Examen programación 2º Parcial

Crear un nuevo proyecto llamado "Ex2-nombreAlumno". Las clases se crean siempre dentro de algún paquete.

Diseñar un programa para gestionar un colegio. Se debe almacenar el personal (profesor o alumno) y las asignaturas. Para ello debemos diseñar una serie de clases sin método main() que definen el comportamiento del programa. Finalmente se solicitará mostrar el uso del programa en una clase con método main().

Requisitos obligatorios (Crear las clases necesarias para desarrollar el programa):

- (0,4) Las asignaturas tienen un código, un nombre, una cantidad de horas anuales, un profesor y una lista de alumnos.
- El personal del colegio son únicamente elementos de tipo profesor o de tipo alumno, los cuales tienen algunas características en común. Aplicar una herencia para estos elementos:
  - (0,2) Los profesores tienen dni, nombre, y departamento (String).
  - (0,2) Los alumnos tienen dni, nombre y fechaMatriculación (LocalDate) y nacionalidad.
- (1,2 pto) La clase "GestionColegio" almacena los datos del colegio **utilizando únicamente dos estructuras de datos**: una para almacenar las asignaturas, y otra para almacenar tanto los elementos de tipo profesor como los de alumno (en la misma estructura, habiendo diseñado la herencia correspondiente).

Otros requisitos:

Además, la clase GestionColegio tiene una serie de operaciones (métodos en su mayoría) para gestionar las asignaturas, los profesores y los alumnos. Se pueden crear todos los métodos que el alumno considere necesarios, aparte de los siguientes:

- (0,75 ptos) Permite listar solo los profesores, o solo los alumnos.
- (0,75 ptos) Permite dar de alta profesores, y alumnos dando valor a todos sus datos. Todos los datos que reciben estos métodos son de tipo String. Recordar que ambos se guardan en la misma estructura de datos.
- (1 pto) Permite dar de alta una asignatura. Se da de alta a partir de los datos: código, nombre, nº de horas, y el dni del profesor. Para dar de alta la asignatura hay que comprobar previamente 3 cosas: (0,3) que no exista otra asignatura en la lista con su mismo código, (0,3) que el profesor también exista en la lista de profesores, y (0,3) que el dni dado sea de un profesor. Si se dan las condiciones se crea la asignatura y se añade a la lista.
- (0,5 ptos) La lista que contiene las asignaturas está siempre ordenada por el nº de horas.
- (1 pto) Permite matricular alumnos en la asignatura. Esta operación se lleva a cabo dando el código de la asignatura y el dni del alumno. Para poder realizar la matriculación se debe comprobar que (0,3) la asignatura existe y que el (0,4) dni facilitado corresponde a un alumno.
- (0,75 ptos) Permite mostrar por consola el expediente del alumno. Consta de los datos del alumno y los datos de las asignaturas en las que está matriculado. (String dni)
- (0,75 ptos) Permite mostrar la ficha de un profesor, la cual consta de sus datos, los datos de las asignaturas que imparte, junto con el número de alumnos que tiene cada una. (String dni)
- (0,75 ptos) Permite listar todos los alumnos de un año concreto y de una asignatura concreta (recibe int anno y String asignatura).
- (0,5 ptos) Una operación permite guardar todos los datos (asignaturas, profesores y alumnos) en fichero.
- (0,5 ptos) Una operación permite cargar todos los datos desde fichero. No se puede valorar si no funciona guardar.

(0,75) Por último crear una clase que tenga un método main() y nos permita probar el programa realizando las siguientes operaciones. (Si no se han implementado todas las operaciones, se valorará proporcionalmente).

1. Crear 5 alumnos, 2 profesores y 3 asignaturas.
2. Listar solo los profesores y solo los alumnos.
3. Matricular algunos alumnos en las asignaturas.
4. Mostrar el expediente de dos alumnos.
5. Mostrar la ficha de los dos profesores.
6. Listar los alumnos de un año y asignatura.
7. Guardar los datos, y comprobar que cargan.