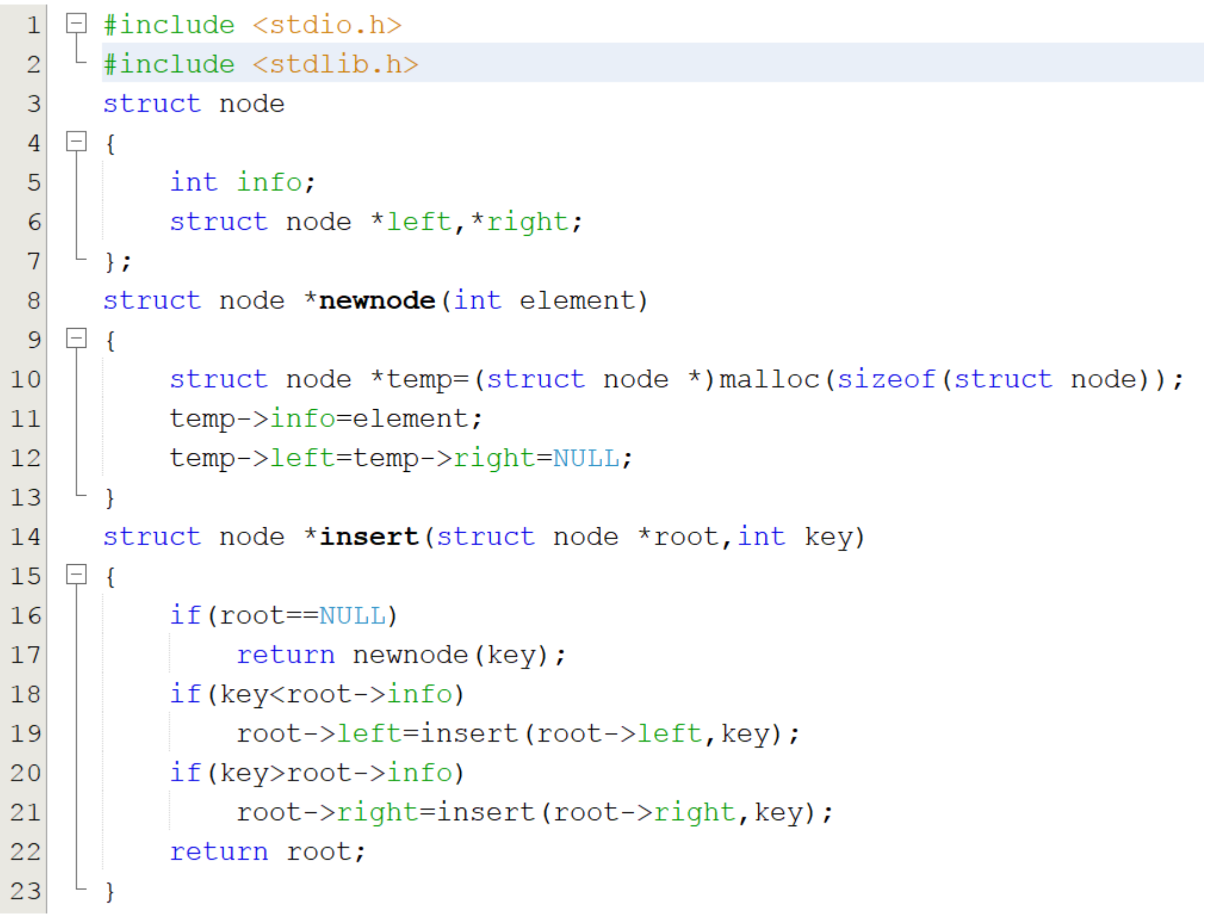
Laboratory 9

1. Questions

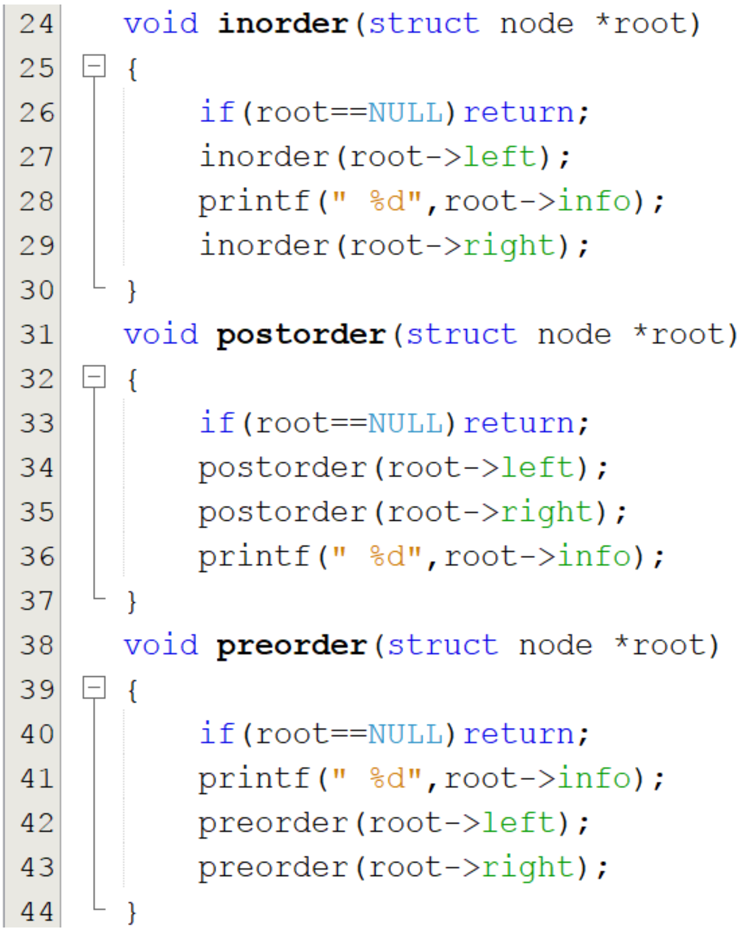
    1. Write a C program to construct a binary search tree and perform the Preorder, post order and in order
    2. Write a C program to implement a linked list to construct a tree and count the number of leaves in a tree.

2. Program

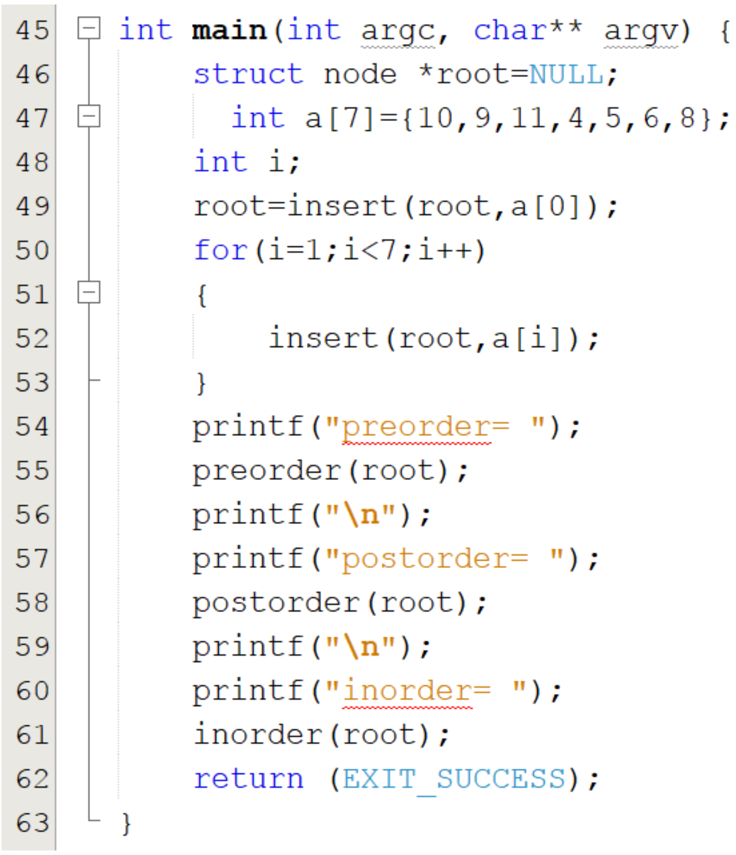
Q1) program to construct a binary search tree and perform the Pre order, post order and in order traversal:-



```c
#include <stdio.h>
#include <stdlib.h>
struct node
{
    int info;
    struct node *left,*right;
};
struct node *newnode(int element)
{
    struct node *temp=(struct node *)malloc(sizeof(struct node));
    temp->info=element;
    temp->left=temp->right=NULL;
}
struct node *insert(struct node *root,int key)
{
    if(root==NULL)
        return newnode(key);
    if(key<root->info)
        root->left=insert(root->left,key);
    if(key>root->info)
        root->right=insert(root->right,key);
    return root;
}
```

