CSGE602040 - Struktur Data dan Algoritma
Semester Gasal - 2019/2020
WS 4 - Selasa
Deadline: Selasa, 05-11-2019, 18.00 WIB

Mengecek Tree

Deskripsi

Namron ★ merupakan seorang mahasiswa pecinta tumbuhan, khususnya *binary search tree*. Di kelas Struktur Data dan Algoritma, Namron ★ diajarkan tentang *postorder traversal* dari sebuah binary tree. Sekarang Namron ★ penasaran, jika diberikan **T** buah hasil *postorder traversal* dari sebuah **binary search tree**, apakah benar ada *binary search tree* yang memiliki *postorder traversal* tersebut dan jika ada, berapa tinggi *tree* tersebut?

Masukan

Baris pertama berisi sebuah bilangan bulat **T** yaitu banyak kasus uji.
T baris berikutnya berisi sebuah bilangan bulat **N** yaitu banyak *node binary search tree* pada awalnya diikuti dengan **N** bilangan yang **mungkin** merupakan hasil *postorder traversal* dari sebuah **binary search tree**. Dijamin nilai-nilai tersebut unik.

Keluaran

T baris yang masing-masing berisi sebuah bilangan yang merupakan tinggi (*height*) dari *tree* pada tiap kasus uji atau -1 jika tidak ada *binary search tree* yang memiliki *postorder* seperti itu.

Batasan

$1 \leq \mathbf{T} \leq 5$
$1 \leq \mathbf{N}_t \leq 1.000$
$1 \leq v_{ti} \leq 10^9$ (v_{ti} adalah nilai elemen pada *tree* pada kasus uji ke-t)
$v_{ti} = v_{tj}$ jika dan hanya jika i = j

Contoh Masukan 1

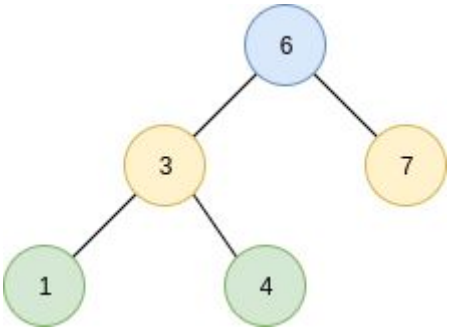
```
3
5 1 4 3 7 6
3 3 1 2
4 4 1 2 3
```

Contoh Keluaran 1

```
2
-1
-1
```

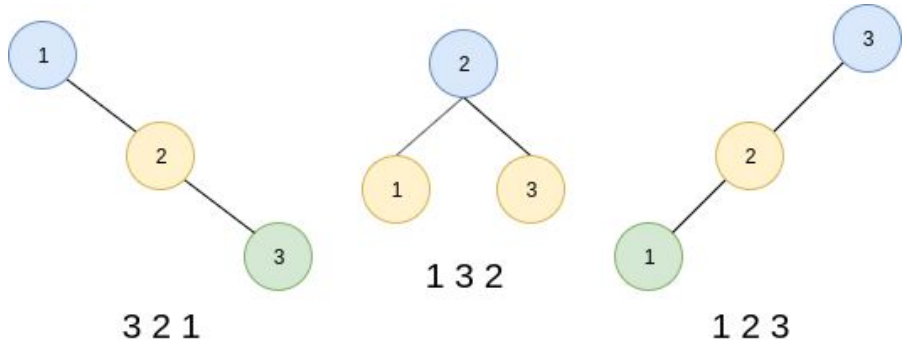
Penjelasan

Tree pada kasus uji pertama adalah sebagai berikut.



Dengan demikian, tinggi dari *tree* pada kasus uji pertama adalah 2.

Semua kemungkinan *binary search tree* dengan 3 node bernilai 1, 2, dan 3 beserta *postorder*-nya adalah sebagai berikut.



Dapat dilihat tidak ada *binary search tree* dengan *postorder* 3 1 2. Dengan demikian, jawaban untuk kasus uji kedua adalah -1. Dapat dibuktikan pula bahwa tidak ada *binary search tree* dengan *postorder* 4 1 2 3.

Contoh Masukan 2

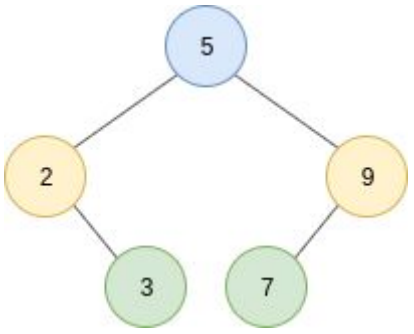
```
2
5 3 2 7 9 5
2 1 2
```

Contoh Keluaran 2

```
2
1
```

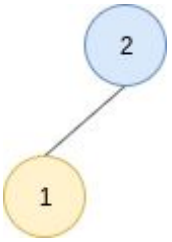
Penjelasan

Tree pada kasus uji 1 adalah sebagai berikut.



Dengan demikian, tinggi dari *tree* tersebut adalah 2

Sedangkan *tree* pada kasus uji 2 adalah sebagai berikut.



Tinggi *tree* tersebut adalah 1.

Hint: 70% test case terdiri dari kasus uji yang semuanya merupakan *postorder* dari suatu *binary search tree* yang valid.