Ejercicios Persistencia de Objetos

Descargar estos ejercicios

Índice

☑ Ejercicio 1

> Aquí va enunciado de Xusa

Ejercicio 2

Vamos a copiar el ejercicio anterior pues partiremos del mismo. Pero en este caso vamos a ampliar la clase Alumno y vamos a serializarla a binario de forma manual.
1. Añadiremos la clase Nota:
a) Dentro definirá el tipo enumerado Modulo con los valores ED, PROG, SI, FOL, LM y GBD
b) La clase definirá los campos modulo de tipo Modulo, nota de tipo ushort y trimestre de tipo ushort y valores posibles 1, 2 y 3.
c) Definiremos accesores o getters para los 3 campos.
d) Un constructor que reciba todos los campos y un constructor copia.
e) El método void Serializa(Stream flujo) que usará un adaptador BinaryWriter para el flujo y guardará en el mismo campo a campo del objeto nota que estemos serializando. Si se produce un excepción, la capturaremos y la relanzaremos añadiendo un mensaje.
f) El método static Nota Deserializa(Stream flujo) que usará un adaptador BinaryReader para el flujo y leerá los campos de la nota en el mismo orden en que los serializamos. Gestionaremos cualquier excepción de forma análoga al método serializa.
2. Cambios en la clase Alumno:
a) Borraremos los métodos GuardaCSV y LeeCSV.
b) Añaremos una nuevo campo que será una array de notas (clase Nota) que inicializaremos a null en el constructor.
c) Añadiremos un método para añadir notas al array (dimensionándolo si procede) haciendo una

copia de la nota al guardarla en el array.
d) Modificaremos el método ToString para que muestre los datos del alumno de forma similar a esta.
1 Pérez Pepa
1º 2º 3º
ED 4 2 4
PROG 9 8 3
…
e) Por último añade la serialización y deserialización de Alumno de forma análoga a cómo la hemos relizado con su nota.
Nota: Posiblemente necesites un nuevo constructor que deberá ser privado.
3. Modifica el programa principal para encajar las nuevas definiciones propuestas. Para introducir las notas por módulo y trimestre, puedes hacerlo de forma aleatoria através del siguiente método void AñadeNotasAleatorias(Alumno a).

Ejercicio 3

> Vamos a copiar el ejercicio anterior pues partiremos del mismo. Pero en este caso vamos a realizar la serialización de forma 'automática' a través de la clase `BinaryFormatter` y el atributo `[Serializable]` como describe en los apuntes.
> Indica con un comentario en el programa principal los métodos que has tenido que quitar y por qué.

Ejercicio4 (Ampliación larga)

> Se propone añadir también la clase notas al ejercicio de guardar y recuperar alumnos de un CSV. Sin embargo, en este caso las notas se guardarán en otro fichero denominado `notas.csv` de tal manera que se guardará junto a cada nota el nia del alumno al que pertenece (a modo de clave ajena en DB).
> Plantéate qué habría que hacer para añadir una opción de modificación de datos de un alumno.