Práctica 2

Introducción al Diseño Orientado a Objetos

Inicio: Según el grupo, la semana del 12 de febrero.
Duración: 2 semanas.
Entrega: Semana según el grupo del 26 de febrero. Grupos del lunes tarde: viernes 2 de marzo.
Peso de la práctica: 15%

El objetivo de esta práctica es introducir los conceptos básicos de la programación orientada a objetos desde el punto de vista del diseño, así como de su correspondencia en Java, con particular énfasis en el diseño de clases con relaciones de herencia, asociación y composición mediante diagramas de clases en UML.

Apartado 1 (3 puntos):

Nos han encargado desarrollar en lenguaje Java las clases de un diagrama de clases UML descrito abajo. Estas clases se integrarán en una aplicación destinada a la gestión de una cadena de autoescuelas. La autoescuela da clases a los alumnos que tienen que aprobar exámenes teórico y práctico. Por ello, imparte clases de teoría y clases prácticas, y participa en el proceso de examinación.

De forma más precisa, la clase abstracta *Clase* representa las posibles clases que se imparten en cada centro de la autoescuela y que son de varios tipos:

- teóricas,
- prácticas,
- ejercicios en polígono, y
- exámenes.

Las clases teóricas se imparten en un aula; las otras con un coche.

Todas las *clases* tienen (entre otros)

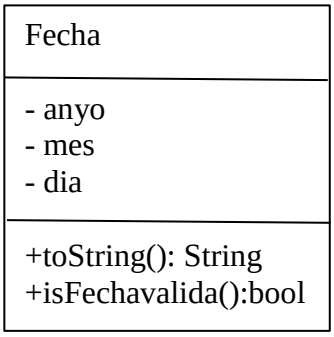
- una fecha y hora en la que se imparten,
- lugar,
- un precio básico,
- una duración (minutos), y
- alumnos.

Los datos que se guardan para cada alumno incluyen datos personales (DNI, nombre, apellidos, teléfono, fecha de nacimiento), su fecha de matriculación en la autoescuela, el tipo de carnet de conducir que pudiese tiene, la categoría de carnet para el que se ha matriculado, la fecha de aprobación del examen de teoría y la fecha de aprobación del examen de prácticas, así como un posible descuento de precio individual en tanto por ciento.

Cada clase tiene su precio según la categoría de carnet. Además, las clases de ejercicios y prácticas tienen coste adicional, según la categoría de carnet. Las clases *Teoría*, *Práctica*, *Ejercicios*, *Examen* recogen la funcionalidad de cada uno de los tipos de clase que se imparten en la autoescuela. La fecha, como es de esperar, está formada por año, mes y día; en esta práctica vamos a implementarla a pesar de que existen ya varias clases implementadas en varias librerías.

Ejercicio en clase: Dibujar los diagramas UML correspondientes a las clases de alumnos, fecha y clases.

Por ejemplo, una posible implementación puede tener el siguiente diagrama UML:



Las operaciones que deben implementarse en las clases Java apropiadas son las siguientes:

- Comprobar si la fecha de la clase es una fecha válida.
- Escribir la información de cada clase con el formato adecuado que puede verse en el código de prueba que se muestra a continuación.

Ejercicio en clase: Dibujar en el diagrama de las relaciones entre alumno y fecha.

Se pide:

1. Dibujar las relaciones entre la clase que corresponde al *alumno* y la *fecha*.
2. Desarrollar las clases *Alumno* y la clase *Fecha*.
3. Desarrollar en Java las clases especificadas en el diagrama anterior de forma que tu implementación permita compilar y ejecutar el siguiente programa de prueba obteniendo la salida que se indica más abajo.

Tester

```
package p2.autoescuela;

import p2.autoescuela.clases.Alumno;

/**
 * Tester el primer apartado de la P2
 * @author Profesores ADS
 */
public class TesterAutoescuela {
   public static void main(String[] args) {
        Alumno a1=new Alumno("3141243T","Jose", "Jimenez",2016,10,3,"A");
        Alumno a2=new Alumno("3141243T","Sandra", "Goya",2016,22,3,"C");
        Alumno a3=new Alumno("3141243T","Carlos","Pascual",2015,2,29,"E");
        Fecha c1= a1.getFechaMatr();
        Fecha c2= a2.getFechaMatr();
        Fecha c3= a3.getFechaMatr();

        System.out.println("isValida <" + c1 + "> ? " + c1.isFechaValida());
        System.out.println("isValida <" + c2 + "> ? " + c2.isFechaValida());
        System.out.println("isValida <" + c3 + "> ? " + c3.isFechaValida());
        System.out.println("Datos de alumno 1 " + a1);
        System.out.println();
}
```

Siendo la salida esperada de la ejecución del programa la siguiente:

```
isValida <2016-10-3> ? true
isValida <2016-22-3> ? false
isValida <2015-2-29> ? false

Datos del alumno 1:
        Nombre: Jose
        Apellido: Jimenez
        DNI: 3141243T
        Fecha Matricula: 2016-10-3
        Tipo carnet: A
```

Podéis añadir el código oportuno para comprobar el funcionamiento de todas partes de su implementación **después del código indicado en main**.

