Patrón Singleton

- **Introducción:**
  El patrón singleton consiste en crear una única instancia de una clase, dicha clase será llamada múltiples veces en diferentes partes del código sin volver a crear los objetos de la clase. Este patrón se logra al declarar la clase constructora como privada y estática (private static class), lo que impide el acceso directo fuera de dicha clase, por ello es necesario la creación de un método que regule el ciclo de instanciación o de vida de la clase constructora; en otras palabras, la misma clase administra su accesibilidad a través del código. Este patrón se encuentra en un área gris dentro la comunidad, ya que el trade-off que brinda es cuestionable, por el aumento de dificultad que le da al código para aumentar su eficiencia en recursos de memoria. (Programación ATS, 2019)

- **Ventajas:**
  - Elimina la necesidad de repetir la creación de objetos de la clase, cada vez que dicha clase es instanciada dentro del programa.
  - Reduce la notación de efectividad de O(N) a O (1) respecto al parámetro de memoria cada vez que se instancia la clase.
  - A diferencia de la mala práctica de utilización de variables globales, el patron singleton si cuenta con un método de regulación de acceso, que no pueden ser modificado.

  (Rodriguez, 2019)
- **Desventajas:**
  - No cumple el principio único de responsabilidad en la clase privada, la clase privada deberá cumplir con el rol asignado mas el control del ciclo de instancia de ella misma.
  - Complica innecesariamente el proceso de creación de pruebas unitarias por la misma restricción de instancia.
  - Elimina la posibilidad de modificación y utilización de diferentes objetos, lo cual no es recomendado en ciertos ensenarías.

  (Creadores Códigos, 2023)

- **Opinión:**
  Este patrón tiene el potencial de ser una herramienta esencial al momento de reducir el uso de recursos de memoria dentro de un gran programa, sin embargo, para el programa de Calculadora pensamos que es irrelevante. El programa de Calculadora se considera que tiene un diseño simple y ligero respecto al uso de recursos, tanto de tiempo como de memoria, por ello el trade-offer que da el programa no es esencial en este escenario. La dificultad extra que brinda al momento de crear las pruebas unitarias no fue recibida con agrado por los integrantes del equipo, ya que estamos acostumbrados a tener instancias pensadas para la ejecución e instancias pensadas para el testing del programa. A pesar del

aspecto negativo que brindo al proyecto no apoyamos la idea que esta metodología sea descrita como un antipatrón, por el simple hecho que complica el proceso de programación y porque maneja un sistema parecido a la metodología de variables globales. Creemos que el patrón singleton tiene potencial para mejorar el uso de recursos dentro de los programas, sin embargo, las características que introduce; limitante en la modificación de los objetos instanciados, ineficiencia en las pruebas unitarias y violación al pensamiento de responsabilidad única; de cierta manera opacan el potencial que puede brindar.

- Referencias:
  - Creadores Códigos. (2023, 10 noviembre). *Lista simplemente enlazada en Java | Estructura de datos en Java* [Video]. YouTube. https://youtu.be/7g39JuMY2kc
  - Programación ATS. (2019, 15 octubre). *Lista doblemente enlazada en Java* [Video]. YouTube. https://youtu.be/GGq6s7xhHzY
  - Rodríguez Portela, A. E. (2019). Patrón de diseño Singleton. *Arquitectura de Software*.