```
#include<stdio.h>
#include<conio.h>
struct node
{
int info;
struct node*llink,*rlink;
};
typedef struct node*NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
{
printf("\n Out of memory");
exit(0);
}
return x;
}
NODE insert(int item,NODE root)
{
NODE tep,cur,prev;
char d[20];
int i;
temp=getnode();
temp->info=item;
temp->llink=temp->rlink=NULL;
if(root==NULL)
return temp;
printf("\nGive the direction Where you want to insert\n");
scanf("%s",&d);
prev=NULL;
cur=root;
toupper(d);
for(i=0;i<strlen(d);i++)
{
if(cur==NULL)
break;
prev=cur;
if(d[i]=='L')
cur=cur->llink;
else
cur=cur->rlink;
}
if(cur!=NULL||i!=strlen(d))
{
printf("\n insertation not possible");
free(temp);
return root;
}
if(d[i-1]=='L'])
prev->llink=temp;
else
prev->rlink=temp;
```

```
return root;
}
void display(NODE root,int level)
{
int i;
if(root==NULl)
return ;
display(root->rlink,level+1);
for(i=0;i<=level;i++)
printf(" ");
printf("%d\n",root->info);
display(root->llink,level+1);
}
void preorder(NODE root)
{
if(root==NULL)
return ;
printf("%D\t",root->info);
preorder(root->llink);
preorder(root->rlink);
}
void postorder(NODE root)
{
if(root==NULL)
return;
postorder(root->llink);
postorder(root->rlink);
printf("%D\t",root->info);
}
void inorder(NODE root)
{
if(root==NULL)
return ;
inorder(root->llink);
printf("%d\t",root->info);
inorder(root->rlink);
}
void main()
{
NODE root=NULL;
int item,ch;
clrscr();
for(;;)
{
printf("\n1.Insert\t2.preorder\n");
printf("3,inorder\t4.postorder\t5.Exit\n");
printf("\n Enter your choice");
scanf("%d",&ch);
switch(ch)
{
case 1:printf("\n Enter the item to be inserted:");
scanf("%d",&item);
root=insert(item,root);
break;
```

```
case 2:
if(root==NULL)
printf("\n Tree is empty");
else
{
printf("\n The given tree is:\n");
display(root,1);
printf("\n preorder traversal is:\n");
preorder(root);
printf("\n");
}
break;
case 3:
if(root==NULL)
printf("\n Tree is empty");
else
{
printf("\n The given tree is:\n");
display(root,1);
printf("\n Inorder traversal is :\n");
inorder(root);
printf("\n");
}
break;
case 4:
if(root==NULL)
printf("\n tree is empty");
else
{
printf("\n The given tree is\n")
dispaly(root,1);
printf("\n Postorder traversal is :\n");
postorder(root);
printf("\n");
}
break;
case 5: exit(0);
break;
default : printf("Wrong choice\n");
}
}
}
```