

<b>Started on</b>	Monday, 4 March 2024, 7:48 AM
<b>State</b>	Finished
<b>Completed on</b>	Monday, 4 March 2024, 8:48 AM
<b>Time taken</b>	1 hour
<b>Marks</b>	20.00/20.00
<b>Grade</b>	<b>10.00</b> out of 10.00 ( <b>100%</b> )

Question 1

Correct

Mark 10.00 out of 10.00

You are given a pointer to the root of a binary search tree and values to be inserted into the tree. Insert the values into their appropriate position in the binary search tree and return the root of the updated binary tree. You just have to complete the function.

You are given a function,

```
Node * insert (Node * root ,int data) {  
  
}
```

Input Format

- First line of the input contains  $t$ , the number of nodes in the tree.
- Second line of the input contains the list of  $t$  elements to be inserted to the tree.

Constraints

- No. of nodes in the tree,  $1 \leq t \leq 5000$
- Value of each node in the tree,  $1 \leq t[i] \leq 10000$

Output Format

Return the items in the binary search tree after inserting the values into the tree. Start with the root and follow each element by its left subtree, and then its right subtree.

Sample Input

```
6  
4 2 3 1 7 6
```

Sample Output

```
4 2 1 3 7 6
```

Sample Explanation

The binary tree after inserting the 6 elements in the given order will look like this.



For example:

Input	Result
6 4 2 3 1 7 6	4 2 1 3 7 6
19 44 67 91 20 87 20 31 11 19 39 86 65 57 84 10 72 84 15 46	44 20 11 10 19 15 20 31 39 67 65 57 46 91 87 86 84 72 84

Answer: (penalty regime: 0 %)

Reset answer

```
1 #include <bits/stdc++.h>  
2  
3 using namespace std;  
4  
5 class Node {  
6     public:  
7         int data;  
8         Node *left;  
9         Node *right;
```

```

10 Node(int d) {
11     data = d;
12     left = NULL;
13     right = NULL;
14 }
15 };
16
17 class Solution {
18     public:
19
20     void preOrder(Node *root) {
21
22         if( root == NULL )
23             return;
24
25         std::cout << root->data << " ";
26
27         preOrder(root->left);
28         preOrder(root->right);
29     }
30
31     /*
32     Node is defined as
33
34     class Node {
35     public:
36         int data;
37         Node *left;
38         Node *right;
39         Node(int d) {
40             data = d;
41             left = NULL;
42             right = NULL;
43         }
44     };
45
46     */
47
48     Node* insert(Node* root, int data) {
49         if (root == NULL) {
50             root = new Node(data);
51         } else if (data < root->data) {
52             root->left = insert(root->left, data);

```

	Input	Expected	Got	
✓	6 4 2 3 1 7 6	4 2 1 3 7 6	4 2 1 3 7 6	✓
✓	19 44 67 91 20 87 20 31 11 19 39 86 65 57 84 10 72 84 15 46	44 20 11 10 19 15 20 31 39 67 65 57 46 91 87 86 84 72 84	44 20 11 10 19 15 20 31 39 67 65 57 46 91 87 86 84 72 84	✓

Passed all tests! ✓

► [Show/hide question author's solution \(Cpp\)](#)

Correct

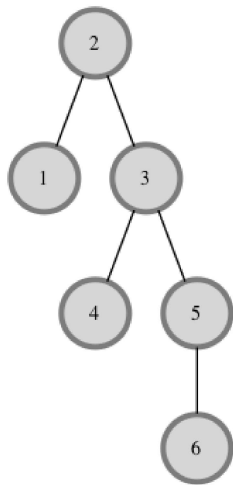
Marks for this submission: 10.00/10.00.

## Question 2

Correct

Mark 10.00 out of 10.00

You are given pointer to the root of the binary search tree and two values ***v1*** and ***v2***. You need to return the lowest common ancestor (LCA) of ***v1*** and ***v2*** in the binary search tree.



In the diagram above, the lowest common ancestor of the nodes **4** and **6** is the node **3**. Node **3** is the lowest node which has nodes **4** and **6** as descendants.

### Function Description

Complete the function `lca` in the editor below. It should return a pointer to the lowest common ancestor node of the two values given.

`lca` has the following parameters:

- `root`: a pointer to the root node of a binary search tree
- `v1`: a `node.data` value
- `v2`: a `node.data` value

### Input Format

The first line contains an integer, ***n***, the number of nodes in the tree.

The second line contains ***n*** space-separated integers representing ***node.data*** values.

The third line contains two space-separated integers, ***v1*** and ***v2***.

To use the test data, you will have to create the binary search tree yourself.

### Constraints

$$1 \leq n, \text{node.data} \leq 5000$$

$$1 \leq v1, v2 \leq 5000$$

$$v1 \neq v2$$

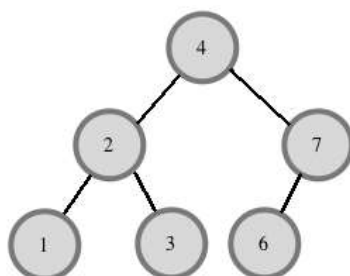
The tree will contain nodes with `data` equal to ***v1*** and ***v2***.

### Output Format

Return the value of the node that is the lowest common ancestor of ***v1*** and ***v2***.

### Sample Input

```
6
4 2 3 1 7 6
1 7
```



$v1 = 1$  and  $v2 = 7$ .

#### Sample Output

4

#### Explanation

LCA of **1** and **7** is **4**, the root in this case.

Return a pointer to the node.

For example:

Input	Result
6 4 2 3 1 7 6 1 7	4
2 1 2 1 2	1

Answer: (penalty regime: 0 %)

```
1 #include <iostream>
2 using namespace std;
3
4 struct TreeNode {
5     int value;
6     TreeNode *left, *right;
7     TreeNode(int x) : value(x), left(nullptr), right(nullptr) {}
8 };
9
10 // Function to insert a node into the binary search tree
11 TreeNode* insert(TreeNode* root, int x) {
12     if (root == nullptr)
13         return new TreeNode(x);
14     if (x < root->value)
15         root->left = insert(root->left, x);
16     else
17         root->right = insert(root->right, x);
18     return root;
19 }
20
21 // Function to find the lowest common ancestor of two nodes in the binary search tree
22 TreeNode* lowestCommonAncestor(TreeNode* root, int value1, int value2) {
23     while (root != nullptr) {
24         if (value1 < root->value && value2 < root->value)
25             root = root->left;
26         else if (value1 > root->value && value2 > root->value)
27             root = root->right;
28         else
29             return root;
30     }
31     return nullptr;
32 }
33
34 int main() {
35     TreeNode* root = nullptr;
36     int n;
37     cin >> n;
38     for (int i = 0; i < n; i++) {
39         int x;
40         cin >> x;
41         root = insert(root, x);
42     }
43     int value1, value2;
44     cin >> value1 >> value2;
45     TreeNode* result = lowestCommonAncestor(root, value1, value2);
46     if (result != nullptr)
47         cout << result->value;
48     return 0;
49 }
```

	Input	Expected	Got	
✓	6 4 2 3 1 7 6 1 7	4	4	✓
✓	2 1 2 1 2	1	1	✓
✓	3 5 3 7 3 7	5	5	✓

Passed all tests! ✓

[▶ Show/hide question author's solution \(Cpp\).](#)

Correct

Marks for this submission: 10.00/10.00.