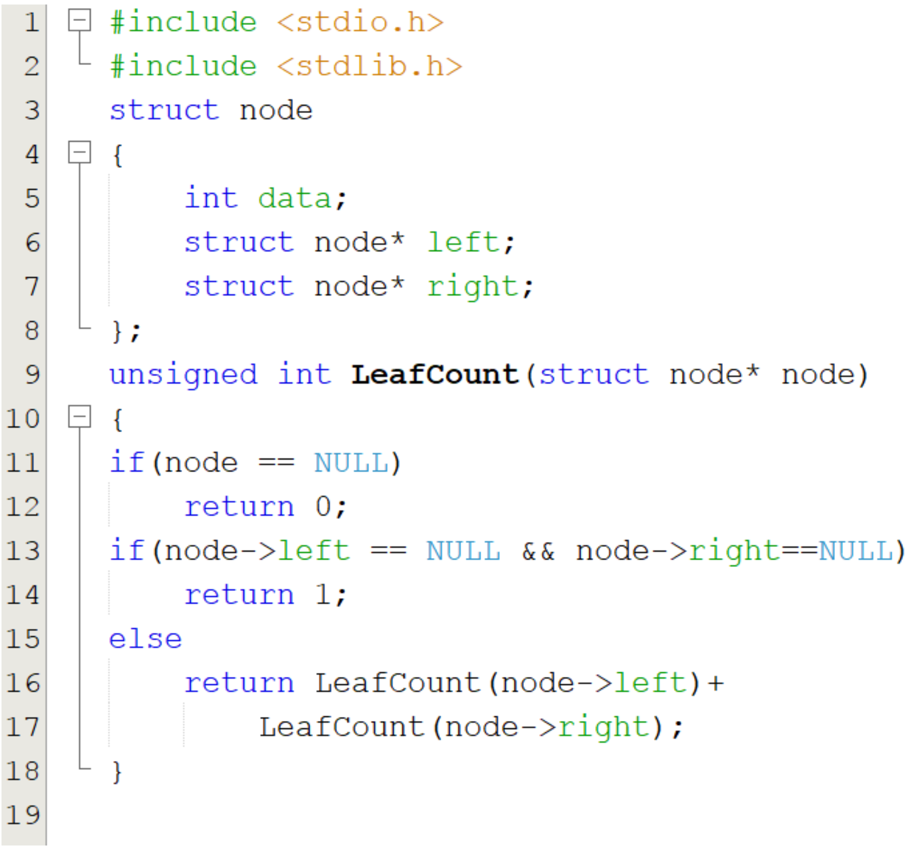


```
void inorder(struct node *root)
{
    if(root==NULL)return;
    inorder(root->left);
    printf(" %d",root->info);
    inorder(root->right);
}
void postorder(struct node *root)
{
    if(root==NULL)return;
    postorder(root->left);
    postorder(root->right);
    printf(" %d",root->info);
}
void preorder(struct node *root)
{
    if(root==NULL)return;
    printf(" %d",root->info);
    preorder(root->left);
    preorder(root->right);
}
```

