1. Ejercicios

Nota

- Debes tratar las excepciones que correspondan en cada caso. Puedes consultar las excepciones de cada método del paquete java.io en la documentación oficial.
- En algunos casos deberás consultar el manejo de Strings estudiados en la unidad 5

Ejercicio 1. Escribe un programa que muestre la lista de ficheros y directorios de una ruta del sistema introducida por el usuario. El programa mostrará el contenido del directorio, si existe, y volverá a pedir otra ruta al usuario; terminará cuando el usuario introduzca una ruta vacía.

Ejercicio 2. Amplía el programa anterior para que pida al usuario una extensión de fichero concreta (por ejemplo, ".txt") y muestre todos los ficheros del directorio dado con esa extensión. Crea y utiliza un método llamado ArrayList<String> filtrarPorExtension(File fichero, String extensión) que se encargará de filtrar la ruta según la extensión pasada en el parámetro. Nota: el usuario pasa la extensión sin el punto (.)

Ejercicio 3. Escribe un programa que diga cuántos párrafos tiene un fichero de texto. Además, debe mostrar una lista con el número de párrafo y la cantidad de letras que tiene el párrafo correspondiente.

Ejercicio 4. Escribe un programa que vaya pidiendo frases por teclado (separadas por Intro) y que las vaya escribiendo en un archivo de texto. Se guardará cada frase en una línea. El programa terminará cuando se introduzca la palabra Fin, que no se escribirá en el fichero.

Ejercicio 5. Escribe un programa que simula la agenda de teléfonos. El programa va pidiendo nombre y teléfono y los guarda el un fichero, un par de valores por línea. Por ejemplo, fichero podría quedar así:

```
Anakin-612428333
Leia-683293841
```

Ejercicio 6. Implementa un programa que muestre por pantalla los valores máximos y mínimos del archivo 'numeros.txt'

Ejercicio 7. El archivo 'alumnos_notas.txt' contiene una lista de 10 alumnos y las notas que han obtenido en cada asignatura. El número de asignaturas de cada alumno es variable. Implementa un programa que muestre por pantalla la nota media de cada alumno junto a su nombre y apellido, ordenado por nota media de mayor a menor.

Ejercicio 8. Haz un programa que genere un fichero con algunos números enteros, lo cierre y lo vuelve a abrir y muestre el contenido. A continuación, le añadirá más enteros y que vuelva a mostrar el fichero. Para ello:

- Haz el método *generarFicheroEnteros*, al que se le pase un nombre de fichero y una cantidad, y guardará en dicho fichero tantos números como la cantidad indicada. Los números se generarán de forma aleatoria.

- Haz el método *mostrarFicheroEnteros* al que se le pasa como parámetro un fichero y debe mostrar todos los números del fichero (sabemos que son enteros) por pantalla, separados por un tabulador.

Ejercicio 9. Escribe un programa que permita leer y escribir un fichero binario con información de empleados de una empresa. Cada empleado tendrá los siguientes datos:

- Nombre completo (cadena de caracteres)
- Edad (entero)
- Salario (double)

El programa deberá permitir al usuario elegir entre dos opciones:

- Crear un nuevo fichero de empleados y agregar información de empleados al mismo.
- Leer un fichero existente de empleados y mostrar en pantalla la información de todos los empleados en formato legible.

Cuando se seleccione la opción 1, el programa deberá solicitar al usuario que ingrese la información de un nuevo empleado. La información ingresada deberá ser validada antes de ser escrita en el fichero binario. Si el fichero no existe, deberá crearse. Si ya existe, la información del nuevo empleado deberá ser agregada al final del fichero.

Cuando se seleccione la opción 2, el programa deberá solicitar al usuario que ingrese el nombre del fichero que se desea leer. Si el fichero existe y contiene información de empleados, esta información deberá ser leída y mostrada en pantalla de forma legible.

Deberás tres clases:

- Empleado: clase serializable con los atributos necesarios (nombre, edad y salario), el constructor y los getters/setters.
- FicheroEmpleados: clase que manejará el fichero de los empleados. Como atributo tendrá el nombre de fichero, y dos métodos:
    - public void agregarEmpleado(Empleado empleado): se encargará de añadir un objeto (serializará) *Empleado* al final del fichero binario.
    - public void mostrarEmpleados(): se encargará de leer y mostrar en pantalla todos los objetos *Empleado* (deserializará) almacenados en el fichero binario.
- Ejercicio10Test: contiene el método main. Mostrará el menú y se encargará de pedir el ingreso de datos

Ejemplo de salida:

```
Introduce el nombre del fichero de empleados: empleados.dat

Menú
1. Agregar empleado
2. Mostrar empleados
```

```
3. Salir

Opción: 1

Nombre: Anakin
Edad: 40
Salario: 1.200

Empleado agregado correctamente.

1. Agregar empleado
2. Mostrar empleados
3. Salir

Opción: 2

Nombre               Edad       Salario
Anakin               40         1200.0
```

Ejercicio 10. Queremos tener una agenda telefónica en la que podamos guardar el nombre y teléfono de un contacto. Realiza un programa con un bucle y un menú con las siguientes opciones. Evidentemente debes crear una clase Contacto:

1. Dar de alta un contacto
2. Consultar un contacto por su nombre
3. Saber la cantidad de amigos grabados
4. Mostrar toda la agenda por pantalla
5. Borrar un contacto
6. Modificar los datos de un contacto
7. Importar datos
8. Exportar datos

Las opciones más importantes de este ejercicio, en cuanto al uso de ficheros, son las de importar y exportar datos. La opción de importar datos solicitará un nombre de fichero donde están los contactos y lo pasará a *ArrayList*. Y la opción de exportar datos pedirá un nombre de fichero guardará allí los datos de *ArrayList*. El resto de las opciones será a través de la gestión del *ArrayList*.

Nota: en este ejercicio debes considerar la clase Contacto como serializable y utilizar la clase ObjectInputStream y ObjectOutputStream.