

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 using namespace std;
5
6 class BST {
7 public:
8     int value;
9     BST *left;
10    BST *right;
11
12    BST(int val);
13 };
14
15 // O(n) time | O(n) space
16 vector<int> inOrderTraverse(BST *tree, vector<int> array) {
17     if (tree->left != NULL) {
18         array = inOrderTraverse(tree->left, array);
19     }
20     array.push_back(tree->value);
21     if (tree->right != NULL) {
22         array = inOrderTraverse(tree->right, array);
23     }
24     return array;
25 }
26
27 // O(n) time | O(n) space
28 vector<int> preOrderTraverse(BST *tree, vector<int> array) {
29     array.push_back(tree->value);
30     if (tree->left != NULL) {
31         array = preOrderTraverse(tree->left, array);
32     }
33     if (tree->right != NULL) {
34         array = preOrderTraverse(tree->right, array);
35     }
36     return array;
37 }
38
39 // O(n) time | O(n) space
40 vector<int> postOrderTraverse(BST *tree, vector<int> array) {
41     if (tree->left != NULL) {
42         array = postOrderTraverse(tree->left, array);
43     }
44     if (tree->right != NULL) {
45         array = postOrderTraverse(tree->right, array);
46     }
47     array.push_back(tree->value);
48     return array;
49 }
50 }
```

Solution 1

Solution 2

Solution 3

```
1 #include <vector>
2 using namespace std;
3
4 class BST {
5 public:
6     int value;
7     BST *left;
8     BST *right;
9
10    BST(int val);
11 };
12
13 vector<int> inOrderTraverse(BST *tree, vector<int> array) {
14     // Write your code here.
15     return {};
16 }
17
18 vector<int> preOrderTraverse(BST *tree, vector<int> array) {
19     // Write your code here.
20     return {};
21 }
22
23 vector<int> postOrderTraverse(BST *tree, vector<int> array) {
24     // Write your code here.
25     return {};
26 }
27 }
```

Run or submit code when you're ready.