Height of a Tree

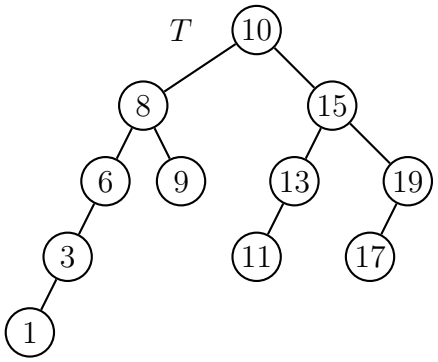
Time Limit: 2 seconds

Problem Description

Let us recall what is a binary search tree. In a binary search tree, each node of the binary search tree has a comparable key K. If a node N has a right child and R is a node of N's right subtree, then the key K_N of N must be no more than the key K_R of R. If a node N has a left child and L is a node of N's left subtree, then the key K_N of N must be no less than the key K_L of L. To represent a binary search tree T, you are given its preorder traversal. The preorder traversal of T is generated by the following procedure.

1. Print the key of the root of T.
2. If the root of T has left child, then print the preorder traversal of the left subtree of T.
3. If the root of T has right child, then print the preorder traversal of the right subtree of T.

For example, the preorder traversal of the following tree T is `10 8 6 3 1 9 15 13 11 19 17`.



The height of a node N in a tree is the length of the longest downward path to a leaf from N. In the above tree T, the height of 10 is 4, and the height of 1 is 0. The height of a tree is the height of its root, so the height of T is 4. The efficiency of a binary search tree heavily depends on its height. In this problem, you are asked to compute the heights of binary search trees.

Technical Specifications

1. The number of test cases is no more than 20.
2. The number of nodes in a test case now is no more than 20,000.
3. The keys are distinct positive integers and less than 100,000.

Input Format

The first line of the input file contains an integer indicating the number of test cases. The first line of each test case contains an integer n which denotes the number of nodes in the binary search tree. The second line contains n integers $k_1, \ldots, k_n$ separated by blanks where "$k_1 \ldots k_n$" is the preorder traversal of the binary search tree.

Output Format

For each test case, output the height of the binary search tree.

Sample Input

```
3
1
1
11
10 8 6 3 1 9 15 13 11 19 17
11
11 1 2 3 10 9 8 7 4 6 5
```

Sample Output

```
0
4
10
```