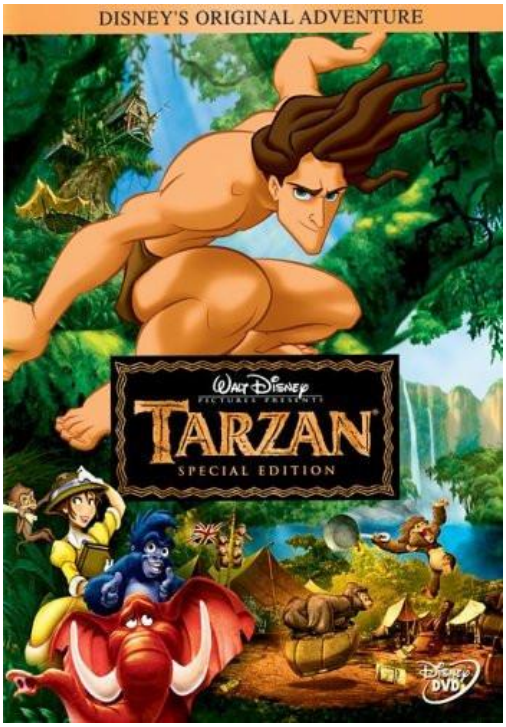
Examen Mundial I de Programación
Curso 2018-2019

Tarzán de la jungla

NOTA: Si usted está leyendo este documento sin haber extraído el compactado que se le entregó, ciérrelo ahora, extraiga todos los archivos en el escritorio, y siga trabajando desde ahí. Es un error común trabajar en la solución dentro del compactado, lo cual provoca que los cambios no se guarden. Si usted comete este error y entrega una solución vacía, no tendrá oportunidad de reclamar.



Tarzán vive en la jungla desde hace mucho tiempo y ha decidido fijar su residencia permanente en la comunidad de simios que lo acogió luego de su naufragio. Teniendo en cuenta su supuesta inteligencia superior, al ser un miembro de la especie *homo sapiens*, se ha dado a la tarea de mejorar la vida de la comunidad.

Producto del contacto con las personas que visitan la jungla, la calidad de vida de los simios ha mejorado, permitiéndoles vivir más años. Es por ello que el envejecimiento poblacional es una realidad de la jungla contemporánea.

La idea de Tarzán es construir escaleras para facilitar la ascensión de los simios más ancianos a sus hogares en la copa de los árboles. Para ello cuenta con una cantidad de tablas y cuerdas. Las cuerdas funcionan como peldaños de las escaleras y son recursos de muy fácil adquisición, prácticamente ilimitados. Es además deseable que la cantidad de cuerdas sea la mayor posible para facilitar la escalada. Por otro lado, las tablas, que se emplearían como base estructural de las escaleras, son escazas. Con este fin, Tarzán tiene pensado construir las escaleras consecutivas, creando una gran estructura contigua, ya que generalmente los simios viven en árboles adyacentes. De esta manera, cada tabla intermedia forma parte de dos escaleras, exceptuando las de los extremos del bloque, que solo participan en una. Las tablas tienen agujeros por donde se van a pasar las cuerdas para formar escaleras. Las restricciones son las siguientes:

- Las cuerdas deben pasar por uno y solo un agujero de cada una de las tablas del bloque de escaleras.
- El diámetro del orificio en la primera y la última tabla por la que pasa cada cuerda debe ser exactamente el mismo.
- Los diámetros de los agujeros por los que pasa la soga siempre deben ser mayores o iguales al diámetro del agujero de la tabla inicial a la que está sujeta.
- Las cuerdas no pueden cruzarse.

A continuación, algunos ejemplos ilustrativos de posibles soluciones. Las columnas representan a las tablas con sus diámetros de orificios, y los colores, que pasan por todas las filas, representan los recorridos de las cuerdas desde la tabla inicial hasta la final.

En la configuración siguiente (Figura 1) el mayor número de cuerdas que se puede poner es 4. Note que los diámetros iniciales y finales coinciden y durante el recorrido de cada soga los valores intermedios siempre son mayores o iguales que los de los extremos, lo que permite que la soga pase. En la Figura 2 se presenta una configuración alternativa con igual cantidad de cuerdas empleadas, lo que implica que la mejor solución no tiene que ser única.

