Tema 1: Acceso a ficheros
Boletín 2: Flujos binarios I

Primero: Plantilla (no se entrega)

Crea una clase llamada ***Plantilla.java*** con los métodos que se detallan a continuación. Estos métodos deberán hacer escritura/lectura de tipos de datos primitivos (***int***, ***String***, ***float***, etc.) en un fichero binario llamado ***plantilla.dat*** (el fichero estará ubicado en la carpeta ***src/datos*** del proyecto). Usaremos acceso secuencial. La clase podrá contar con un método ***main()*** para hacer pruebas y que en ningún caso será evaluado.

- Método ***nuevoEmpleado***:

```
public static void nuevoEmpleado (String nombre, int edad, double
sueldo, boolean jubilado)
```

  Este método debe escribir los cuatro datos del empleado al final del fichero ***plantilla.dat***. Si el fichero no existe previamente, lo inaugurará con esos datos.

- Método ***muestraEmpleados***:

```
public static void muestraEmpleados ()
```

  Este método mostrará los cuatro datos de todos los empleados que hay en el fichero, un empleado por línea.

- Método ***muestraEmpleado***:

```
public static void muestraEmpleado (int orden)
```

  Este método mostrará los cuatro datos del empleado indicado. Consideramos que el primer empleado se corresponde con el número 1. Si no existe el empleado solicitado, se informará al usuario de ello.

- Método ***eliminaEmpleado***:

```
public static void eliminaEmpleado(int orden)
```

  Este método elimina un empleado (borrado físico) a partir del número de orden. Si no existe el empleado se mostrará un mensaje indicándolo.

Segundo: Plantilla2 (no se entrega)

Crea una clase llamada ***Plantilla2.java*** con los métodos que se detallan a continuación. Estos métodos deberán hacer escritura/lectura de objetos en un fichero binario llamado ***plantilla2.dat*** (el fichero estará ubicado en la carpeta ***src/datos*** del proyecto). Usaremos acceso secuencial. La clase podrá contar con un método ***main()*** para hacer pruebas y que en ningún caso será evaluado. Tendrás que hacer uso de la clase ***Empleado*** que se te proporciona.

- Método ***nuevoEmpleado***:

```
public static void nuevoEmpleado (Empleado emp)
```

  Este método debe escribir el objeto que recibe al final del fichero ***plantilla2.dat***. Si el fichero no existe previamente, lo inaugurará con esos datos.

- Método ***muestraEmpleados***:

```
public static void muestraEmpleados ()
```

  Este método mostrará los cuatro datos de todos los empleados que hay en el fichero ***plantilla2.dat***, uno por línea.

- Método ***muestraEmpleado***:

```
public static void muestraEmpleado (int orden)
```

Este método mostrará los cuatro datos del empleado indicado. El primer empleado se corresponde con el número 1. Si no existe el empleado pedido, se informará al usuario de ello.

- Método ***eliminaEmpleado***:

```
public static void eliminaEmpleado(int orden)
```

Este método elimina un empleado (borrado físico) a partir del número de orden. Si no existe el empleado se mostrará un mensaje indicándolo. El acceso al fichero en este ejercicio será secuencial.

Tercero: Colegio (no se entrega)

Crea una clase llamada ***Colegio.java*** con los métodos que se detallan a continuación. Estos métodos deberán hacer escritura/lectura de tipos de datos primitivos (***int***, ***String***, ***float***, etc.) en el fichero binario ***alumnado.dat*** (el fichero se ubica en la carpeta ***src/datos*** del proyecto), que será accedido de forma aleatoria. La clase podrá contar con un método ***main()*** para pruebas y que en ningún caso será evaluado.

- Método ***nuevoAlumno***:

```
public static void nuevoAlumno(int id, String nombre, float calificacion)
```

Este método añade un nuevo alumno en el fichero ***alumnado.dat***, en la posición que indica el identificador. Si ya existe un alumno con ese identificador o es un identificador incorrecto (menor que 0), avisa y no lo inserta.

El ***id*** de alumno será siempre un número mayor o igual a 1. Cada alumno estará situado, dentro del fichero, en la posición que indica su ***id***. Los alumnos que no existan tendrán el campo ***id*** con valor 0 (borrado lógico). El nombre del alumno tendrá una longitud de 20 caracteres.

- Método ***alumno***:

```
public static void alumno(int id)
```

Este método muestra en pantalla los datos de alumno cuyo identificador se pasa como argumento. Si no existe el alumno, lo comunica.

- Método ***cambiaNota***:

```
public static void cambiaNota(int id)
```

Este método sube 0,5 la calificación del alumno indicado, sin superar el valor 10. Si no existe el alumno, lo comunica.

- Método ***borraAlumno***:

```
public static void borraAlumno(int id)
```

Este método elimina el alumno indicado con un borrado lógico (cambiando su id a 0). Si no existe el alumno, lo comunica.

Cuarto: Números mágicos (entrega obligatoria)

Como ya sabemos, los ficheros binarios pueden ser de tipos muy diversos (imagen ***PNG***, vídeo ***MPEG***, comprimidos ***ZIP***, comprimidos ***RAR***, etc). Como hemos visto en el tema, cada fichero tendrá una extensión y un número mágico según su tipo. Los números mágicos pueden ser tan cortos o tan largos como se quiera pero en este ejercicio solamente vamos a trabajar con números mágicos de 4 bytes de tamaño y vamos a suponer que todos los números mágicos tienen ese tamaño.

