import java.util.\*;

class TreeNode {

int val;

TreeNode left;

TreeNode right;

TreeNode(int val) {

this.val = val;

}

}

class Solution {

public TreeNode sortedArrayToBST(int[] nums) {

return buildBST(nums, 0, nums.length - 1);

}

private TreeNode buildBST(int[] nums, int left, int right) {

if (left > right) return null;

int mid = left + (right - left) / 2;

TreeNode root = new TreeNode(nums[mid]);

root.left = buildBST(nums, left, mid - 1);

root.right = buildBST(nums, mid + 1, right);

return root;

}

}

public class Main5 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Take the entire sorted array as a line of input

System.out.println("Enter sorted integers separated by space:");

String line = sc.nextLine();

// Split and parse into array

String[] parts = line.trim().split("\\s+");

int[] nums = new int[parts.length];

for (int i = 0; i < parts.length; i++) {

nums[i] = Integer.parseInt(parts[i]);

}

// Build BST

Solution solution = new Solution();

TreeNode root = solution.sortedArrayToBST(nums);

// Print output in level-order with nulls

System.out.print("Output: ");

printLevelOrderWithNulls(root);

}

public static void printLevelOrderWithNulls(TreeNode root) {

if (root == null) {

System.out.println("[]");

return;

}

List<String> result = new ArrayList<>();

Queue<TreeNode> queue = new LinkedList<>();

queue.add(root);

while (!queue.isEmpty()) {

TreeNode curr = queue.poll();

if (curr != null) {

result.add(String.valueOf(curr.val));

queue.add(curr.left);

queue.add(curr.right);

} else {

result.add("null");

}

}

// Remove trailing nulls

int i = result.size() - 1;

while (i >= 0 && result.get(i).equals("null")) {

result.remove(i);

i--;

}

System.out.println(result.toString());

}

}