29/06/2025

Sunday

**public** **class** Node {

**int** data;

  Node next;

**public** Node(**int** d) {

**this**.data = d;

  }

}

**public** **class** LinkedListExample {

  Node head;

**public** **void** add(**int** data) {

    Node newNode =  **new** Node(data);

**if**(head==**null**) {

      head = newNode;

    }**else** {

      Node temp = head;

**while**(temp.next != **null**) {

        temp = temp.next;

      }

      temp.next=newNode;

    }

  }

**public** **void** display() {

    Node temp=head;

**while**(temp!=**null**) {

      System.***out***.print(temp.data+"->");

      temp=temp.next;

    }

    System.***out***.println("null");

  }

**public** **static** **void** main(String[] args) {

    LinkedListExample list = **new** LinkedListExample();

    list.add(5);//5,3,1,7,8

    list.add(3);

    list.add(1);

    list.add(7);

    list.add(8);

    list.display();

  }

}

import java.util.\*;

import java.io.\*;

class Node {

    Node left;

    Node right;

    int data;

    Node(int data) {

        this.data = data;

        left = null;

        right = null;

    }

}

class Solution {

/\* you only have to complete the function given below.

Node is defined as

class Node {

    int data;

    Node left;

    Node right;

}

\*/

    public static void inOrder(Node root) {

        if(root ==null )

        return;

        inOrder(root.left);

        System.out.print(root.data + " ");

        inOrder(root.right);

    }

    public static Node insert(Node root, int data) {

        if(root == null) {

            return new Node(data);

        } else {

            Node cur;

            if(data <= root.data) {

                cur = insert(root.left, data);

                root.left = cur;

            } else {

                cur = insert(root.right, data);

                root.right = cur;

            }

            return root;

        }

    }

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);

        int t = scan.nextInt();

        Node root = null;

        while(t-- > 0) {

            int data = scan.nextInt();

            root = insert(root, data);

        }

        scan.close();

        inOrder(root);

    }

}

import java.util.\*;

import java.io.\*;

class Node {

    Node left;

    Node right;

    int data;

    Node(int data) {

        this.data = data;

        left = null;

        right = null;

    }

}

class Solution {

/\* you only have to complete the function given below.

Node is defined as

class Node {

    int data;

    Node left;

    Node right;

}

\*/

    public static void postOrder(Node root) {

        if(root ==null )

        return;

        postOrder(root.left);

        postOrder(root.right);

        System.out.print(root.data + " ");

    }

    public static Node insert(Node root, int data) {

        if(root == null) {

            return new Node(data);

        } else {

            Node cur;

            if(data <= root.data) {

                cur = insert(root.left, data);

                root.left = cur;

            } else {

                cur = insert(root.right, data);

                root.right = cur;

            }

            return root;

        }

    }

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);

        int t = scan.nextInt();

        Node root = null;

        while(t-- > 0) {

            int data = scan.nextInt();

            root = insert(root, data);

        }

        scan.close();

        postOrder(root);

    }

}

import java.util.\*;

import java.io.\*;

class Node {

    Node left;

    Node right;

    int data;

    Node(int data) {

        this.data = data;

        left = null;

        right = null;

    }

}

class Solution {

/\* you only have to complete the function given below.

Node is defined as

class Node {

    int data;

    Node left;

    Node right;

}

\*/

    public static void preOrder(Node root) {

        if(root ==null )

        return;

        System.out.print(root.data + " ");

        preOrder(root.left);

        preOrder(root.right);

    }

    public static Node insert(Node root, int data) {

        if(root == null) {

            return new Node(data);

        } else {

            Node cur;

            if(data <= root.data) {

                cur = insert(root.left, data);

                root.left = cur;

            } else {

                cur = insert(root.right, data);

                root.right = cur;

            }

            return root;

        }

    }

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);

        int t = scan.nextInt();

        Node root = null;

        while(t-- > 0) {

            int data = scan.nextInt();

            root = insert(root, data);

        }

        scan.close();

        preOrder(root);

    }

}