```
#include <stdio.h>
#include <conio.h>
#include <malloc.h>
struct node
{
    struct node *left;
    int data;
    struct node *right;
};

void main()
{
    void insert(struct node **,int);
    void inorder(struct node *);
    void postorder(struct node *);
    void preorder(struct node *);

    struct node *ptr;
    int no,i,num;

    ptr = NULL;
    ptr->data=NULL;
    clrscr();

    printf("\nProgram for Tree Traversal\n");
    printf("Enter the number of nodes to add to the tree.<BR>\n");
    scanf("%d",&no);

    for(i=0;i<no;i++)
    {
        printf("Enter the item\n");
        scanf("%d",&num);
        insert(&ptr,num);
    }

    //getch();
    printf("\nINORDER TRAVERSAL\n");
    inorder(ptr);

    printf("\nPREORDER TRAVERSAL\n");
    preorder(ptr);

    printf("\nPOSTORDER TRAVERSAL\n");
    postorder(ptr);

    getch();
}

void insert(struct node **p,int num)
{
    if((*p)==NULL)
    {
        printf("Leaf node created.");
        (*p)=malloc(sizeof(struct node));
        (*p)->left = NULL;
        (*p)->right = NULL;
```

```
        (*p)->data = num;
        return;
    }
    else
    {
        if(num==(*p)->data)
        {
            printf("\nREPEATED ENTRY ERROR VALUE REJECTED\n");
            return;
        }
        if(num<(*p)->data)
        {
            printf("\nDirected to left link.\n");
            insert(&((*p)->left),num);
        }
        else
        {
            printf("Directed to right link.\n");
            insert(&((*p)->right),num);
        }
    }
    return;
}

void inorder(struct node *p)
{
    if(p!=NULL)
    {
        inorder(p->left);
        printf("\nData :%d",p->data);
        inorder(p->right);
    }
    else
    return;
}

void preorder(struct node *p)
{
    if(p!=NULL)
    {
        printf("\nData :%d",p->data);
        preorder(p->left);
        preorder(p->right);
    }
    else
    return;
}

void postorder(struct node *p)
{
    if(p!=NULL)
    {
        postorder(p->left);
        postorder(p->right);
        printf("\nData :%d",p->data);
    }
    else
```

```
    return;
}
```