Universidad del Valle de Guatemala

Facultad de Ingeniería

Departamento de Ciencias de la Computación

Cristian Túnchez – 231359

Dulce Ambrosio - 231143

Algoritmos y Estructuras de Datos

Hoja de Trabajo No. 4

---

1. **Ventajas y Desventajas al Utilizar el Patrón Singleton**

   El patrón Singleton es un patrón de diseño que se utiliza para garantizar que una clase tenga una única instancia y proporcionar un punto de acceso global a esa instancia. Si bien puede parecer similar a una variable global en términos de acceso desde cualquier parte del código, tiene algunas ventajas y desventajas distintivas (Canelo, 2023):

   Ventajas del patrón Singleton:

   - Control sobre la instancia: El patrón Singleton proporciona un control estricto sobre cómo se crea y se accede a la instancia única de una clase. Esto puede ser útil para gestionar recursos compartidos o conexiones a bases de datos, por ejemplo (Caules, 2023).
   - Evita la inicialización repetida: Garantiza que solo se cree una instancia de la clase, lo que evita la inicialización repetida y conserva recursos (Caules, 2023).

- Promueve la coherencia del estado: Al tener una única instancia, se garantiza que cualquier modificación en el estado de esa instancia se refleje en todas las partes del código que acceden a ella (KeepCoding, 2022).
- Facilita el reemplazo: Si en el futuro necesitas cambiar la implementación de la clase, puedes hacerlo sin cambiar el código que la utiliza, siempre y cuando mantengas la misma interfaz (KeepCoding, 2022).

Desventajas del patrón Singleton:

- Dificulta la prueba unitaria: Puede hacer que las pruebas unitarias sean más difíciles, ya que el acoplamiento con la instancia única puede interferir con la capacidad de probar componentes de forma independiente (Caules, 2023).
- Puede introducir acoplamiento global: Si se utiliza de manera indiscriminada, puede conducir a un acoplamiento global no deseado, lo que puede hacer que el código sea más difícil de mantener y depurar (KeepCoding, 2022).
- Puede ocultar dependencias: El uso de un Singleton puede ocultar dependencias reales entre clases, lo que dificulta comprender la estructura y las relaciones del código (KeepCoding, 2022).
- Puede generar problemas de concurrencia: Si la instancia Singleton es mutable y se accede concurrentemente desde múltiples hilos, puede conducir a problemas de concurrencia que deben ser manejados adecuadamente (Caules, 2023).

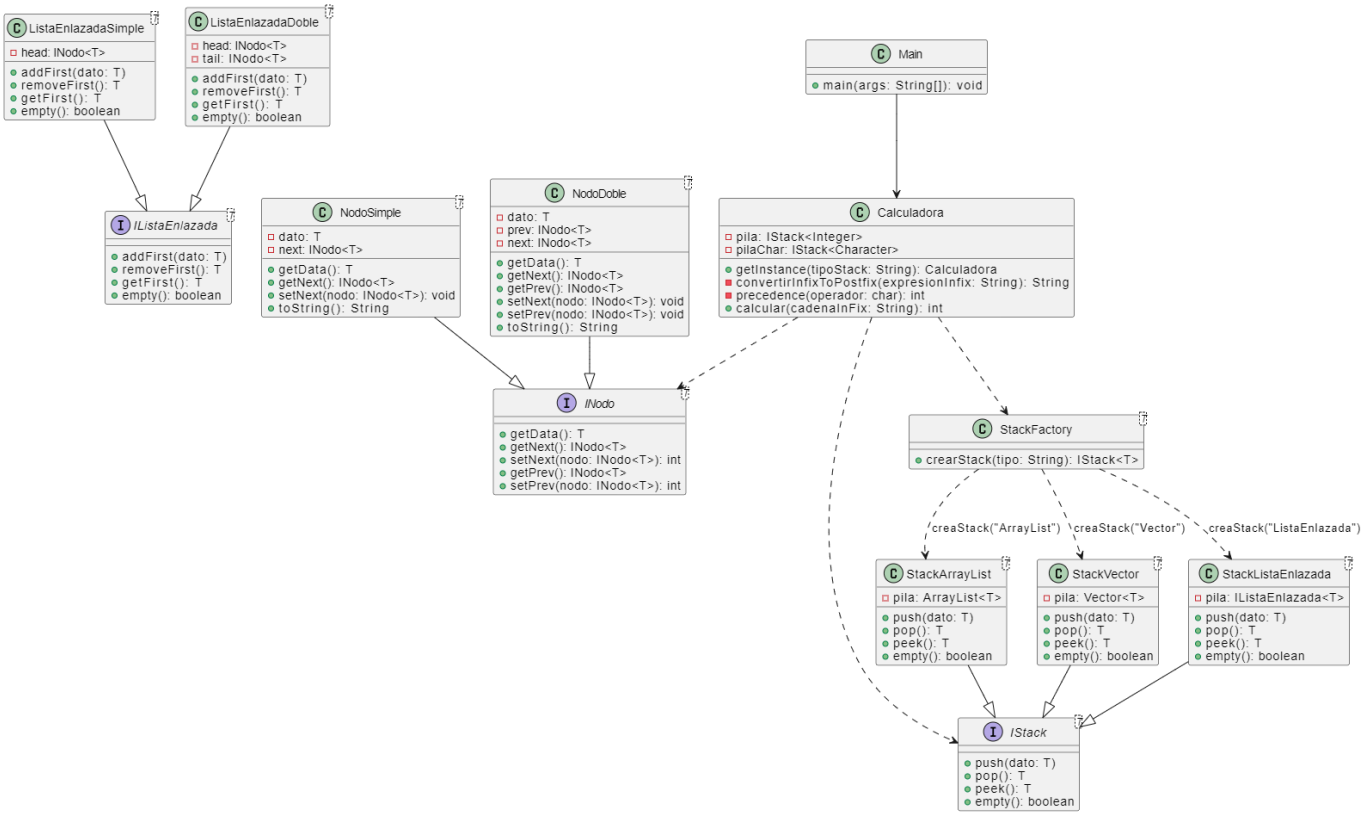
2. **¿Cree que su uso es adecuado en este programa?**

   Sí, consideramos adecuado el uso del patrón Singleton para garantizar que solo exista una instancia de la calculadora en este programa. Esto asegura que en todo momento se esté trabajando con la misma instancia de la calculadora, lo que puede ser útil en situaciones donde se requiera compartir estado o configuraciones entre diferentes partes del programa.

3. **Controlador de Versiones**

   https://github.com/Dulce2004/Hoja-de-Trabajo-4.-AED

4. **Diagrama de Clases (UML)**



Referencias

Canelo, M. M. (5 de septiembre de 2023). *¿Qué son los patrones de diseño de software?* Profile Software Services. https://profile.es/blog/patrones-de-diseno-de-software/

Caules, C. Á. (6 de marzo de 2023). *Ejemplo de Java Singleton (Patrones y ClassLoaders)*. Arquitectura Java. https://www.arquitecturajava.com/ejemplo-de-java-singleton-patrones-classloaders/

KeepCoding, R. (10 de noviembre de 2022). ¿Qué es el patrón singleton? | KeepCoding Bootcamps. *KeepCoding Bootcamps*. https://keepcoding.io/blog/el-patron-singleton/