```
#include<iostream>
using namespace std;
//binary tree node declaration
struct bintree_node{
   bintree_node *left;
   bintree_node *right;
   char data;
} ;
class bintree_class{
   bintree_node *root;
   public:
   bintree_class(){
      root=NULL;
   }
   int isempty() {
      return(root==NULL);
   }
   void insert_node(int item);
   void inorder_seq();
   void inorder(bintree_node *);
   void postorder_seq();
   void postorder(bintree_node *);
   void preorder_seq();
   void preorder(bintree_node *);
};
void bintree_class::insert_node(int item){
   bintree_node *p=new bintree_node;
   bintree_node *parent;
   p->data=item;
   p->left=NULL;
   p->right=NULL;
```

```
   parent=NULL;
   if(isempty())
      root=p;
   else{
      bintree_node *ptr;
      ptr=root;
      while(ptr!=NULL)       {
         parent=ptr;
         if(item>ptr->data)
            ptr=ptr->right;
         else
            ptr=ptr->left;
      }
      if(item<parent->data)
         parent->left=p;
      else
         parent->right=p;
   }
}
void bintree_class::inorder_seq()
{
   inorder(root);
}
void bintree_class::inorder(bintree_node *ptr)
{
   if(ptr!=NULL){
      inorder(ptr->left);
      cout<<"  "<<ptr->data<<"     ";
      inorder(ptr->right);
   }
}
```

```
void bintree_class::postorder_seq()
{
   postorder(root);
}
void bintree_class::postorder(bintree_node *ptr)
{
   if(ptr!=NULL){
      postorder(ptr->left);
      postorder(ptr->right);
      cout<<"  "<<ptr->data<<"    ";
   }
}
void bintree_class::preorder_seq()
{
   preorder(root);
}
void bintree_class::preorder(bintree_node *ptr)
{
   if(ptr!=NULL){
      cout<<"  "<<ptr->data<<"    ";
      preorder(ptr->left);
      preorder(ptr->right);
   }
}
int main()
{
   bintree_class bintree;
   bintree.insert_node('A');
   bintree.insert_node('B');
   bintree.insert_node('C');
   bintree.insert_node('D');
```

```
    bintree.insert_node('E');
    bintree.insert_node('F');
    bintree.insert_node('G');
    cout<<"Inorder traversal:"<<endl;
    bintree.inorder_seq();
    cout<<endl<<"Postorder traversal:"<<endl;
    bintree.postorder_seq();
    cout<<endl<<"Preorder traversal:"<<endl;
    bintree.preorder_seq();
}
```