DigitalHouse>
Coding School

Certified Tech Developer
The Ultimate Degree

Back End I

Ejercicio mesas de trabajo: patrón flyweight

Objetivo

Realizar el diagrama **UML** y **programar en Java**, implementando patrón **Flyweight** según el siguiente enunciado.
A tener en cuenta:

- Ejercicio individual
- Nivel de complejidad intermedia: 🔥🔥

Enunciado

En un negocio necesitan crear árboles para poder ver cuánto lugar ocuparían. El programa que tiene actualmente posee un elevado consumo de los recursos. Necesita crear 1.000.000 árboles. El proceso para crear los árboles son, cada **árbol** tiene un Alto, Ancho, color y tipo de árbol. Los tipos de árboles que hay son:

- Ornamentales
    - Alto: 200.
    - Horizontal: 400.
    - Color: verde.

- Frutales
    - Alto: 500.
    - Horizontal: 300.
    - Color: rojo.
- Florales
    - Alto: 100.
    - Horizontal: 200.
    - Color: celeste.

El **bosque** es un conjunto de árboles y permitirá **plantar los árboles. ArbolFactory** será el lugar donde se almacenarán los diferentes **tipos de árboles.** Permitirá, antes de crear el objeto, validar si ya existe uno idéntico al que se le está solicitando. De ser así, regresa el objeto existente; de no existir, crea el nuevo objeto y lo almacena en caché para ser reutilizado más adelante.

Se necesita un sistema que muestre 1.000.000 de árboles la mitad rojo y la otra mitad verde y que luego informe por pantalla los árboles y cuanta memoria se está usando.

Usando esta sentencia se podrá ver la memoria usada:

```
Runtime runtime = Runtime.getRuntime();
System.out.println("Memoria usada: " + (runtime.totalMemory() -
runtime.freeMemory()) / (1024 * 1024));
```

Se deben desarrollar las clases necesarias para lograr el proceso explicado de **preparación**.

¡Muchos éxitos!