```
/**** Binary Tree Traversal ****/

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

struct node *root = NULL;
int count = 0;
struct node* insert(struct node *root,int data)
{
  struct node *newnode;
  newnode = (struct node*)malloc(sizeof(struct node));
  newnode->data = data;
  if(root == NULL)
  {
    newnode->left = newnode->right = NULL;
    root = newnode;
    count++;
  }
  else
  {
    if(count%2 != 0)
          root->left = insert(root->left,data);
    else
          root->right = insert(root->right,data);
  }
  return root;
}
void inorder(struct node *root)
{
  if(root != NULL)
  {
    inorder(root->left);
    printf("%d\t",root->data);
    inorder(root->right);
  }
}
void preorder(struct node *root)
{
  if(root != NULL)
  {
    printf("%d\t",root->data);
    preorder(root->left);
    preorder(root->right);
```

```
  }
}
void postorder(struct node *root)
{
  if(root != NULL)
  {
    postorder(root->left);
    postorder(root->right);
    printf("%d\t",root->data);
  }
}
void main()
{
  int opt, data;
  clrscr();
  while(1){
    printf("\n1.Insert\n2.Inorder\n3.Preorder\n4.Postorder\n5.Exit\n");
    printf("Enter your option: ");
    scanf("%d",&opt);
    switch(opt)
    {
          case 1:
            printf("Enter the value to be insert: ");
            scanf("%d", &data);
            root = insert(root,data);
            break;
          case 2:
            if (root==NULL)
              printf("Tree is empty\n");
            else
              inorder(root);
            break;
          case 3:
            if (root==NULL)
              printf("Tree is empty\n");
            else
              preorder(root);
            break;
          case 4:
            if (root==NULL)
              printf("Tree is empty\n");
            else
              postorder(root);
            break;
          case 5:
            exit(0);
          default:
            printf("Invalid Option\n");
```

```
    }
  }
}
```