Preorder Traversal 4

Write a java program to perform preorder tree order tree traversal

Input Format

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

Constraints

No constraints

Output Format

Preorder Traversal is: 10 20 30 40 50

Sample Input 0

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

Sample Output 0

```
Preorder Traversal is:
10 20 30 40 50
```


Java 7

```java
//224G1A0553
import java.util.*;
class Node {
    int data;
    Node left;
    Node right;
    public Node( int item) {
        this.data = item;
        this.left = null;
        this.right = null;
    }}
```

```
class StackNode {
    Node node;
    StackNode next;
    public void StackNode(Node b) {
        this.node = b;
        this.next = null;
    }}
public class NonRecursivePreorder {
    StackNode top;
    Node root;
    public void NonRecursivePreorder() {
        top = null;
        root = null;
    }
    boolean isEmpty() {
        if(top == null) {
            return true;
        }
        return false;
    }
    void push(Node b) {
        StackNode temp;
        temp = new StackNode();
        if(temp == null) {
            System.out.printf("Stack is overflow.\n");
        } else {
            temp.node = b;
            temp.next = top;
            top = temp;
        } }
    Node peek() {
        if (top == null) {
            return null;
        }
        return top.node;
    }
    Node pop() {
        StackNode temp;
        Node b;
        if(top == null) {
            System.out.printf("Stack is underflow.\n");
            return null;
        } else {
            temp = top;
            top = top.next;
            b = temp.node;
            return b;
        }    }
    void preorderInBST(Node root) {
        Node curr = root;
        push(root);
        while(true) {
            curr = pop();
            System.out.printf("%d ",curr.data);
            if(curr.right != null) {
                push(curr.right);
            }
            if(curr.left != null) {
                push(curr.left);
            }
            if(isEmpty())
                break;
        }}
/* Insertion into binary search tree */
    Node insertBinarySearchTree(Node root, int item) {

```

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

Line: 11 Col: 6

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
2
3
```

Your Output (stdout)

```
Preorder Traversal is:
10 20 30 40 50
```

Expected Output

```
Preorder Traversal is:
10 20 30 40 50
```