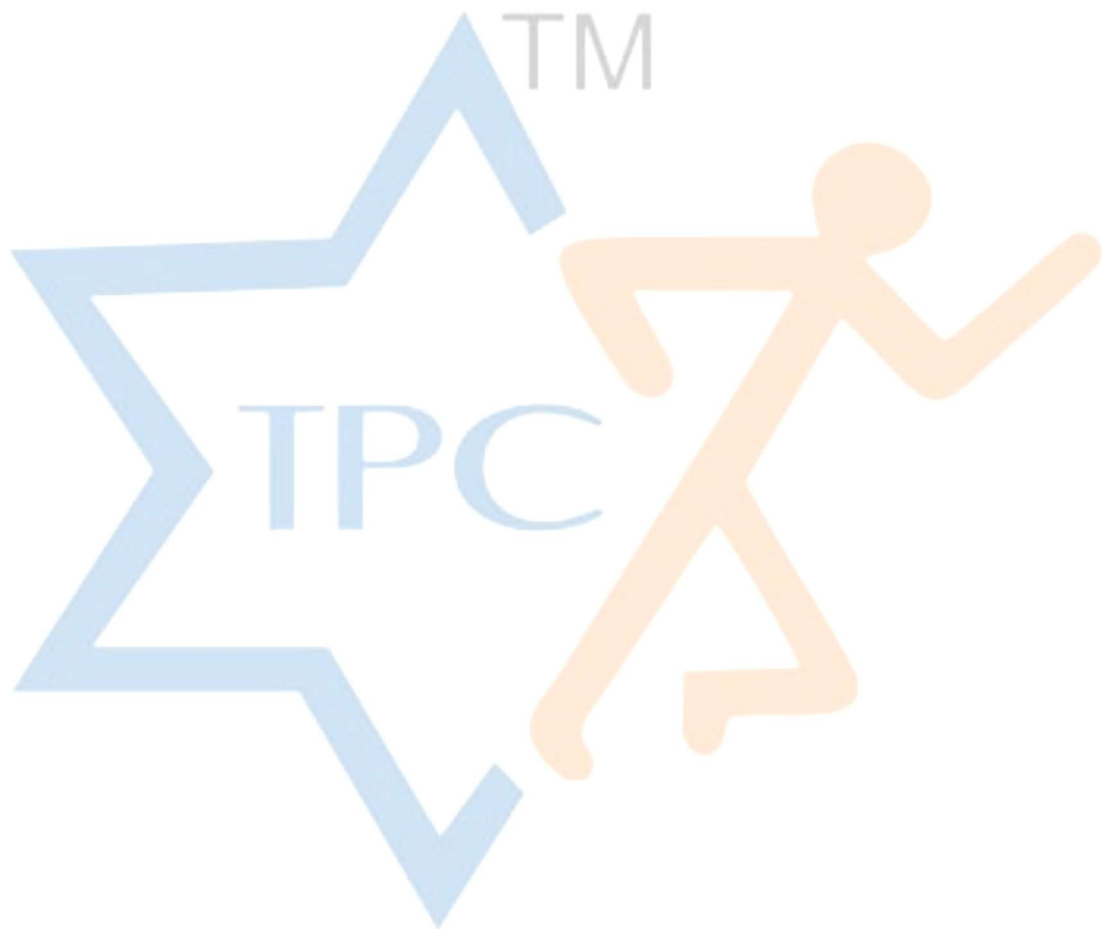
```
void inorder(Node* root) {  
    if (root == NULL) return;  
    inorder(root->left);  
    cout << root->data << " ";  
    inorder(root->right);  
}
```

```
int main() {  
    Node* root = NULL;  
  
    // Insert nodes  
    root = insert(root, 1);  
    root = insert(root, 2);  
    root = insert(root, 3);  
    root = insert(root, 4);  
    root = insert(root, 5);  
  
    cout << "Inorder Traversal of Tree: ";  
    inorder(root);  
    cout << endl;  
  
    return 0;  
}
```

Output-

Inorder Traversal of Tree: 4 2 5 1 3

www.tpcglobal.in



www.tpcglobal.in