Apartado 2 (4.5 puntos):

El diagrama UML del apartado anterior contendrá una pequeña parte del sistema de gestión de la cadena de autoescuelas. El objetivo de este apartado es diseñar el módulo completo teniendo en cuenta la siguiente información:

Las autoescuelas tienen un código interno, una dirección y un encargado. Además, disponen de una serie de aulas para impartir teoría y realizar tests, y una flota de coches para las clases de prácticas.

Los alumnos que se matriculan en las autoescuelas tienen que proporcionar la información que se menciona en el apartado 1. Como mínimo están los datos personales, su fecha de matrícula y el tipo de carnet que aspira cada alumno.

Las clases que se imparten en la autoescuela son de los tipos y características que se mostraron en el apartado 1. Debes relacionar esta parte con el resto de elementos del módulo. Por ejemplo, cada clase es impartida por un único profesor de la autoescuela.

La información de interés sobre los profesores incluye sus datos personales (iguales a los de alumnos), su número de la Seguridad Social, la antigüedad en la autoescuela, el sueldo y las categorías de carnets que pueden impartir. Por norma general, un profesor de la cadena de autoescuelas trabaja en una de ellas, pero si lo solicita, puede cambiar a otra autoescuela de la cadena, de manera que interesa también saber la fecha de inicio y de fin de actividad del profesor en cada autoescuela. El salario que se paga al profesor está constituido de un sueldo base y de unos complementos que dependen del número de clases prácticas que ha impartido en cada mes. El sueldo base es de 900 euros y cada clase práctica se paga a 20 euros.

Para las clases prácticas, cada autoescuela dispone de una serie de coches, que tienen la categoría de carnet correspondiente, una matrícula, un número de bastidor, una marca y un modelo. El funcionamiento de la clase práctica permite que puedan ir un máximo de dos alumnos en el coche, de manera que se alternen en el recorrido.

En cuanto a las clases de ejercicios y en polígono, los alumnos pueden asistir a ellas y realizar el ejercicio tipo test, obteniendo una nota que se guarda para tratamientos posteriores ajenos a este ejercicio.

Se pide:

Realizar un diagrama de clases UML para el problema de la gestión de autoescuelas.

Nota: En el diagrama de clases en UML no es necesario incluir constructores, ni métodos *getters* y *setters*.

Apartado 3 (2.5 puntos):

La cadena de autoescuelas para la que se ha realizado el diseño del apartado anterior ha llegado a un convenio con una compañía de talleres de reparación de vehículos, para que se encargue del mantenimiento de su flota de coches. A partir de ahora, cuando uno de los coches tenga un problema o deba pasar una revisión rutinaria se enviará a un taller de esta compañía, del que interesa saber su nombre comercial, dirección, encargado y teléfono. Los trabajos que se realicen en el coche serán por tanto de dos tipos: reparación o revisión. En el primer caso, se debe obtener un presupuesto y una lista de los trabajos necesarios, así como las piezas que harían falta para reparar el vehículo. Además, en caso de que se acepte la reparación se deberá calcular el precio final, incluyendo un IVA del 21% y un descuento del 15% solo para las piezas en virtud del convenio entre ambas compañías. El caso de las revisiones es bastante más simple, ya que el acuerdo implica aplicar una cantidad fija, siempre que los trabajos sean los especificados de común acuerdo entre las partes: cambio de aceite, filtros y pastillas de freno.

Se pide:

Añadir las modificaciones necesarias al diseño UML realizado en el apartado anterior para representar la nueva funcionalidad de reparación de vehículos.

Sólo se deberá entregar el diagrama de clases UML modificado, no es necesario entregar la implementación *Java*.

Apartado 4 (Opcional, 1 punto):

Implementa en Java las clases *Profesor* y *Autoescuela*, de manera que pueda obtenerse la información de los periodos de tiempo en que cada profesor ha impartido clases en una autoescuela cualquiera de la cadena. Para la implementación de Java utiliza la parte del diseño de tu diagrama UML que recoge esta funcionalidad, omitiendo aquellos aspectos que no sean necesarios. Por ejemplo, el programa debería ofrecer información como:

El profesor Antonio Pérez García trabajó en la autoescuela Arenal desde el 2/4/2004 hasta el 4/10/2009.

Normas de Entrega:

- Se deberán entregar los apartados 1, 2, 3 y, opcionalmente, el 4.
- La entrega la realizará uno de los alumnos de la pareja a través de Moodle.
- Si el ejercicio pide código Java, se entregará el código fuente, y la documentación generada con *JavaDoc*.
- Si el ejercicio pide un diagrama de diseño, se deberá entregar en PDF junto con una breve explicación (dos o tres párrafos a lo sumo).
- Se debe entregar un único fichero ZIP / RAR con todo lo solicitado, que deberá llamarse de la siguiente manera: GR<numero_grupo>_<nombre_estudiantes>.zip. Por ejemplo Marisa y Pedro, del grupo 2261, entregarían el fichero: GR2261_MarisaPedro.zip.
- La estructura de los ficheros entregados deberá estructurarse en directorios, uno por cada apartado.