Iluminando el árbol de Navidad

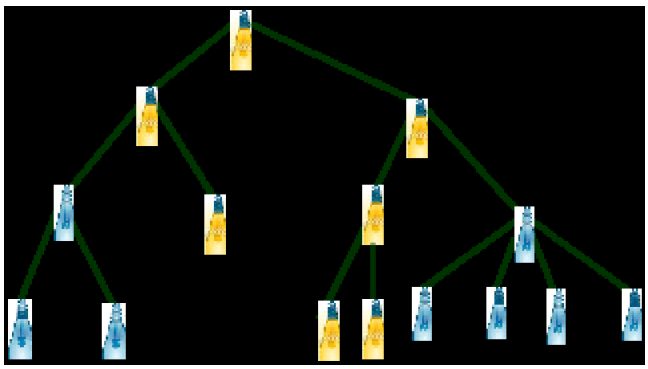
Contribución de Facundo Gutiérrez

Descripción del problema

En un pequeño poblado, los habitantes han decidido iluminar para Navidad, un inmenso pino que tienen en el centro de su plaza principal. El juego de luces que han conseguido armar, tiene **L** foquitos conectados entre sí en forma de árbol, y uno de ellos se encuentra conectado directamente a una fuente de energía. Obviamente, para todos los demás foquitos existe un único camino del cableado que lo conecta con la fuente de energía.

En un juego de luces tan grande, algunos foquitos pueden estar fallados, y otros funcionar normalmente. Un foquito cualquiera estará encendido solamente si todos los foquitos presentes en el camino desde la fuente de energía hasta él funcionan normalmente. Notablemente, si un foquito se quema, impide que se prendan todos los que cuelgan de él.
En esta situación, se desea determinar la cantidad de luces encendidas.

Puede apreciarse una posible situación en la siguiente figura:



A los efectos de tener un descripción completa del estado del juego de luces, se te pide que elabores una función **navidad (L, Arbol)** que, dada la descripción del juego de luces, y consultando el estado de algunos foquitos, permita determinar la cantidad total de foquitos encendidos. Los parámetros son:

L: cantidad total de luces en el árbol **(1 ≤ L ≤ 10.000).**
Arbol: arreglo de 2x(L-1) indicando los enlaces entre pares de foquitos que se numeran de 1 a L, donde 1 es el foquito conectado a la fuente de energía.

La función debe retornar la cantidad de foquitos encendidos.

Para obtener puntaje, tu programa debe determinar correctamente la cantidad de luces encendidas. Además, el puntaje de un programa que determina la cantidad correctamente dependerá de la cantidad de consultas realizadas: mientras menos consultas se realicen, mejor será el puntaje que se obtendrá (se obtendrán cero puntos si se realizan más de L consultas).

Ejemplo

Si tuvieses un juego de luces como el de la figura cuyo **L**=15, en un árbol descrito como sigue,

1 2 1 3 3 6 2 5 2 4
3 7 4 8 7 15 6 10 6 11
7 12 7 13 7 14 4 9

la respuesta debería ser 7.

Detalles de implementación

En un único archivo, llamado **navidad.cpp**, **navidad.c**, o **navidad.pas** debes enviar una función que obtenga la cantidad de luces encendidas, usando los siguientes prototipos:

En **C/C++:** long **navidad** (long L, long Arbol [][2])

En **Pascal**:
type ArbolArray = array [1..10000-1,1..2] of longint;
function **navidad** (L : longint; var Arbol : ArbolArray) : longint;

Para realizar las consultas, puedes realizar llamadas una función externa.
En **C/C++:**
int encendido(int foquito);
En **Pascal**:
function encendido(foquito : longint) : longint;
(Esta función pascal se encuentra en la unidad "consulta", por lo que se debe usar "uses consulta" en el código que se enviará)

Esta función devuelve 1 si el foquito está encendido, 0 si está apagado, y se genera un error si se consulta por un número de foquito inexistente.

Evaluador local

El evaluador local (programa para probar ejemplos propios) lee la entrada por stdin en el siguiente formato:

Línea 1: L
Línea 2+i (0 ≤ i < L-2): 2 números entre 1 y L que indican el enlace entre 2 foquitos.
Línea L+i (1 ≤ i < L): Un 0 si el foquito i está apagado, y un 1 si está encendido. El evaluador de ejemplo que se les provee usará estos valores provistos directamente en sus respuestas a las consultas realizadas por el programa.

El evaluador muestra la interacción y resultado provisto por el programa por la consola.

Para el caso del ejemplo la entrada sería:

```
15
1 2
1 3
3 6
2 5
2 4
3 7
4 8
7 15
6 10
6 11
7 12
7 13
7 14
4 9
1
1
1
0
1
1
0
0
0
```

```
1
1
0
0
0
0
```

Subtareas

Habrá casos de prueba por un total de **30** puntos en los cuales solamente el foco conectado a la fuente de energía podrá tener más de un hijo.