```
/* tree creation and traversals using recursion*/

#include <stdio.h>
struct node
{
struct node *left,*right;
int info;
}*root=NULL;
main()
 {
struct node *nptr,*tmp;
char wish='y';
while(wish=='y')
 {
nptr=(struct node *)malloc(sizeof(struct node));
printf("enter the item:");
scanf("%d",&nptr->info);
nptr->left=nptr->right=NULL;
tmp=root;
if(tmp==NULL)
root=tmp=nptr;
else
  {
while(nptr->info>tmp->info||nptr->info<tmp->info)
  {
while(nptr->info>tmp->info)
   {
if(tmp->right==NULL)
    {
          tmp->right=nptr;
          break;
    }
else
tmp=tmp->right;
  }
while(nptr->info<tmp->info)
  {
if(tmp->left==NULL)
  {
tmp->left=nptr;
break;
  }
else
tmp=tmp->left;
  }
 }
 }
printf("want to enter more");
scanf(" %c",&wish);
 }
```

```
printf("\n inorder traversal        :");
travinorder(root);
printf("\n preorder traversal      :");
travpreorder(root);
printf("\n postordertravrsal       :");
travpostorder(root);
getch();
 }

travinorder(struct node *tmp)
 {
if(tmp!=NULL)
 {
travinorder(tmp->left);
printf("%d\t",tmp->info);
travinorder(tmp->right);
  }
 }

travpreorder(struct node *tmp)
 {
if(tmp!=NULL)
  {
printf("%d\t",tmp->info);
travpreorder(tmp->left);
travpreorder(tmp->right);
  }
 }

travpostorder(struct node *tmp)
 {
if(tmp!=NULL)
  {
travpostorder(tmp->left);
travpostorder(tmp->right);
printf("%d\t",tmp->info);
   }
  }


/* tree creation and non recursive traversals*/

#include <stdio.h>
struct node
{
struct node *left,*right;
int info;
}*root=NULL,*stack[100],*rstack[100];
int top=0;
main()
 {
```

```
struct node *nptr,*tmp;
char wish='y';
while(wish=='y')
 {
nptr=(struct node *)malloc(sizeof(struct node));
printf("enter the item:");
scanf("%d",&nptr->info);
nptr->left=nptr->right=NULL;
tmp=root;
if(tmp==NULL)
root=tmp=nptr;
else
  {
while(nptr->info>tmp->info||nptr->info<tmp->info)
  {
while(nptr->info>tmp->info)
   {
if(tmp->right==NULL)
    {
          tmp->right=nptr;
          break;
    }
else
tmp=tmp->right;
  }
while(nptr->info<tmp->info)
  {
if(tmp->left==NULL)
  {
tmp->left=nptr;
break;
  }
else
tmp=tmp->left;
  }
 }
 }
 printf("want to enter more");
 scanf(" %c",&wish);
 }
 printf("\nPreorder traversal \n");
 stack[top]=NULL;
 tmp=stack[++top]=root;
 while(stack[top]!=NULL)
 {
 tmp=stack[top--];
 if (tmp!=NULL)
  printf("\t%d",tmp->info);
 if (tmp->right!=NULL)
  stack[++top]=tmp->right;
 if (tmp->left!=NULL)
```

```
 stack[++top]=tmp->left;
}
printf("\nInorder traversal \n");
tmp=root;
top=0;
while(top>=0)
{
while(tmp!=NULL)
{  stack[++top]=tmp;
   tmp=tmp->left;
}
tmp=stack[top--];
if (tmp!=NULL)
 printf("\t%d",tmp->info);
tmp=tmp->right;
}
printf("\nPostorder traversal \n");
top=-1;
tmp=root;
travleft:
 while(tmp!=NULL)
 {
   stack[++top]=tmp;
   if (tmp->right!=NULL)
   {
    stack[++top]=-1;
    rstack[top]=tmp->right;
   }
   tmp=tmp->left;
 }
 tmp=stack[top--];
 while((int)stack[top+1]>0)
 {
  printf("\t%d",tmp->info);
  tmp=stack[top--];
 }
 if ((int)stack[top+1]==-1)
 {
  tmp= rstack[top+1];
  goto travleft;
 }
getch();
}
```