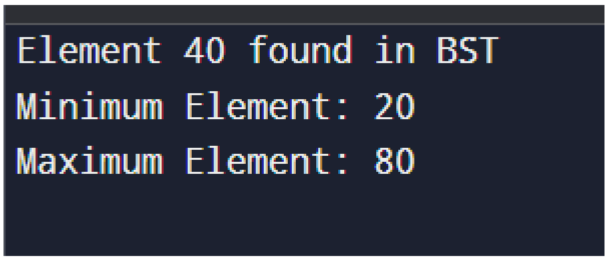
Code:

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

typedef struct Node {
    int data;
    struct Node *left, *right;
} Node;

// Create a new node
Node* newNode(int data) {
    Node* temp = (Node*)malloc(sizeof(Node));
    temp->data = data;
    temp->left = temp->right = NULL;
    return temp;
}

// Insert into BST
Node* insert(Node* root, int data) {
    if (root == NULL)
        return newNode(data);

    if (data < root->data)
        root->left = insert(root->left, data);
    else
        root->right = insert(root->right, data);

    return root;
}
```

```
// Search a value
Node* search(Node* root, int key) {
    if (root == NULL || root->data == key)
        return root;

    if (key < root->data)
        return search(root->left, key);

    return search(root->right, key);
}

// Find Minimum
Node* findMin(Node* root) {
    while (root && root->left != NULL)
        root = root->left;
    return root;
}

// Find Maximum
Node* findMax(Node* root) {
    while (root && root->right != NULL)
        root = root->right;
    return root;
}

int main() {
    Node* root = NULL;

    // Insert elements
    root = insert(root, 50);
    root = insert(root, 30);
    root = insert(root, 20);
    root = insert(root, 40);
    root = insert(root, 70);
    root = insert(root, 60);
    root = insert(root, 80);

    int key = 40;

    Node* s = search(root, key);

    if (s != NULL)
        printf("Element %d found in BST\n", key);
    else
        printf("Element %d NOT found in BST\n", key);

    printf("Minimum Element: %d\n", findMin(root)->data);
    printf("Maximum Element: %d\n", findMax(root)->data);

    return 0;
}
```

Output:



```
Element 40 found in BST
Minimum Element: 20
Maximum Element: 80
```