TP 1 – POO y Patrones de Software

Alumno: Nuñez Souto, Aaron Agustín

1. Los patrones se clasifican en los siguientes tipos principales:
2. **Patrones de creación:** se ocupan de los mecanismos de creación de objetos, tratando de crear objetos de una manera adecuada a la situación.
3. **Patrones estructurales:** Tratan la manera en que los objetos se conectan con otros objetos, para asegurar que los cambios del sistema no requieren cambiar esas conexiones.
4. **Patrones de comportamiento:** Tratan a los objetos que manejan tipos particulares de acciones dentro de un programa. Éstos encapsulan los procesos que deben ejecutarse dentro de la funcionalidad de la aplicación, como interpretar un lenguaje, completar una petición, moverse a través de una secuencia o implementar un algoritmo.
5. Un patrón de software es una abstracción que ayuda a estructurar los diseños de sistemas. Es una forma de escribir y abordar por su nombre una solución o enfoque repetible para un problema de diseño común, es decir, una forma común de resolver un problema genérico.
6. Los patrones de software se clasifican en:
7. Creacionales
8. Abstract Factory
9. Constructor
10. Fábrica
11. Prototipo
12. Singleton
13. Estructurales
14. Puente
15. Composición
16. Decorador
17. Fachada
18. Flyweight
19. Proxy
20. Comportamiento
21. Cadena de responsabilidad
22. Comando
23. Intérprete
24. Iterador
25. Mediador
26. Memento
27. Observador
28. Estado
29. Estrategia
30. Método template
31. Visitante
32. Los patrones de creación se clasifican en:
33. **Constructor**: se utiliza para separar la construcción de un objeto complejo de su representación para que el mismo proceso de construcción pueda crear diferentes representaciones de objetos.
34. **Fábrica**: una clase de utilidad que crea una instancia de una clase a partir de una familia de clases derivadas.
35. **Abstract Factory**: define un método para crear los objetos, que las subclases pueden anular para especificar el tipo derivado que se creará. Por lo tanto, en tiempo de ejecución, a este método se le puede pasar una descripción de un objeto deseado y devolver un puntero de clase base a una nueva instancia de ese objeto.
36. **Prototipo**: se utiliza cuando el tipo de objetos a crear está determinado por una instancia prototípica, que se clona para producir nuevos objetos.
37. **Singleton**: garantiza que una clase tenga solo una instancia y proporciona un punto de acceso global a esa instancia.
38. Ejemplos de uso de patrones de creación:
39. **Constructor**: