Patrones de Diseño

# Objetivos:

* Aplicar patrones de diseño tipo creacionales en la solución de problema, para optimizar su código.

# Parte 1

La concesionaria “TuCarro” es una empresa que se encarga de embalaje, venta y distribución de vehículos, para eso lo ha contratado para desarrollar un software que modele el proceso de ensamblaje y venta de sus vehículos.

Su trabajo es modelar el ensamblaje de vehículos Chevrolet Sail y Chevrolet Sparkt las cuales tienen las siguientes características:

**Chevrolet Sail:**

marca: Chevrolet

Modelo:Sail

Motor: 1.4

almacenamiento: 1000 lb

Carrocería: 1J23KO23

**Chevrolet Sparkt:**

marca: Chevrolet

Modelo:Sparkt

Motor: 1.2

almacenamiento: 500 lb

Carrocería: JWE9EWF8

En esta concesionaria todos los autos ensamblados son almacenados y preparados para su próxima venta.

# Desarrollar

1. Indique que patrones podrían servir dentro del desarrollo de este sistema. (explique)

**Patrón creacional**

**Builder:**

Un auto tiene características similares, pero no siempre es igual con este patrón si cada auto tiene algo que el otro no tiene, ayudaría mucho para realizar una composición de objetos en la clase auto. Es decir, nos permite crear distintos objetos en base a un objeto principal usando un mismo builder. Ejemplo: puede que un auto sail sea full equipo y una chrevolet sparkt no, pero tenga otras características que no tiene el sail.

**Abstract Factory:**

Debido a que una clase tiene la definición de los métodos para la fabricación del auto, hay otra que hereda de ella y construye ese vehículo. Es decir, en el main se crea una instancia de la clase que implementa la fabricación del auto; la cual usa y crea para luego devolver varios tipos de objetos.

**Patrón de comportamiento**

**Strategy:**

Para un caso general y realizando una pequeña modificación en el código puede colocar como referencia a que siempre llame para Chevrolet y se podrían colocar para varias marcas de autos, también si deseamos mostrar por costo del vehículo desde menor a mayor. Así en los tiempos de respuesta del algoritmo en función de algún parámetro, se ejecutará la estrategia que se desee en el momento.

Aplicar Strategy es buena idea ya que prioriza 2 principios SOLID que son: responsabilidad única, abierto y cerrado. Ayudando mucho en un futuro si se desea extender nuevas funciones en el programa.

1. Implemente el sistema
2. (Adicional) Agregue un nuevo vehículo diferente a los que están en el ejemplo