Análisis del Patrón Singleton en la Calculadora

Ventajas:

- el patrón Singleton asegura que solo exista una instancia de una clase en toda la aplicación. Esto es útil para evitar la creación innecesaria de objetos y el consumo excesivo de memoria.
- Al usar Singleton, cualquier parte del código puede acceder a la misma instancia sin necesidad de pasar referencias.
- Como solo hay una instancia, todos los métodos y atributos permanecen consistentes en toda la aplicación.
- Se usa frecuentemente en controladores de bases de datos, gestores de configuración o caches, donde se requiere una única instancia.

Desventajas:

- Como Singleton crea una única instancia en toda la aplicación, es difícil realizar pruebas unitarias sin modificar su estado global.
- Puede llevar a que una clase maneje múltiples responsabilidades que no deberían estar en la misma instancia.
- Al ser accesible globalmente, muchas partes del código pueden depender de Singleton, haciendo que su modificación afecte a varias secciones del programa.

Para la hoja de trabajo 4:

Creemos que no hay necesidad de usar el patron Singleton, mas alla de fines educativos y de aprendizaje, ya que no hay necesidad real de una única instancia ya que cada vez que se quiera evaluar una nueva expresión, se podría crear una nueva instancia de la Calculadora sin afectar el funcionamiento del programa. Por otra parte, esto puede dificultar las pruebas unitarias. La unica forma en la que vemos el uso practico para el patron del singleton, es cuando se quiere tener un historial de todos los los calculos realizados por la calculadora.

Referencias:

Singleton Pattern. (2023). Refactoring Guru. Retrieved from https://refactoring.guru/design-patterns/singleton.