Relación de Ejercicios de Ficheros (3)
Ficheros con Registros

Tenemos el siguiente registro:

```
public class FichaAlumno
{
    public string nombre;
    public int edad;
    public double calificacion;
}
```

y una lista formada a base de dichos registros:

```
List<FichaAlumno> listaAlumnos = new ArrayList<FichaAlumno>();
```

Escribiremos un programa con un menú con varias opciones. Después de ejecutarse cada opción, volverá al inicio del menú. Cada opción será una de las funciones siguientes:

1. Escribe una función **LeeAlumnoLista** a la que le pasas la lista de alumnos y te pide un nuevo alumno desde el teclado, cuyos datos se añadirán a la lista.

2. Escribe una función **ImprimeListaAlumnos** a la que la pasas la lista de alumnos y te la imprime por pantalla.

3. Escribe una función **EscribeFicheroAlumnosBinario** a la que le pasas la lista y el nombre del fichero y te escribe la lista en el fichero. La estructura del fichero será la siguiente:
    - Al principio habrá un entero que será el número de alumnos que hay en la lista.
    - Después irán los registros, escribiéndose un *string* para el nombre, un *int* para la edad y un *double* para la nota.
    - Iremos escribiendo todos los registros uno a uno hasta el final.

4. Escribe una función **LeeFicheroAlumnosBinario** a la que le pasas una lista y el nombre del fichero y leerá la lista desde el fichero. El fichero tendrá la misma estructura que el del ejercicio anterior (evidentemente). La lista que nos pasen la borraremos antes de leer los datos del fichero.

   Para poder meter el alumno en la lista primero tendremos que crear un registro de tipo FichaAlumno:

   ```
   FichaAlumno fa = new FichaAlumno();
   ```

   Guardaremos los tres valores que hemos leído en el registro y lo añadiremos a la lista.

5. Escribe la función **EscribeFicheroAlumnosTexto** similar a la función del ejercicio 3 pero usando un fichero de texto. La estructura del fichero será similar, pero ahora guardaremos un valor en cada línea. Ejemplo:

```
2
Pedro
23
7,2
Juan
15
2,1
```

6. Escribe la función **LeeFicheroAlumnosTexto** similar a la función del ejercicio 4 pero que funciona con los ficheros de texto del ejercicio anterior.

Los ficheros CSV (Comma Separated Values) (Valores Separados por Comas) son la forma más simple de guardar una tabla de una base de datos. Cada registro va en una línea y los valores de cada registro van separados por comas u otro valor de separación (por ejemplo, punto y coma).

7. Escribe la función **EscribeFicheroAlumnosCSV**. En este caso, no vamos a guardar el número de registros en la primera línea, ya que los ficheros CSV estándar no lo hacen. Como carácter separador usaremos el punto y coma, ya que uno de los datos que usamos ya contiene comas.
   Un ejemplo del fichero sería:

```
Pedro;23;7,2
Juan;15;2,1
```

8. Escribe la función **LeeFicheroAlumnosCSV** que lee los datos del fichero anterior en una lista. Para separar los datos, lo más fácil es usar la función *Split* de las cadenas. Con esto conseguiremos un array de cadenas compuesto por tres cadenas: “Nombre”, “Edad”, “Calificación”, que tendremos que convertir al tipo de dato adecuado y meter en un registro *FichaAlumno* y luego en la lista.