Tarea Tema 2-3

En esta tarea **consolidaremos lo aprendido en el tema 2 con patrones de diseño y en el tema 3 con el control de versiones**. Se entregará en un **zip apellidosnombreTarea2-3.zip** que **contendrá el proyecto**. Podéis llamarle al **proyecto, proyectoexamen**. Nos basaremos en el ejercicio que hicimos en el examen.

Parte I. Partiendo de la siguiente información construir la clase estudiante con el constructor con todos los parámetros y los getters y los setters

```
public abstract class Estudiante {
	private String nombre="";
	private String apellidos="";
	private int edad;
	private String direccion="";
	private long telefono;

public abstract void realizarMatricula();


	public Estudiante(String nombre, String apellidos, int edad,
String direccion, long telefono)
```

1. **Crear la clase EstudianteSecundaria que herede de Estudiante que añadirá a sus atributos curso (una cadena), y rama *CIENCIAS, LETRAS o SOCIALES* con constantes.**

2. **Construir el constructor con parámetros para EstudiantesSecundaria.**

3. **Construir el toString para EstudianteSecundaria para que devuelva**

```
EstudianteSecundaria [ nombre=Carlos, apellidos=Latre, edad=16,
direccion=Calle carr, telefono=9686660, curso=4 ESO, rama=1]
```

4. **Implementar el método abstracto realizarMatricula para que devuelva:**

Se matricula en 4 ESO rama Ciencias

5. Crear un **repositorio llamado repositoriotema3tarea** y añadid el **control de versiones GIT en Eclipse**. Etiquetarlo como **versión 1.**
6. Crear **una nueva rama llamada Rama version1** para guardar esta **primera versión del proyecto.**

Parte II. Crearemos el EstudianteFP y EstudianteBuilder. Continuaremos nuestro desarrollo en la **rama Master**.

1. **Crear Estudiante FP** que herede de Estudiante.
2. Tras **acabar el EstudianteFP, realizar un Commit** y etiquetarlo como **Version 2.0.**
3. **Añadid el fichero .gitignore conforme a las especificaciones de los apuntes.**

```
public class EstudianteFP                    {


    private String especialidad;
    private String nivel;
    private boolean practicasRealizadas=false;
```

Y tenemos la clase EstudianteBuilder:

4. **Crear la clase EstudianteBuilder** para **crear EstudianteFP o EstudianteSecundaria**, según el tipo ***SECUNDARIA O FP.***
5. Crear una **clase Main como punto de entrada a la aplicación** y **comprobar que funciona** creando un estudiante secundaria y otro FP.
6. **Realizar un commit** cuando todo funcione etiquetándolo como **versión 2.1.0**
7. Crear una nueva **rama** llamada Version2.

Parte III. Introducimos estudiante tutorizado. Continuamos nuestro desarrollo en la rama Master.

1. Crear la clase **EstudianteTutorizado**, única en cada país, **con el atributo pais**. El **Estudiante será único** y se creará con un **patrón Singlenton. Implementar los métodos de creación de EstudianteTutorizado suponiendo que hereda de Estudiante.**
2. Añadid al builder la posibilidad de crear un EstudianteTutorizado con constante ***TUTORIZADO.*** El estudianteTutorizado se seguirá creando con el singleton dentro del builder.
3. **Realizar un commit etiquetado como Version 2.2.0 Crear una nueva rama**

versión2-2.

Parte IV. Cambiando el Builder por un Factory. Seguimos nuestro desarrollo en la rama Master.

1. **Borramos la Clase EstudianteBuilder y añadimos la clase EstudianteFactory**
2. **La clase FactoryEstudiante** que creará estudiantes de secundaria o de FP según el tipo EstudianteFactory ***SECUNDARIA, TUTORIZADO O FP*** y los parámetros pasados. Añadid las constantes necesarias.
3. **Crea en la factoría un método creaEstudiante** (parámetros necesarios, .., tipo) para crear un estudiante según el tipo.
4. **Probar en la clase Main que podéis crear un Estudiante de cada tipo, usando un objeto de tipo EstudianteFactory.**

Parte V. Comentar con Javadoc vuestros módulo, paquetes, clases y métodos, y **generar la documentación** conforme a lo realizado en clase. **Utilizar la etiqueta @version y @ since a nivel de clase** teniendo en cuenta **vuestra versión en el control de versiones** Git, para que sean coherentes.