

14. Write a C program to implement the Tree Traversals (Inorder, Preorder, Postorder)

```
#include <stdio.h>

#include <stdlib.h>

struct Node {
    int data;
    struct Node *left, *right;
};

struct Node* createNode(int value) {
    struct Node* newNode = (struct Node*) malloc(sizeof(struct Node));
    newNode->data = value;
    newNode->left = newNode->right = NULL;
    return newNode;
}

void inorder(struct Node* root) {
    if (root != NULL) {
        inorder(root->left);
        printf("%d ", root->data);
        inorder(root->right);
    }
}

void preorder(struct Node* root) {
    if (root != NULL) {
        printf("%d ", root->data);
        preorder(root->left);
        preorder(root->right);
    }
}

void postorder(struct Node* root) {
    if (root != NULL) {
        postorder(root->left);
```

```

        postorder(root->right);

        printf("%d ", root->data);
    }
}

int main() {

    struct Node* root = createNode(1);

    root->left = createNode(2);

    root->right = createNode(3);

    root->left->left = createNode(4);

    root->left->right = createNode(5);

    printf("Inorder Traversal: ");

    inorder(root);

    printf("\nPreorder Traversal: ");

    preorder(root);

    printf("\nPostorder Traversal: ");

    postorder(root);

    return 0;

}

```

main.c	Output
<pre> 1 #include <stdio.h> 2 #include <stdlib.h> 3 struct Node { 4 int data; 5 struct Node *left, *right; 6 }; 7 struct Node* createNode(int value) { 8 struct Node* newNode = (struct Node*) malloc(sizeof(struct Node 9)); 9 newNode->data = value; 10 newNode->left = newNode->right = NULL; 11 return newNode; 12 } 13 void inorder(struct Node* root) { 14 if (root != NULL) { 15 inorder(root->left); 16 printf("%d ", root->data); 17 inorder(root->right); 18 } 19 } 20 void preorder(struct Node* root) { 21 if (root != NULL) { 22 printf("%d ", root->data); 23 preorder(root->left); 24 preorder(root->right); 25 } 26 } 27 void postorder(struct Node* root) { 28 if (root != NULL) { </pre>	<pre> Inorder Traversal: 4 2 5 1 3 Preorder Traversal: 1 2 4 5 3 Postorder Traversal: 4 5 2 3 1 === Code Execution Successful === </pre>