Taller Integrador: Patrones de diseño

# Sistema de Requerimientos Académicos.

# Objetivos

* Evaluar que patrones de diseño se deben utilizar en el desarrollo de un sistema.
* Aplicar varios patrones de diseño dentro de un mismo sistema.

# Requerimientos del sistema:

Se desea desarrollar un sistema de cajero automático que permita manejar consultas de saldo, retiros y depósitos de cuentas bancarias, para esto se le provee una clase llamada Account, la cual es la única que no debe ser modificada, pero también se proveen 3 clases más, que pueden ser modificadas a conveniencia, y sobre las que se debe aplicar los patrones de diseño elegidos.

# Desarrollar

1. Indique para cada uno de los patrones estudiados si pudiera o no servir dentro del desarrollo de este sistema. (explique)
   1. Creacionales.

**Abstract Factory:** No se puede aplicar este patrón debido a que el cajero automático trabaja bajo un solo sistema y no es necesario crear familias de objetos como especifica este patrón ya que cada objeto tiene una funcionalidad única.

**Factory Method:** No se puede aplicar tampoco este patrón por la sencilla razón de que se complicaría dejar el manejo de un objeto a subclases.

**Singleton:** Si se puede aplicar este patrón debido a que se crea una única instancia de la clase del AmtUK.

* 1. Estructurales.

**Adapter:** No se puede utilizar este patrón porque la funcionalidad es única del ATM no se necesita agregar cosas extras a los métodos.

**Composite:** No se puede utilizar este patrón porque cada objeto puede funcionar de manera independiente sin la necesidad de estar creando algo más complejo.

**Decorator:** Se puede aplicar este patrón para añadir funcionalidad al objeto de cuenta de modo que esto permite no crear más clases que hereden de la primera, sino que una nueva funcionalidad se puede asociar a la primera.

* 1. De Comportamiento.

**Chain of Responsability:** se lo aplica para poder hacer que los manejadores realicen correctamente las peticiones de ingresar o retirar dinero de la atm.

**Iterator:** se lo aplica para poder buscar una cuenta en específico usando el id al tener un conjunto de cuentas.

**Memento:** no se lo aplica debido a que no se desea guardar el estado antiguo o posterior de algún objeto del sistema después de un cambio.

**Strategy:** no se puede aplicar debido a que no se tienen varios procesos que tengan el mismo fin/objetivo.

1. Diseñe un diagrama de clases del sistema, aplicando los patrones elegidos.

