```c
#include<stdlib.h>
#include<stdio.h>

struct bin_tree {
int data;
struct bin_tree * right, * left;
};
typedef struct bin_tree node;

void insert(node ** tree, int val)
{
    node *temp = NULL;
    if(!(*tree))
    {
        temp = (node *)malloc(sizeof(node));
        temp->left = temp->right = NULL;
        temp->data = val;
        *tree = temp;
        return;
    }

    if(val < (*tree)->data)
    {
        insert(&(*tree)->left, val);
    }
    else if(val > (*tree)->data)
    {
        insert(&(*tree)->right, val);
    }

}

void print_preorder(node * tree)
{
    if (tree)
    {
        printf("%d\n",tree->data);
        print_preorder(tree->left);
        print_preorder(tree->right);
    }

}

void print_inorder(node * tree)
{
    if (tree)
    {
        print_inorder(tree->left);
        printf("%d\n",tree->data);
        print_inorder(tree->right);
    }
}

void print_postorder(node * tree)
{
    if (tree)
    {
        print_postorder(tree->left);
        print_postorder(tree->right);
        printf("%d\n",tree->data);
    }
}

void deltree(node * tree)
{
    if (tree)
    {
        deltree(tree->left);
        deltree(tree->right);
        free(tree);
    }
}

node* search(node ** tree, int val)
{
    if(!(*tree))
    {
        return NULL;
    }

    if(val < (*tree)->data)
    {
        search(&((*tree)->left), val);
    }
    else if(val > (*tree)->data)
    {
        search(&((*tree)->right), val);
    }
    else if(val == (*tree)->data)
    {
        return *tree;
    }
}

void main()
{
    node *root;
    node *tmp;
    //int i;

```

```
101         root = NULL;
102         /* Inserting nodes into tree */
103         insert(&root, 9);
104         insert(&root, 4);
105         insert(&root, 15);
106         insert(&root, 6);
107         insert(&root, 12);
108         insert(&root, 17);
109         insert(&root, 2);
110
111         /* Printing nodes of tree */
112         printf("Pre Order Display\n");
113         print_preorder(root);
114
115         printf("In Order Display\n");
116         print_inorder(root);
117
118         printf("Post Order Display\n");
119         print_postorder(root);
120
121         /* Search node into tree */
122         tmp = search(&root, 4);
123         if (tmp)
124         {
125             printf("Searched node=%d\n", tmp->data);
126         }
127         else
128         {
129             printf("Data Not found in tree.\n");
130         }
131
132         /* Deleting all nodes of tree */
133         deltree(root);
134     }
135
```

Écrire le programme suivant
Exécutez le programme
Que fait ce programme
Ajouter des méthodes de calcule de :
- la somme des contenus de tous les nœuds,
- le minimum des valeurs contenues.
- le maximum des valeurs contenues