International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia

dreaming

Spanish — 1.0

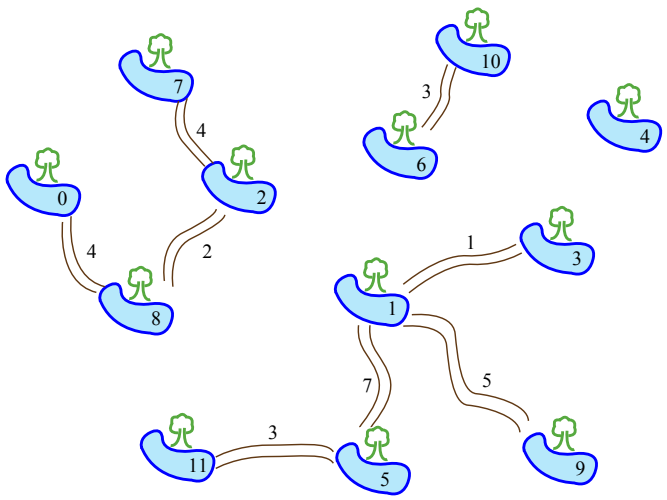
Esta historia sucedió hace mucho tiempo, cuando se creó el mundo y la IOI aún no había sido soñada.

Serpiente vive en un lugar que tiene N estanques (agujeros llenos de agua), numerados $0, \ldots, N-1$. Hay M senderos bidireccionales uniendo pares de estanques, por los cuales Serpiente puede viajar. Cada par de estanques está conectados (directamente o indirectamente) por como mucho una secuencia de senderos, aunque algunos pares de estancos podrían no estar conectados en absoluto (o sea $M \leq N-1$). Serpiente viaja a traves de cada sendero tardando un número concreto que puede ser distinto por cada sendero.

Canguro, el amigo de Serpiente, quiere construir $N-M-1$ senderos nuevos, para que así Serpiente pueda viajar entre cualquier par de estanques. Canguro puede crear senderos entre cualquier par de estanques, y Serpiente tardará L días en cruzar los nuevos senderos de Canguro.

Además, Canguro quiere que Serpiente viaje lo más rápido posible. Canguro construirá nuevos senderos para que la distancia entre dos estanques sea tan pequeña como sea posible. Ayuda a Canguro y Serpiente y determina el viaje más largo entre dos estanques después que Cangur haya consturido los senderos de esta manera.

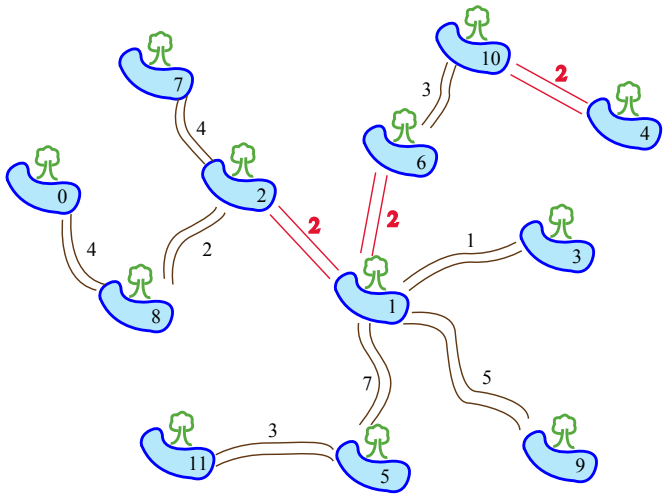
Ejemplos



En la figura adjunta hay `N = 12` estanques y `M = 8` senderos. Supongamos `L = 2`, eso es, cualquier sendero nuevo requerirá a Serpiente 2 días para cruzarlo.

Entonces Canguro podría construir 3 nuevos senderos:

- entre los estanques 1 y 2;
- entre los estanques 1 y 6;
- entre los estanques 4 y 10.



La figura superior muestra el conjunto de los senderos finales. El viaje más largo es de 18 días, entre los estanques 0 y 11. Este es el resultado más pequeño posible - no importa como Canguro construya los senderos, que siempre existirá un par de estanques que harán que Serpiente necesite para viajar 18 días o más.

Implementación

Tendrás que enviar un archivo que implemente la función `travelTime()` como se describe:

Tu función: `travelTime()`

C/C++

```
int travelTime(int N, int M, int L,
        int A[], int B[], int T[]);
```

Pascal

```
function travelTime(N, M, L : LongInt;
        var A, B, T : array of LongInt) : LongInt;
```

Descripción

Esta función debería calcular el mayor de los tiempos (medido en días) para viajar entre cualquier par de estanques, asumiendo que canguro ha añadido `N - M - 1` senderos de modo que todos los estanques esten conectados y este tiempo máximo sea tan pequeño como se pueda.

Parámetros

- `N` : Número de estanques.
- `M` : Número de senderos que ya existen.
- `L` : Tiempo en días para los que Serpiente tardará en cruzar si viaja a traves de estos senderos nuevos.
- `A` , `B` y `T` : Arrays de longitud `M` que especifica los puntos iniciales, finales y el tiempo de viaje de cada sendero pre-existente. Eso es que el sendero `i`-ésimo une los estanques `A[i-1]` y `B[i-1]` , y cuesta `T[i-1]` días para cruzarlo en cualquier dirección.
- *Returns*: El tiempo de viaje máximo entre cualquier par de estanques, dada la descripción previa.

Sesión de Muestra

La siguiente sesión describe el ejemplo anterior:

Parameter	Value
N	12
M	8
L	2
A	[0, 8, 2, 5, 5, 1, 1, 10]
B	[8, 2, 7, 11, 1, 3, 9, 6]
T	[4, 2, 4, 3, 7, 1, 5, 3]
Returns	18

Restricciones

- Time limit: 1 second
- Memory limit: 64 MiB
- $1 \le N \le 100,000$
- $0 \le M \le N - 1$
- $0 \le A[i], B[i] \le N - 1$
- $1 \le T[i] \le 10,000$
- $1 \le L \le 10,000$

Sub-tareas

Subtarea	Puntos	Restricciones adicionales de la entrada
1	14	$M = N - 2$, y hay exactamente uno o dos senderos preexistentes saliendo de cada estanque. En otras palabras, hay dos conjuntos de estanques conectados, y en cada conjunto los senderos forman un camino sin ramificaciones.
2	10	$M = N - 2$ y $N \le 100$
3	23	$M = N - 2$
4	18	Como mucho hay un sendero pre-existente saliendo de cada estanque.
5	12	$N \le 3,000$
6	23	*(Ninguna)*

Experimentación

El grader en tu ordenador leerá la entrada del archivo `dreaming.in`, que tendrá el siguiente formato:

- línea 1: `N M L`
- lines 2, …, `M + 1`: `A[i] B[i] T[i]`

Por ejemplo el caso mostrado anteriormente debe ser codificado como de la siguiente forma:

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

Notas del Lenguaje

C/C++	You must `#include "dreaming.h"`.
Pascal	You must define the `unit Dreaming`. All arrays are numbered beginning at `0` (not `1`).

See the solution templates on your machine for examples.