```c
// C program to find maximum and minimum in a Binary Tree
#include <limits.h>
#include <stdio.h>
#include <stdlib.h>

// A tree node
struct Node {
    int data;
    struct Node *left, *right;
};

// A utility function to create a new node
struct Node* newNode(int data)
{
    struct Node* node
        = (struct Node*)malloc(sizeof(struct Node));
    node->data = data;
    node->left = node->right = NULL;
    return (node);
}

// Returns maximum value in a given Binary Tree
int findMax(struct Node* root)
{
    // Base case
    if (root == NULL)
        return INT_MIN;

    // Return maximum of 3 values:
    // 1) Root's data 2) Max in Left Subtree
    // 3) Max in right subtree
    int res = root->data;
    int lres = findMax(root->left);
    int rres = findMax(root->right);
    if (lres > res)
        res = lres;
    if (rres > res)
        res = rres;
    return res;
```

```c
}

// Returns minimum value in a given Binary Tree
int findMin(struct Node* root)
{
    // Base case
    if (root == NULL)
    return INT_MAX;

    // Return minimum of 3 values:
    // 1) Root's data 2) Max in Left Subtree
    // 3) Max in right subtree
    int res = root->data;
    int lres = findMin(root->left);
    int rres = findMin(root->right);
    if (lres < res)
    res = lres;
    if (rres < res)
    res = rres;
    return res;
}


// Driver code
int main(void)
{
    struct Node* NewRoot = NULL;
    struct Node* root = newNode(2);
    root->left = newNode(7);
    root->right = newNode(5);
    root->left->right = newNode(6);
    root->left->right->left = newNode(1);
    root->left->right->right = newNode(11);
    root->right->right = newNode(9);
    root->right->right->left = newNode(4);

    // Function call
    printf("Maximum element is %d \n", findMax(root));
    printf("Minimum element is %d \n", findMin(root));

    return 0;
}

```