

```

#include<iostream>

using namespace std;
struct btree {
    int data;
    btree *right;
    btree *left;
};
btree *ekle(btree *t, int x) {
    if (t == NULL)
    {
        t = new btree;
        t->data = x;
        t->right = NULL;
        t->left = NULL;
    }
    else
    {
        if (t->data < x)
            t->right = ekle(t->right, x);
        else
            t->left = ekle(t->left, x);
    }
    return t;
}
int elemansay(btree *t) {
    if (t == NULL)
        return 0;
    else
        return elemansay(t->right) + 1 + elemansay(t->left);
}
int maximum(btree *t) {
    if (t == NULL)
        return 0;
    else
        while (t->right != NULL)
            t = t->right;
    return t->data;
}
int minimum(btree *t) {
    if (t == NULL)
        return 0;
    else
        while (t->left != NULL)
            t = t->left;
    return t->data;
}
btree *arama(btree *t, int x) {
    if (t != NULL)
    {
        if (t->data == x)
            return t;
        else
        {
            if (t->data < x)
                arama(t->right, x);
            else
                arama(t->left, x);
        }
    }
    else
        return NULL;
}

```

```

btree *del(btree *t, int x) {
    if (t == NULL)
        return NULL;
    if (t->data == x)
    {
        if (t->right == NULL && t->left == NULL)
            return NULL;
        if (t->left == NULL)
        {
            t->data = minimum(t->right);
            t->right = del(t->right, t->data);
            return t;
        }
        t->data = maximum(t->left);
        t->left = del(t->left, t->data);
        return t;
    }
    if (t->data < x)
    {
        t->right = del(t->right, x);
        return t;
    }
    t->left = del(t->left, x);
    return t;
}

btree *yazdir(btree *t) {
    if (t == NULL)
        return 0;
    else
    {
        cout << t->data << " ";
        yazdir(t->left);
        yazdir(t->right);
    }
    return 0;
}

```

```
void main() {
    btree *tree;
    tree = new btree;
    tree = NULL;
    tree = ekle(tree, 12);
    tree = ekle(tree, 20);
    ekle(tree, 30);
    ekle(tree, 9);
    ekle(tree, 15);
    ekle(tree, 8);
    ekle(tree, 10);
    ekle(tree, 45);
    ekle(tree, 5);
    ekle(tree, 22);
    ekle(tree, 11);
    yazdir(tree);
    cout << "\neleman sayisi:\t" << elemansay(tree);
    cout << "\nmaksimum eleman:" << maximum(tree);
    cout << "\nminimum eleman:\t" << minimum(tree);
    int x = 12;
    btree *ar;
    ar = arama(tree, x);
    if (ar == NULL)
        cout << "\nsonuc yok";
    else
        cout << "\nsonuc var " << ar->data;
    del(tree, x);
    cout << endl;
    yazdir(tree);
    cout << endl;
    system("pause");
}
```