Find if the Binary Search Tree an AVL Tree

Given a binary search tree (BST), which is represented in arrays as an implicit data structure as explained in http://en.wikipedia.org/wiki/Binary_tree#Arrays. In this structure, if a node has an index i, its children (if any) are found at indices 2i+1 and 2i+2, while its parent (if any) is found at index floor((i-1)/2). An AVL tree is a self-balancing BST where the Balance Factor (balanceFactor = height(left subtree) - height(right subtree)) of every node is -1, 0 or +1. Otherwise, it is not an AVL tree. Find whether the given BST is an AVL tree or not.

Input

The input begins with the number t of test cases in a single line (1 <= t <= 100). Each test case beings on a new line with a nonnegative integer n followed by n integers separated by spaces (0 <= n <= 100), where n is the number of places required to store the BST in the array representation. The array representation of the binary tree would have node values (0 <= node value <= 10000), and null-nodes are represented as -1s. The given binary tree is guaranteed to be a BST.

Output

For each test case, print T or F on a new line to indicate whether the given BST is an AVL tree or not, respectively.

Example

Input:

```
12
0
1 10
2 20 10
3 20 10 30
4 30 20 -1 10
5 30 10 -1 -1 20
7 20 10 30 -1 -1 -1 40
7 10 -1 20 -1 -1 -1 30
7 40 20 60 10 30 50 70
10 40 20 60 -1 30 50 70 -1 -1 25
11 10 05 20 04 07 12 -1 02 -1 -1 08
11 10 05 20 04 07 -1 -1 02 -1 -1 08
```

Output:

```
T
T
T
T
F
F
T
F
T
```

F
T
F