```
#include <iostream>
#include<iomanip>
using namespace std;

class Node
{
public :
    Node *left;
    int value;
    Node *right;
    Node(int value)
    {
        left=nullptr;
        this->value=value;
        right=nullptr;
    }
};

class Binary
{
public:
    Node *root ;

    Binary()
    {
    root=nullptr;
    }

Node *addnode(Node *node, int value)
{
    if(nullptr == node)
        return new Node(value);

    if(value > node->value)
        node->right = addnode(node->right,value);

    else
        node->left = addnode(node->left,value);
    return node;
}

void printinorder( Node *temp)
{
    if(temp)
    {
    printinorder(temp->left);
    cout<<temp->value<<endl;
    printinorder(temp->right);
    }
}

void printpreorder(Node *temp)
{
    if(temp)
    {
    cout<<temp->value<<endl;
    printpreorder(temp->left);
    printpreorder(temp->right);
    }
}

void postorder(Node *temp)
    {
        if(temp)
        {
        postorder(temp->left);
```

```
67          postorder(temp->right);
68          cout<<temp->value<<endl;
69          }
70      }
71
72  Node *removenode(Node *node, int value)
73  {
74      if(value > node->value)
75      {
76          node->right=removenode(node->right,value);
77      }
78      else if(value < node->value)
79      {
80          node->left=removenode(node->left,value);
81      }
82
83      else
84      {
85      if(nullptr==node->left && nullptr==node->right)
86      {
87          delete node;
88          return nullptr;
89      }
90
91
92      if(nullptr==node->left && nullptr!=node->right)
93      {
94          Node *ar = node->right;
95          delete node;
96          return ar;
97      }
98      if(nullptr==node->right && nullptr!=node->left)
99      {
100         Node *ar =node->left;
101         delete node;
102         return ar;
103     }
104
105     Node * suc= node->right;
106     while(suc->left!=nullptr)
107     {
108         suc=suc->left;
109         node->value=suc->value;
110         node->right=removenode(node->right,suc->value);
111     }
112
113     }
114     return node;
115 }
116
117 void print(Node * node)
118 {
119     static int level=0;
120     if(node)
121     {
122         level++;
123         print(node->right);
124         cout<<setw(level * 4)<<""<<node->value<<endl;
125         print(node->left);
126         level--;
127     }
128 }
129
130 };
131
132 int main()
```

```
{
    Binary b;
    int num,num1;
    while(cout<<"Enter number in binary tree or 0 to exit: ",cin>>num,num!=0)
    {
        b.root=b.addnode(b.root,num);
        b.print(b.root);

    }


    while(cout<<"Enter 1:preorder 2:postorder 3:inorder or 0 to exit: ",cin>>num1,num1!=0)

    {
        if(num1==1)
        b.printpreorder(b.root);

        if(num1==2)
        b.postorder(b.root);

        if(num1==3)
            b.printinorder(b.root);
    }
    while(cout<<"Enter number delete or 0 to exit: ",cin>>num,num!=0)
    {
        b.root=b.removenode(b.root,num);
        b.print(b.root);
    }
        return 0;
}
```