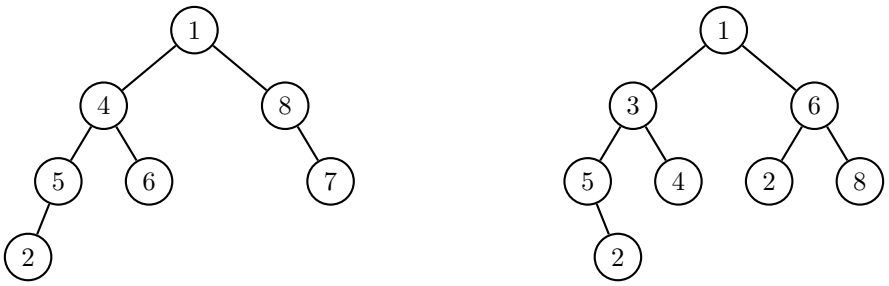
Problema D

Dado un árbol binario de enteros, se dice que está *pareado* si la diferencia entre la cantidad de números pares en el hijo izquierdo y en el hijo derecho no excede la unidad y, además, tanto el hijo izquierdo como el derecho están pareados. Dicho de otra forma, todos los nodos cumplen que la diferencia entre las cantidades de números pares en sus dos hijos es como mucho uno.

Por ejemplo, de los siguienes árboles, el de la izquierda no está pareado porque la raíz no cumple la condición, al tener tres pares en su hijo izquierdo y solamente uno en su hijo derecho (aunque el resto de nodos sí cumplen la condición). El de la derecha sí está pareado.



Dado un árbol binario queremos averiguar si está pareado o no.

Requisitos de implementación.

Se debe implementar una función *externa* a la clase `bintree` que explore el árbol de manera eficiente averiguando si está pareado o no. La función no podrá tener parámetros de entrada/salida.

Entrada

La entrada comienza con el número de casos que vienen a continuación. Cada caso de prueba consiste en una línea con la descripción de un árbol binario: si el árbol es vacío se representa con un –1; si no, primero aparece su raíz, y a continuación la descripción del hijo izquierdo y después la del hijo derecho, dadas de la misma manera.

Salida

Para cada árbol, se escribirá una línea con un `SI` si el árbol está pareado y un `NO` si no lo está.

Entrada de ejemplo

```
4
1 4 5 2 -1 -1 -1 6 -1 -1 8 -1 7 -1 -1
1 3 5 -1 2 -1 -1 4 -1 -1 6 2 -1 -1 8 -1 -1
-1
2 -1 3 -1 4 -1 5 -1 6 -1 -1
```

Salida de ejemplo

```
NO
SI
SI
NO
```