

# Rajalakshmi Engineering College

Name: NITHISH RAJ L  
Email: 240701366@rajalakshmi.edu.in  
Roll no: 2116240701366  
Phone: 8072719523  
Branch: REC  
Department: I CSE FD  
Batch: 2028  
Degree: B.E - CSE

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 5\_COD\_Question 4

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

John, a computer science student, is learning about binary search trees (BST) and their properties. He decides to write a program to create a BST, display it in post-order traversal, and find the minimum value present in the tree.

Help him by implementing the program.

##### ***Input Format***

The first line of input consists of an integer N, representing the number of elements to insert into the BST.

The second line consists of N space-separated integers data, which is the data to be inserted into the BST.

### **Output Format**

The first line of output prints the space-separated elements of the BST in post-order traversal.

The second line prints the minimum value found in the BST.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 3

5 10 15

Output: 15 10 5

The minimum value in the BST is: 5

### **Answer**

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

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

```
struct Node {  
    int data;  
    struct Node* left;  
    struct Node* right;  
};
```

```
struct Node* createNode(int data) {  
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));  
    newNode->data = data;  
    newNode->left = newNode->right = NULL;  
    return newNode;  
}
```

```
// You are using GCC
```

```
struct Node* insert(struct Node* root, int data) {  
    if(root==NULL)  
    {  
        struct Node* newnode=(struct Node*)malloc(sizeof(struct Node));  
        newnode->data=data;  
        newnode->right=newnode->left=NULL;  
        return newnode;  
    }
```

```

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

```

```

void displayTreePostOrder(struct Node* root) {
    if(root!=NULL)
    {
        displayTreePostOrder(root->left);
        displayTreePostOrder(root->right);
        printf("%d ",root->data);
    }
}

```

```

int findMinValue(struct Node* root) {
    if(root==NULL)
    {
        return -1;
    }
    while(root->left!=NULL)
    {
        root=root->left;
    }
    return root->data;
}

```

```

int main() {
    struct Node* root = NULL;
    int n, data;
    scanf("%d", &n);

    for (int i = 0; i < n; i++) {
        scanf("%d", &data);
        root = insert(root, data);
    }
}

```

```
displayTreePostOrder(root);  
printf("\n");  
  
int minValue = findMinValue(root);  
printf("The minimum value in the BST is: %d", minValue);  
  
return 0;  
}
```

**Status :** Correct

**Marks : 10/10**