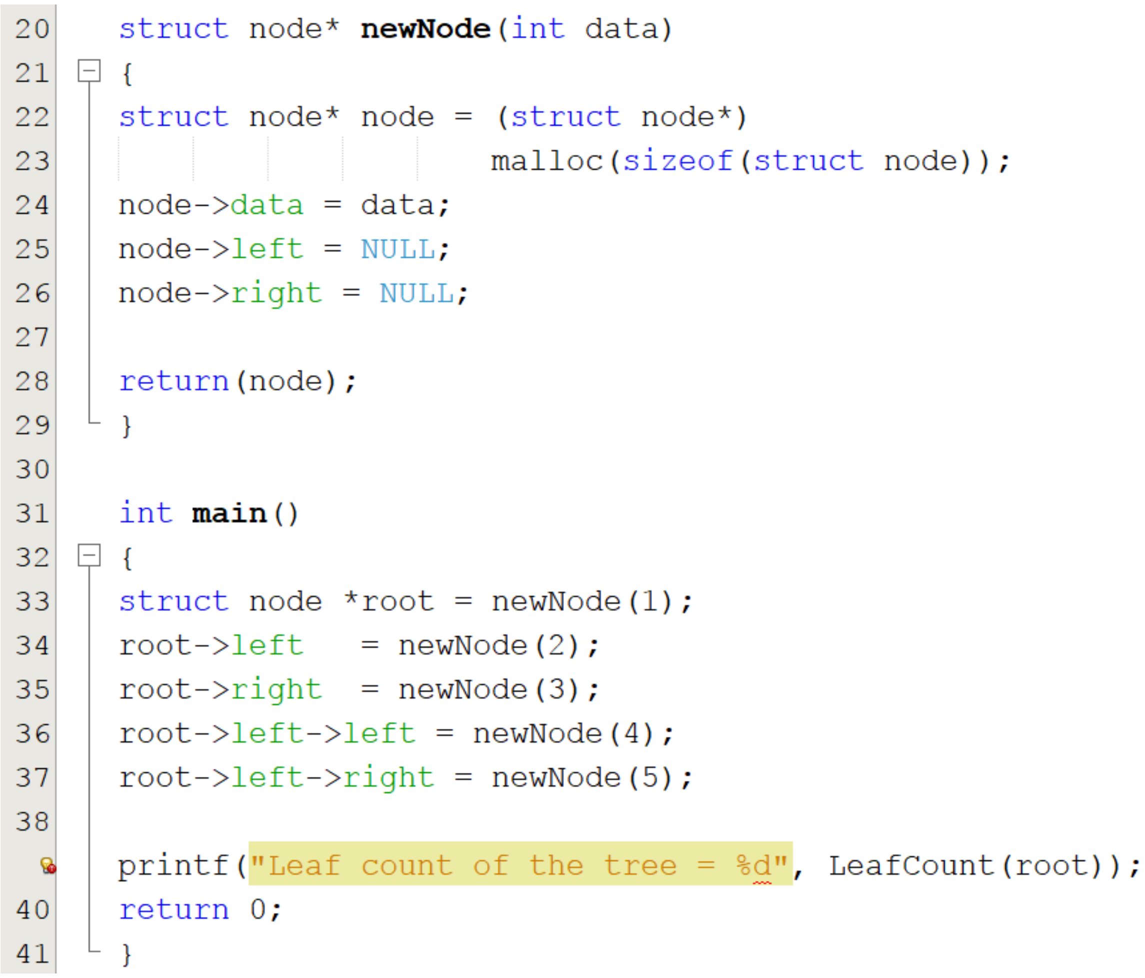


```
int main(int argc, char** argv) {
    struct node *root=NULL;
     int a[7]={10,9,11,4,5,6,8};
    int i;
    root=insert(root,a[0]);
    for(i=1;i<7;i++)
    {
        insert(root,a[i]);
    }
    printf("preorder= ");
    preorder(root);
    printf("\n");
    printf("postorder= ");
    postorder(root);
    printf("\n");
    printf("inorder= ");
    inorder(root);
    return (EXIT_SUCCESS);
}
```

Q2) to implement a linked list to construct a tree and count the number of leaves in a tree:-



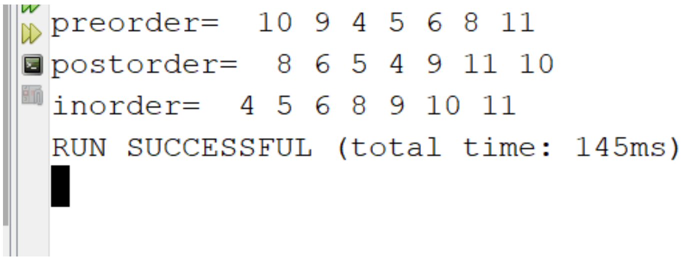
```c
#include <stdio.h>
#include <stdlib.h>
struct node
{
    int data;
    struct node* left;
    struct node* right;
};
unsigned int LeafCount(struct node* node)
{
if(node == NULL)
    return 0;
if(node->left == NULL && node->right==NULL)
    return 1;
else
    return LeafCount(node->left)+
        LeafCount(node->right);
}

```



```c
struct node* newNode(int data)
{
struct node* node = (struct node*)
                    malloc(sizeof(struct node));
node->data = data;
node->left = NULL;
node->right = NULL;

return(node);
}

int main()
{
struct node *root = newNode(1);
root->left    = newNode(2);
root->right   = newNode(3);
root->left->left = newNode(4);
root->left->right = newNode(5);

printf("Leaf count of the tree = %d", LeafCount(root));
return 0;
}
```

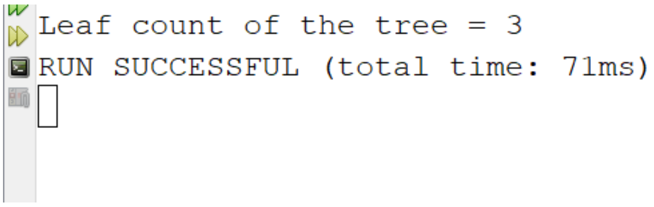
3. Presentation of Results

Q1:-



```
preorder=  10 9 4 5 6 8 11
postorder=  8 6 5 4 9 11 10
inorder=  4 5 6 8 9 10 11
RUN SUCCESSFUL (total time: 145ms)
```

Q2:-



```
Leaf count of the tree = 3
RUN SUCCESSFUL (total time: 71ms)
```

4. Conclusions

All the programs have been executed successfully.

1. We have learned to to construct a binary search tree and perform the Preorder, post order and in order
2. And we have learned to implement a linked list to construct a tree and count the number of leaves in a tree.