Tree Traversals (Inorder, Preorder, Post Order)

```
#include <stdio.h>
#include <stdlib.h>

struct node
{
	int data;
	struct node *left;
	struct node *right;
};

struct node *newNode(int data)
{
	struct node *temp = (struct node *)malloc(sizeof(struct node));
	temp->data = data;
	temp->left = temp->right = NULL;
	return temp;
}

void preorder(struct node *root)
{
	if(root)
	{
		printf("%d\t", root->data);
		preorder(root->left);
		preorder(root->right);
	}
}

void inorder(struct node *root)
{
	if(root)
	{
		inorder(root->left);
		printf("%d\t", root->data);
		inorder(root->right);
	}
}

void postorder(struct node *root)
{
	if(root)
	{
		postorder(root->left);
		postorder(root->right);
		printf("%d\t", root->data);
	}
}

int main()
{
```

```
        struct node *root=NULL;
        root = newNode(10);
        root->left = newNode(20);
        root->right = newNode(30);
        root->left->left = newNode(40);
        root->right->left = newNode(50);
        preorder(root);
        printf("\n");
        inorder(root);
        printf("\n");
        postorder(root);
        return 0;
}
```

Time complexity: O(n)
Space Complexity: O(n)