CSN-261: Data Structures Laboratory

Lab Assignment 7 (L7)

Instruction: Use Java for solving the assignment.

Q1. Given height 'h' of an AVL tree, you are required to find the minimum number of nodes than an AVL tree can have.

Input format: Single integer h, representing the height of the AVL tree.

Output format: Number of nodes N

Test Cases

```
Input 1:                Output 1:
0                       1

Input 2:                Output 2:
5                       20
```

Q2. Given in-order and pre-order traversal of a binary tree, find the post-order traversal.

Input Format: The first line contains an integer N denoting the number of nodes in the binary tree. The second line contains N space-separated integers which are in-order traversal of the tree. The third line contains N space-separated integers which are pre-order traversal of the tree. (Constraint: 6 <= N <= 20)

Output Format: A single line containing a sequence of N integers, describing the post order traversal of the tree

Test Case

```
Input 1:
6
4 2 5 1 3 6

1 2 4 5 3 6

Output 1:
4 5 2 6 3 1
```

Q3. Given a complete binary tree of N nodes and each node has some value Ai. Find the minimum number of swaps you can make to convert the binary tree into a binary search tree. In one swap, you can swap any two nodes (just values of the nodes).

You will be given the array representation of the binary tree. The root of the tree will be at A1. The left child of the root will be at A2 and the right child of the root will be at A3. The left child of the node at the array position k will be at A(2∗k) and right child of the node at the array position k will be at A(2∗k+1). (Considering it to be indexed 1).

Input format: The first line contains an integer, N, denoting the number of nodes. The second line contains N space-separated integers Ai, denoting the value attached to i^{th} node.

Output: Print a single integer, denoting the minimum number of swaps needed to convert a binary tree into a binary search tree.

Test Case

```
Input 1:
9
5 7 9 2 10 6 3 1 8

Output 1:
4
```