2	4	2	3	1
1	3	3	5	3
3	4	3	4	2
2	2	5	2	2
1	5	4	4	1

Figura 1. Configuración con cantidad máxima de cuerdas igual a 4.

2	4	2	3	1
1	3	3	5	3
3	4	3	4	2
2	2	5	2	2
1	5	4	4	1

Figura 2. Configuración alternativa con la misma cantidad de cuerdas.

Es preciso señalar que en correspondencia a los recorridos de las cuerdas la configuración puede no ser la mejor, como ocurre en la Figura 3. Al poner la cuerda en **rojo** estamos empeorando la solución.

2	4	2	3	1
1	3	3	5	3
3	4	3	4	2
2	2	5	2	2
1	5	4	4	1

Figura 1. Configuración con cantidad máxima de cuerdas igual a 4.

2	4	2	3	1
1	3	3	5	3
3	4	3	4	2
2	2	5	2	2
1	5	4	4	1

Figura 3. Configuración no deseada, pues no maximiza la cantidad de cuerdas.

Note que la siguiente solución (Figura 4) no es válida pues las cuerdas en **rojo** y **violeta** se cruzan.

2	4	2	3	1
1	3	3	5	3
3	4	3	4	2
2	2	5	2	2
1	5	4	4	1

Figura 4. Configuración no válida, las cuerdas en **rojo** y **violeta** se cruzan.

En la Figura 5 existen múltiples soluciones, todas con cantidad máxima de cuerdas igual a 1. Mientras, en la Figura 6 no es posible poner ninguna cuerda. La única posible solución sería hacer corresponder los dos orificios de tamaño 5 en las tablas de los extremos, pero se descarta debido a que no existe ningún recorrido de orificios intermedios que sean **TODOS** mayores o iguales a 5.

1	2	2	3	3
5	2	4	5	4
2	5	2	3	1
3	2	5	1	1
2	3	4	5	1

Figura 5. Para este conjunto de tablas existen múltiples configuraciones, todas emplean una cuerda.

4	5	4	5	5
3	2	1	5	2
3	3	4	1	2
5	2	4	2	5
3	2	1	4	1

Figura 6. Para este conjunto no es posible poner ninguna cuerda bajo las restricciones del problema.

Luego de descrito el problema, Tarzán se percata de su complejidad y admite que debe pedir ayuda. Usted, como apasionado de la vida salvaje, decide apoyarlo programando un método que permita determinar la mayor cantidad de cuerdas que se pueden emplear, conocidas las tablas disponibles y los tamaños de los orificios por los que pasarán las cuerdas.

Usted debe haber recibido junto a este documento una solución de Visual Studio con dos proyectos: una biblioteca de clases (*Class Library*) y una aplicación de consola (*Console Application*). Usted debe completar la implementación de la clase `Tarzan` en el *namespace* `Weboo.Examen`. En la biblioteca de clases encontrará la siguiente definición:

```
namespace Weboo.Examen
{
    public static class Tarzan
    {
        public static int ConstruyeEscaleras(int[,] tablas)
        {
            throw new NotImplementedException();
        }
    }
}
```

Usted debe implementar el método *ConstruyeEscaleras* que recibe el array de las tablas, cada una representada como un array de enteros con los diámetros, y devuelve la cantidad máxima de cuerdas que se pueden emplear para dicho conjunto de tablas.

NOTA: Todo el código de la solución debe estar en este proyecto (biblioteca de clases), pues es el único código que será evaluado. Usted puede adicionar todo el código que considere necesario, pero no puede cambiar los nombres del namespace, clase o método mostrados. De lo contrario, el probador automático fallará. En particular, es imprescindible que usted **no cambie la definición** de la clase `Tarzan`. Por supuesto, usted puede (y debe) adicionar todo el código que necesite para su implementación.

NOTA: Los casos de prueba que aparecen en este proyecto son solamente de ejemplo. Que usted obtenga resultados correctos con estos casos no es garantía de que su solución sea correcta y de buenos resultados con otros ejemplos. De modo que usted debe probar con todos los casos que considere convenientes para comprobar la validez de su implementación.