```cpp
// Question-2.cpp : This file contains the 'main' function. Program execution begins and ends there.
//// Question1.cpp : This file contains the 'main' function. Program execution begins and ends there.
////////// /
//------------------------------------------------------------------------------------------
//Name                          Sai Chaitanya Kilambi
//Course                        CPSC 131 Data Structures, Fall, 2022
//Assignment                    No.11 question:2
//Due date                      11/30/2022
// Purpose:
// This program demonstrates insertion of data in array into a Binary Search Tree along with inorder traversal.
// It also demonstrates how to search display the min and max
//------------------------------------------------------------------------------------------
// list of libraries
//
//importing the required libraries

#include<iostream>
#include<climits>
using namespace std;
typedef struct node {
    int value;
    node* pLeft;
    node* pRight;
    node(int val = 0)
    {
        value = val;
        pRight = NULL;
        pLeft = NULL;
    }
}node;
void insert(node** pRoot, int val) {
    if (*pRoot == NULL)
        *pRoot = new node(val);
    else if ((*pRoot)->value <= val)
        insert(&((*pRoot)->pRight), val);
    else if ((*pRoot)->value > val)
        insert(&((*pRoot)->pLeft), val);
}
node* getBST(int* arr, int size) {
    node* pRoot = NULL;
    for (int i = 0; i < size; i++)
        insert(&pRoot, arr[i]);
    return pRoot;
}
```

```cpp
void inOrderTraversal(node* pRoot) {
    if (pRoot && pRoot->pLeft)
        inOrderTraversal(pRoot->pLeft);
    if (pRoot)
        std::cout << pRoot->value << " ";
    if (pRoot && pRoot->pRight)
        inOrderTraversal(pRoot->pRight);

}
int findMin(node* root)
{
    // Base case
    if (root == NULL)
        return INT_MAX;
    int res = root->value;
    int lres = findMin(root->pLeft);
    int rres = findMin(root->pRight);
    if (lres < res)
        res = lres;
    if (rres < res)
        res = rres;
    return res;
}
int findMax(node* root)
{
    if (root == NULL)
        return INT_MIN;
    int res = root->value;
    int lres = findMax(root->pLeft);
    int rres = findMax(root->pRight);
    if (lres > res)
        res = lres;
    if (rres > res)
        res = rres;
    return res;
}
int main() {
    int arr[12];
    srand(time(0));
    cout << "Array elements are: ";
    for (int i = 0; i < 12; i++) {
        arr[i] = rand() % 100;
        cout << arr[i] << " ";
    }
    cout << endl;
    node* pRoot = getBST(arr, sizeof(arr) / sizeof(int));
    cout << "Inorder traversal of tree is: ";
    inOrderTraversal(pRoot);
    cout << endl;
```

```
93      cout << "Maximum element is BST is " << findMax(pRoot) << endl;
94      cout << "Minimum element is BST is " << findMin(pRoot) << endl;
95      return 0;
96  }
```