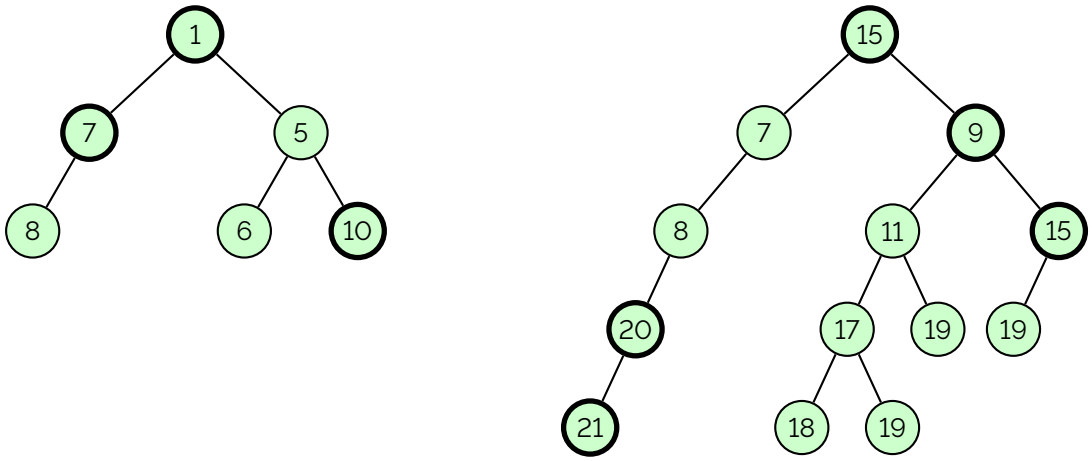
Niveles aventajados en un árbol binario

Como ya sabemos, los nodos de un árbol binario se agrupan en niveles según su distancia a la raíz. De este modo, la raíz se encuentra en el nivel 1, los hijos de la raíz están en el nivel 2, los nietos en el nivel 3, y así sucesivamente.

Decimos que un nivel i es *aventajado* si el valor máximo de los nodos de ese nivel supera estrictamente al valor máximo de los nodos del nivel $i - 1$. Por convenio, consideramos que el nivel 1 (esto es, el de la raíz) nunca es aventajado, ya que no tiene un nivel anterior.

Por ejemplo, consideramos los siguientes árboles, donde se muestran resaltados los nodos que tienen el valor máximo de su correspondiente nivel.



En el árbol de la izquierda, los niveles segundo y tercero son aventajados, ya que $7 > 1$, y $10 > 7$. En el árbol de la derecha, el nivel segundo no es aventajado, pero sí lo son los niveles tercero, cuarto y quinto.

1. Implementa una función `num_aventajados` con la siguiente cabecera:

   ```
   int num_aventajados(const BinTree<int> &tree);
   ```

   Esta función debe devolver el número de niveles aventajados que tiene el árbol pasado como parámetro. Por ejemplo, para el árbol de la izquierda mostrado anteriormente debe devolver 2, y para el árbol de la derecha debe devolver 3.

   Importante: no se permite el uso de parámetros de E/S para resolver este ejercicio.

2. Indica el coste de la función anterior, en función del número de nodos del árbol pasado como parámetro.

Entrada

La entrada comienza con un número que indica el número de casos de prueba que vienen a continuación. Cada caso de prueba consiste en una línea con la descripción de un árbol binario mediante la notación vista en clase. El árbol vacío se representa mediante `.` y el árbol no vacío mediante `(iz x dr)`, siendo `x` la raíz, e `iz` y `dr` las representaciones de ambos hijos. Los valores de cada nodo pueden ser números enteros comprendidos entre 0 y 10^9.

Salida

Para cada caso de prueba debe escribirse una línea con el número de niveles aventajados que tiene el árbol leído en la entrada.

Entrada de ejemplo

```
4
(((. 8 .) 7 .) 1 ((. 6 .) 5 (. 10 .)))
(((((. 21 .) 20 .) 8 .) 7 .) 15 ((((. 18 .) 17 (. 19 .)) 11 (. 19 .)) 9 ((. 19 .) 15 .)))
((. 5 .) 5 (. 5 .))
.
```

Salida de ejemplo

```
2
3
0
0
```

Créditos

Manuel Montenegro.