Se te proporciona de una clase llamada ***FileType*** cuyos objetos contienen extensión, descripción y número mágico de un tipo de fichero. Dispones además de un fichero llamado ***tipos.dat*** que contiene objetos de la clase ***FileType***. Debes colocar ese fichero en la carpeta ***src/datos***.

Por ejemplo, uno de los tipos almacenados en el fichero ***tipos.dat*** es el tipo "***datos comprimidos con poca pérdida***" cuya extensión es ***zip*** y que tiene como número mágico los 4 bytes: ***50 4B 03 04***. Si quieres saber qué otros tipos hay en ese fichero puedes hacerte un método que lea todos los objetos ***FileType*** y los muestre en consola.

En este ejercicio tendrás que crear una clase llamada ***Magico*** con un método ***main()*** que recibirá un nombre de fichero desde la línea de comandos. El programa debe leer el número mágico del fichero y determinar si su tipo está entre uno de los conocidos (los que hay en ***tipos.dat***), o por el contrario, tiene un contenido desconocido. Cuando se trate de un fichero de tipo conocido deberá informar si la extensión del fichero es o no correcta y, si no lo es, cuál sería la extensión adecuada.

Por ejemplo, ante la orden:

```
java Magico imagen.pdf
```

Y supuesto que localiza el fichero correctamente, debería responder una de estas tres cosas:

- Si no conoce el número mágico:

```
Contenido: desconocido.
```

- Si conoce el número mágico y la extensión es correcta:

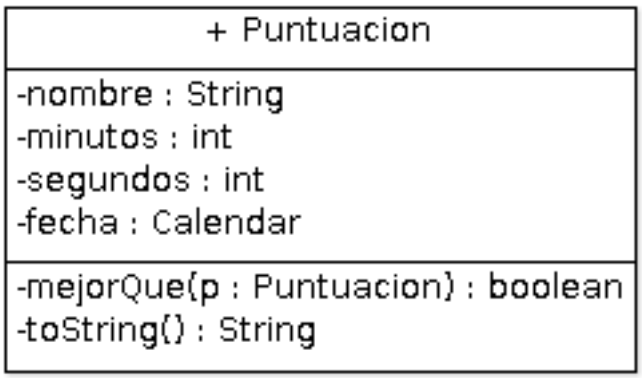
```
Contenido: "formato de documento portátil, de Adobe"
Extensión: correcta.
```

- Si conoce el número mágico y la extensión es errónea:

```
Contenido: "formato de compresión de datos con poca pérdida"
Extensión: errónea, se propone: zip
```

Quinto: Puntuaciones.java (entrega obligatoria)

Como parte del juego del buscaminas, necesitamos guardar en disco las puntuaciones que generan los jugadores. Te han encargado crear una clase ***Puntuacion.java*** con la estructura que ves en el diagrama ***UML*** de la derecha. La clase ***Puntuacion*** tiene que ser ***Serializable*** y la guardaremos en el paquete ***anexo*** (***src/anexo***).



El método ***mejorQue()*** devuelve ***true*** si el tiempo de esta puntuación es menor que el tiempo del que recibe como parámetro; ***false*** en otro caso.

Tendrás que crear también una clase llamada ***Puntuaciones.java*** con un método ***Java*** para almacenar las puntuaciones en un fichero llamado ***src/datos/puntuaciones.dat***, cuya cabecera es la siguiente:

```
public static boolean nuevo (Puntuacion nuevaPuntuacion)
```

Este método hará lo siguiente:

- Si no existe el fichero ***src/datos/puntuaciones.dat***, lo crea con esta nueva puntuación que recibe en la cabecera.
- Si ya existe el fichero, lee todas las puntuaciones y las vuelve a escribir, insertando la nueva puntuación en el lugar que le corresponda por tiempo. A la hora de escribir este método suponemos que las puntuaciones que ya existían en el fichero estaban ordenados, lo cual es cierto si todas las escrituras se hacen desde este método.
- Se mantendrán un máximo de 5 puntuaciones en el fichero que serán los de menor tiempo.
- Cuando la tarea de este método se consigue realizar sin contratiempos, devolverá ***true***. Si no es así, el valor de retorno será ***false***.
- Como siempre, podrás incluir un método ***main()*** que no será valorado, para probar el método.
- Crea otro método ***muestraPuntuaciones()*** para mostrar en pantalla todos las puntuaciones que hay en disco.

```
public static void muestraPuntuaciones()
```

Último:

Una vez terminados los ejercicios, crea una carpeta llamada ***apellido1_apellido2***, copia a esa carpeta todos los ficheros generados (OJO! Solamente los ***.java***) y los ficheros necesarios si los hubiera (imágenes, ...), comprime la carpeta en un ***ZIP*** llamado ***apellido1_apellido2.zip*** y súbelo con el enlace correspondiente de la plataforma ***Moodle***. **Solamente hay que entregar los ejercicios indicados.**

No debes añadir nada más (no crees subcarpetas, no pongas más datos en los nombres de ficheros o carpetas). ***EL INCUMPLIMIENTO DE ESTAS INSTRUCCIONES PUEDE TENER COMO CONSECUENCIA LA NO CORRECCIÓN DE LA PRÁCTICA.***