

Inorder Traversal 5

Problem

Submissions

Leaderboard

Discussions

Write a java program to perform Inorder tree traversal.

Input Format

1 10 1 20 1 30 1 40 1 50 1 60 2 3

Constraints

No Constraints

Output Format

InorderTraversal is: 10 20 30 40 50 60

Sample Input 0

```
1
10
1
20
1
30
1
40
1
50
1
60
2
3
```

Sample Output 0

```
InorderTraversal is:
10 20 30 40 50 60
```

[f](#) [t](#) [in](#)**Contest ends in 2 months**

Submissions: 96

Max Score: 10

Difficulty: Medium

Rate This Challenge:

☆☆☆☆☆

[More](#)

java 7



```
1 //224G1A0553
2 import java.util.*;
3 class Node {
4     int data;
5     Node left;
6     Node right;
7     public Node( int item) {
8         this.data = item;
9         this.left = null;
```

```
10     this.right = null;
11 }
12 class StackNode {
13     Node node;
14     StackNode next;
15     public void StackNode(Node b) {
16         this.node = b;
17         this.next = null;
18     }
19     public class NonRecursiveInorder {
20         StackNode top;
21         Node root;
22         public void NonRecursiveInorder() {
23             top = null;
24             root = null;
25         }
26         boolean isEmpty() {
27             if(top == null) {
28                 return true;
29             }
30             return false;
31         }
32         void push(Node b) {
33             StackNode temp;
34             temp = new StackNode();
35             if(temp == null) {
36                 System.out.printf("Stack is overflow.\n");
37             } else {
38                 temp.node = b;
39                 temp.next = top;
40                 top = temp;
41             }
42         }
43         Node peek() {
44             if (top == null) {
45                 return null;
46             }
47             return top.node;
48         }
49         Node pop() {
50             StackNode temp;
51             Node b;
52             if(top == null) {
53                 System.out.printf("Stack is underflow.\n");
54                 return null;
55             } else {
56                 temp = top;
57                 top = top.next;
58                 b = temp.node;
59                 return b;
60             }
61         }
62         void inorderInBST(Node root) {
63             Node curr = root;
64             while(true) {
65                 if(curr != null) {
66                     push(curr);
67                     curr = curr.left;
68                 } else {
69                     curr = pop();
70                     System.out.printf("%d ", curr.data);
71                     curr = curr.right;
72                 }
73                 if(isEmpty() && curr == null)
74                     break;
75             }
76         }
77     }
78 }
79 /* Insertion into binary search tree */
80 Node insertBinarySearchTree(Node root, int item) {
```

```
76
77  /* If the tree is empty new node became root */
78  if (root == null) {
79      root = new Node(item);
80      return root;
81  }
82  /* Otherwise, if item is less then root then recur left side */
83  if (item < root.data)
84      root.left = insertBinarySearchTree(root.left, item);
85  else if (item > root.data)
86      root.right = insertBinarySearchTree(root.right, item);
87
88  /* return the root node pointer */
89  return root;
90  }
91  // Driver main method Code
92  public static void main(String[] args) {
93      NonRecursiveInorder tree = new NonRecursiveInorder();
94      Scanner sc = new Scanner(System.in);
95      int option;
96      int item;
97      //System.out.println("Enter 1 to insert\nEnter 2 to display BST in inorder\nEnter 3 to
Exit");
98      while(true) {
99          //System.out.print("Enter your option: ");
100         option = sc.nextInt();
101         switch(option) {
102             default:
103                 System.out.println("Enter the right option");
104                 break;
105             case 1:
106                 //System.out.print("Enter the element to insert: ");
107                 item = sc.nextInt();
108                 tree.root = tree.insertBinarySearchTree(tree.root, item);
109                 break;
110             case 2:
111                 if(tree.root == null) {
112                     System.out.println("Tree is empty, root is null");
113                 }else {
114                     System.out.println("InorderTraversal is:");
115                     tree.inorderInBST(tree.root);
116                     System.out.println();
117                 }
118                 break;
119             case 3:
120                 return;
121         }
122     }
123 }
```

Line: 18 Col: 7

[Upload Code as File](#) ☐ Test against custom input

Run Code

Submit Code

Testcase 0 ✓

Congratulations, you passed the sample test case.Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
1
10
1
```

```
20
1
30
1
40
1
50
1
60
2
3
```

Your Output (stdout)

```
InorderTraversal is:
10 20 30 40 50 60
```

Expected Output

```
InorderTraversal is:
10 20 30 40 50 60
```