```c
#include <stdio.h>
#include <stdlib.h>
struct TreeNode {
    int val;
    struct TreeNode* left;
    struct TreeNode* right;
};
void inorderTraversal(struct TreeNode* root) {
    if (root != NULL) {
        inorderTraversal(root->left);
        printf("%d ", root->val);
        inorderTraversal(root->right);
    }
}
void postorderTraversal(struct TreeNode* root) {
    if (root != NULL) {
        postorderTraversal(root->left);
        postorderTraversal(root->right);
        printf("%d ", root->val);
    }
}
void freeTree(struct TreeNode* root) {
    if (root != NULL) {
        freeTree(root->left);
        freeTree(root->right);
        free(root);
    }
}
int main() {
    // Create the nodes
    struct TreeNode* root = (struct TreeNode*)malloc(sizeof(struct TreeNode));
```

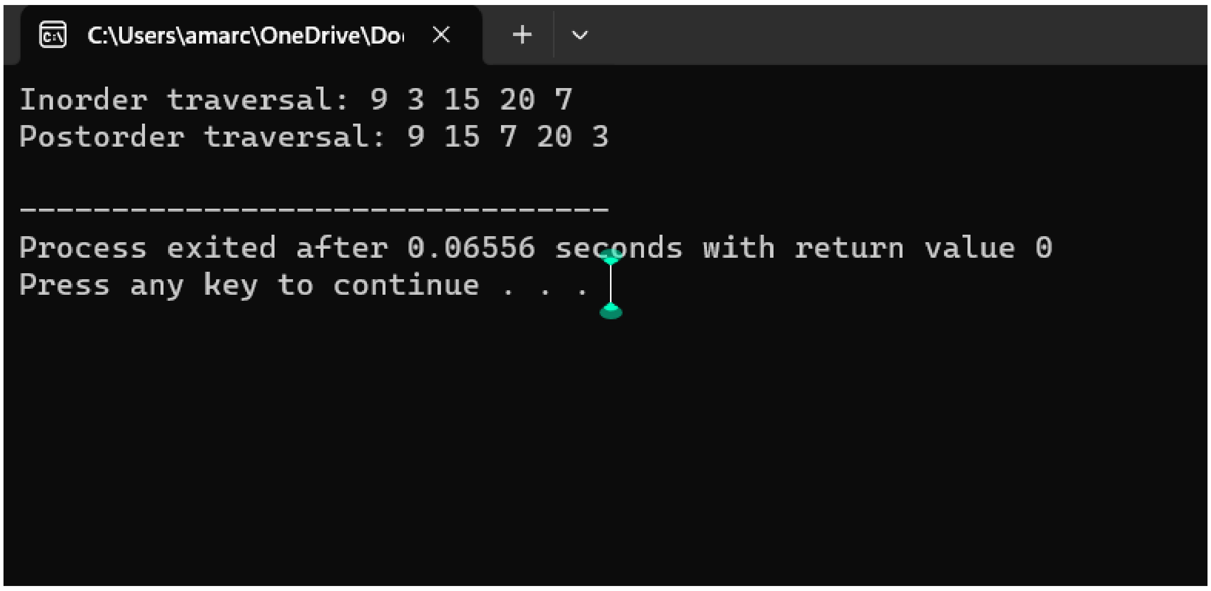
```
 root->val = 3;
root->left = (struct TreeNode*)malloc(sizeof(struct TreeNode));
 root->left->val = 9;
 root->left->left = NULL;
 root->left->right = NULL;
 root->right = (struct TreeNode*)malloc(sizeof(struct TreeNode));
 root->right->val = 20;
 root->right->left = (struct TreeNode*)malloc(sizeof(struct TreeNode));
 root->right->left->val = 15;
 root->right->left->left = NULL;
 root->right->left->right = NULL;
 root->right->right = (struct TreeNode*)malloc(sizeof(struct TreeNode));
 root->right->right->val = 7;
 root->right->right->left = NULL;
 root->right->right->right = NULL;
 printf("Inorder traversal: ");
 inorderTraversal(root);
 printf("\n");

 // Print postorder traversal
 printf("Postorder traversal: ");
 postorderTraversal(root);
 printf("\n");

 // Free memory allocated for the tree nodes
 freeTree(root);

 return 0;
```

```
}
```



```
Inorder traversal: 9 3 15 20 7
Postorder traversal: 9 15 7 20 3

--------------------------------
Process exited after 0.06556 seconds with return value 0
Press any key to continue . . .
```