**Question - 1**

**Solution:**

import java.util.Scanner;

import java.util.Stack;

public class SkyscraperConstruction {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the total number of floors in the building");

int N = scanner.nextInt();

Stack<Integer> floorStack = new Stack<>();

Stack<Integer> tempStack = new Stack<>();

System.out.println("Enter the floor size given on each day:");

for (int i = 0; i < N; i++) {

int size = scanner.nextInt();

// Check if any floors can be assembled

while (!floorStack.isEmpty() && floorStack.peek() < size) {

tempStack.push(floorStack.pop());

}

// Assemble the current floor

floorStack.push(size);

// Move back the floors from tempStack to floorStack

while (!tempStack.isEmpty()) {

floorStack.push(tempStack.pop());

}

// Print the assembled floors for the current day

System.out.print("Day: " + (i + 1) + "\n");

while (!floorStack.isEmpty()) {

System.out.print(floorStack.pop() + " ");

}

System.out.println();

}

scanner.close();

}

}

**Question – 2**

**Solution:**

class Node {

int data;

Node left;

Node right;

public Node(int data) {

this.data = data;

this.left = null;

this.right = null;

}

}

public class Main {

Node node;

// Function to convert BST to a Right Skewed Tree

public Node convertToRightSkewed(Node root) {

if (root == null) {

return null;

}

// Initialize two pointers for the skewed tree

Node prev = null;

Node newRoot = null;

// Perform in-order traversal to convert to skewed tree

while (root != null) {

Node rightChild = root.right;

// Modify the root's left child to null

root.right = prev;

root.left = null;

// Update the previous node and new root

prev = root;

newRoot = newRoot == null ? root : newRoot;

// Move to the right subtree

root = rightChild;

}

return newRoot;

}

// Function to print the nodes in ascending order

public void printInOrder(Node root) {

if (root == null) {

return;

}

printInOrder(root.left);

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

printInOrder(root.right);

}

public static void main(String[] args) {

Main tree = new Main();

tree.node = new Node(50);

tree.node.left = new Node(30);

tree.node.right = new Node(60);

tree.node.left.left = new Node(10);

tree.node.right.left = new Node(55);

System.out.println("Original BST:");

tree.printInOrder(tree.node);

System.out.println();

// Convert BST to Right Skewed Tree

Node skewedRoot = tree.convertToRightSkewed(tree.node);

System.out.println("Skewed Tree (In Ascending Order):");

tree.printInOrder(skewedRoot);

}

}