**IMPLEMENTE EL CÓDIGO FUENTE EN EL LENGUAJE DE PROGRAMACIÓN DE SU PREFERENCIA PARA RESOLVER EL SIGUIENTE EJERCICIO (debe subirlo al repositorio)**

La implementación debe ser hecha sin contemplar solicitud de datos al usuario ni interfaz gráfica, si no, con datos ficticios de prueba, escritos desde los métodos de inicio de los programas. Se deben hacer las impresiones del caso para demostrar la funcionalidad y correctitud de los programas.

EL PROBLEMA

Diseñe un programa que maneje agendas de Contactos y Eventos. Se debe considerar separar la información de la persona como tal asociada a eventos o contactos, del contacto o evento como tal.

Los datos de la persona deben ser datos simples pero que permitan describir de forma clara a una persona.

Los contactos pueden ser simples o específicos. Los específicos serán ampliaciones del contacto simple incluyendo información relacionada con datos familiares y datos empresariales (o sea contactos Familiares y contactos empresariales)

Los Eventos, de forma similar, deben ser eventos simples y eventos específicos con la diferencia de que se deja a criterio del programador el tipo de evento específico a diseñar e implementar.

Incluya como atributo de la clase Agenda UNA SOLA estructura de datos que permita almacenar todos los posibles contactos y eventos de una agenda en cualquier orden que se ingresen. Añada la declaración de los métodos que se necesiten en cada clase para que sea factible, añadir, eliminar o modificar la información contenida en la agenda (contactos o eventos de cualquier tipo).

Añadan a las clases que consideren necesario un método “toString” que facilite la posterior presentación de los resultados en consola.

Cree la clase principal donde se implementará la agenda con elementos “quemados” en cualquier orden.

Posterior a esto, utilice una función filter al estilo de programación funcional determinística con datos inmutables, para imprimir los elementos tipo Evento separados de los elementos tipo Contacto de la Agenda.

Finalmente, modifique el código realizado para agregar dos patrones de diseño: Singleton y Abstract Factory al problema en donde usted considere mejor apliquen (documente en el código fuente, la justificación del por qué considera que es útil usar cada patrón donde lo utilizó). Específicamente para el patrón Singleton, documente en el código fuente también, las diferencias entre un Eager Singleton y un Lazy Singleton y defina cual le parece que sea mejor utilizar en la implementación del problema y por qué?

Ref Example: <https://www.linkedin.com/pulse/two-types-singleton-design-pattern-lazy-eager-arifuzzaman-tanin/>