Ejercicio 21-bis Gestión de alumnos con ObjectInputStream y ObjectOutputStream en **modo añadir datos**

Escribir un programa para escribir y leer datos de alumnos en ficheros binarios, usando las clases `ObjectOutuptStream` y `ObjectInputStream`. De cada alumno se guardan en el fichero los apellidos y el NIA (número de identificación del alumno), que es un número entero.

El programa tendrá cuatro clases:

1) **Alumno:** Implementa la interfaz Serializable. Tiene un constructor al que se le pasan los apellidos y el NIA del alumno, y contiene los métodos getApellidos y getNia para devolver esos valores del objeto.

2) **GestionAlumnos**: Esta es la clase principal que contiene el `main` y presenta el siguiente menú al usuario:
   Seleccione una opción:
   1. Escribir datos de alumnos
   2. Leer datos de alumnos
   3. Salir

3) **EscribirAlumnosFicheroBinario**: Esta clase contiene el método escribirDatos(), que permite al usuario introducir apellidos y NIA de los alumnos, guardándolos en el archivo especificado por el usuario , y que **se le pasa como parámetro al método**. Tras escribir los datos de un alumno se pregunta al usuario si desea añadir otro alumno. Se siguen guardando alumnos mientras el usuario responda que sí. Cada vez que se ejecuta este método se **AÑADE** información si el fichero ya existía.

4) **LeerAlumnosFicheroBinario**: Esta clase tiene el método leerDatos(), que lee los datos de todos los alumnos desde el archivo que especifique el usuario y que **se le pasa como parámetro al método**, y los muestra en pantalla con el siguiente formato:
   Alumno 1: Camino Manzano NIA: 3333333
   Alumno 2: Amores Molina NIA: 11111
   Alumno 3: Moreno García NIA: 22222

Clase Alumno

```
import java.io.Serializable;

class Alumno implements Serializable {
    private static final long serialVersionUID = 2L;

    private String apellidos;
    private int nia;

    public Alumno(String apellidos, int nia) {
        this.apellidos = apellidos;
        this.nia = nia;
    }

    public String getApellidos() {
        return apellidos;
    }

    public int getNia() {
        return nia;
    }
}
```

Clase AppendedObjectOutputStream

```java
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.io.OutputStream;

// Subclase para evitar escribir el encabezado en modo append
class AppendedObjectOutputStream extends ObjectOutputStream {

    public AppendedObjectOutputStream(OutputStream out) throws IOException {
        super(out);
    }

    @Override
    protected void writeStreamHeader() throws IOException {
        // No escribir el encabezado si estamos en modo append
        reset();
    }
}
```

Clase EscribirAlumnosFicheroBinario

```java
class EscribirAlumnosFicheroBinario {

    public void escribirDatos(String rutaFichero) {
        Scanner sc = new Scanner(System.in);
        String apellidos;
        int nia;
        String continuar;

        // Comprobar si el archivo ya existe
        boolean existeArchivo = new File(rutaFichero).exists();

        // Usar ObjectOutputStream o AppendedObjectOutputStream dependiendo de si el archivo ya existe
        try (ObjectOutputStream oos = existeArchivo ?
                new AppendedObjectOutputStream(new FileOutputStream(rutaFichero, true)) :
                new ObjectOutputStream(new FileOutputStream(rutaFichero))) {

            do {
                // Leer apellidos del alumno
                System.out.print("Introduzca los apellidos del alumno: ");
                apellidos = sc.nextLine();

                // Leer el NIA del alumno
                System.out.print("Introduzca el NIA (Número de Identificación del Alumno): ");
                nia = sc.nextInt();
                sc.nextLine();  // Limpia el buffer leyendo el /n que queda después del nextInt()

                // Crear un nuevo alumno y escribirlo en el archivo inmediatamente
                Alumno alumno = new Alumno(apellidos, nia);
                oos.writeObject(alumno);

                // Preguntar si desea continuar
                System.out.print("¿Desea añadir otro alumno? (s/n): ");
                continuar = sc.nextLine();

            } while (continuar.equalsIgnoreCase("s"));

            System.out.println("Datos de los alumnos guardados en el archivo: " + rutaFichero);

        } catch (IOException e) {
            System.out.println("Ocurrió un error al escribir en el fichero: " + e.getMessage());
        }
    }
}
```

Clase LeerAlumnosFicheroBinario

```
class LeerAlumnosFicheroBinario {

    public void leerDatos(String rutaFichero) {
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(rutaFichero))) {

            boolean finDeArchivo = false;
            int contador = 1;

            // Leer y mostrar cada objeto directamente del archivo
            while (!finDeArchivo) {
                try {
                    Alumno alumno = (Alumno) ois.readObject();
                    System.out.printf("Alumno %d: \t%s \tNIA: %d%n", contador++, alumno.getApellidos(), alumno.getNia());
                } catch (EOFException e) {
                    finDeArchivo = true;  // Fin del archivo
                }
            }

        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Ocurrió un error al leer el fichero: " + e.getMessage());
        }
    }
}
```

Clase principal: GestionDeAlumnos

```
public class GestionDeAlumnos {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        //EscribirAlumnosFicheroBinario escribirAlumnos = new EscribirAlumnosFicheroBinario();
        //LeerAlumnosFicheroBinario leerAlumnos = new LeerAlumnosFicheroBinario();

        EscribirAlumnosFicheroBinario escritor = new EscribirAlumnosFicheroBinario();
        LeerAlumnosFicheroBinario lector = new LeerAlumnosFicheroBinario();

        // Menú de opciones
        String opcion;
        do {
            System.out.println("Seleccione una opción:");
            System.out.println("1. Escribir datos de alumnos (si elfichero existe se añaden los nuevos alumnos)");
            System.out.println("2. Leer datos de alumnos");
            System.out.println("3. Salir");
            opcion = sc.nextLine();

            switch (opcion) {
                case "1":
                    System.out.print("Introduzca el nombre del fichero para guardar los datos: ");
                    String ficheroEscritura = sc.nextLine();
                    escritor.escribirDatos(ficheroEscritura);
                    break;
                case "2":
                    System.out.print("Introduzca el nombre del fichero para leer los datos: ");
                    String ficheroLectura = sc.nextLine();
                    lector.leerDatos(ficheroLectura);
                    break;
                case "3":
                    System.out.println("Saliendo del programa...");
                    break;
                default:
                    System.out.println("Opción no válida. Intente de nuevo.");
            }
        } while (!opcion.equals("3"));

        sc.close();

    }
}
```