IES CHAN DO MONTE
C.S. de Desarrollo de Aplicaciones Multiplataforma
Modulo Acceso a datos

ACTIVIDAD 2: FICHEROS SECUENCIALES DE TEXTO

Objetivo:

- Utilizar las clases en Java para manejar flujos de caracteres: Clase FileReader y FileWriter
- Optimizar las operaciones de entrada/salida en ficheros binarios utilizando un buffer.
- Utilizar las clases BufferedReader y BufferedWriter.
- Utilizar las clases para convertir flujo de bytes en flujo de caracteres
- Uso de NIO para ficheros de texto
- Manejar las excepciones y crear excepciones propias.

La resolución de estos ejercicios consistirá en **trabajar con ficheros de texto en Java** aplicando el principio de **responsabilidad única**.

La idea no es solo resolver la tarea (contar líneas, crear directorios, buscar palabras), sino hacerlo de forma **estructurada y reutilizable**, evitando repetir código.

Requisitos de diseño

1. **Separación de responsabilidades**
   - Toda la lógica de **abrir, leer y escribir ficheros** debe estar encapsulada en clases especializadas.
   - La clase Operaciones no debe preocuparse de cómo se abre, se lee, escribe , o se cierra un fichero, solo de la lógica del ejercicio.
2. **Clases a implementar**
   - Archivo **(abstracta)**
     - Define lo común a cualquier fichero: ruta, comprobar existencia, borrar, renombrar. Tendrá dos métodos abstractos abrirArchivo () y cerrarArchivo que tendrán que implementar las clases hijas
   - lecturaTexto
     - Encapsula cómo abrir un fichero de texto en modo lectura y como cerrarlo.
     - Permite leer línea a línea (leerLinea()) devolviendo un string
   - escribirTexto
     - Encapsula cómo abrir un fichero de texto en modo escritura y como cerrarlo.
     - Permite escribir una línea (escribirLinea(String)).
   - Operaciones
     - Contendrá los métodos que resuelven los ejercicios (contar líneas, crear directorios, contar palabra).
     - Delegará siempre en lecturaTexto y escribirTexto para la gestión de ficheros.
   - Main
     - Punto de entrada del programa.
     - Solo contendrá llamadas representativas a los métodos de Operaciones.

Ejercicio 1:

Escribe un método que cuente el número de líneas de cada fichero que se especifique en la línea de comandos (Nota: pueden especificarse varios archivos, como por ejemplo: "exercicio5-1 file1.txt file3.txt file2.txt"). Los archivos deben ser archivos de texto con la extensión txt.

Escribe en un fichero de texto llamado Salida.txt: el nombre de cada fichero, junto con el número de líneas del fichero. Si ocurre un error al intentar leer uno de los ficheros, en el fichero salida.txt se graba un mensaje de error para el archivo, y se deben procesar todos los ficheros restantes.

Ejercicio 2:

A partir de un fichero de texto con el formato CURSO/NUMERO/ALUMNO crear un directorio por cada curso y dentro de este un directorio por cada alumno perteneciente a ese curso. En un fichero de texto llamado ficherolog.txt se irá escribiendo el éxito o fracaso en la creación de cada directorio de alumnos

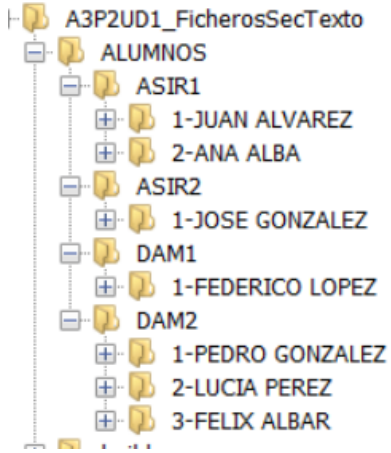
Ejemplo de fichero de log

```
.-PEDRO GONZALEZ  -----> se creo correctamente el directorio
!-LUCIA PEREZ  -----> se creo correctamente el directorio
.-JUAN ALVAREZ  -----> se creo correctamente el directorio
.-JOSE GONZALEZ  -----> se creo correctamente el directorio
!-ANA ALBA  -----> se creo correctamente el directorio
.-FEDERICO LOPEZ  -----> se creo correctamente el directorio
)AM2/JOSE ESTEVEZ----> no tiene el formato CURSO/NUMERO/ALUMNO. NO SE PUEDE CREAR EL DIRECTORI(
)-FELIX ALBAR  -----> se creo correctamente el directorio
```

Ejemplo de fichero de entrada con los datos de los alumnos:

```
run:
DAM2/1/PEDRO GONZALEZ
DAM2/2/LUCIA PEREZ
ASIR1/1/JUAN ALVAREZ
ASIR2/1/JOSE GONZALEZ
ASIR1/2/ANA ALBA
DAM1/1/FEDERICO LOPEZ
DAM2/JOSE ESTEVEZ
DAM2/3/FELIX ALBAR
```

Ejemplo de los directorios:



Ejercicio 3:

Escribe un método en Java que, dado un fichero de texto y una palabra, cuente cuántas veces aparece esa palabra en cada línea del fichero y también el número total de apariciones en todo el documento. La salida se grabará en un fichero de texto.

Recibir como parámetros:

- El nombre del fichero de entrada (texto plano con extensión .txt).
- La palabra a buscar.
- El nombre del fichero de salida donde se guardará el resultado.

Gestionar excepciones:

- Si el fichero de entrada no existe o no se puede leer, mostrar un mensaje de error.
- Si ocurre un error al escribir el fichero de salida, también debe informarse.

Ejemplo fichero de texto de entrada

```
La escoba está en la esquina
No hay escoba aquí
Una escoba, otra escoba, y más escoba
```

Formato del fichero de salida

```
La palabra "escoba" en el fichero entrada.txt:
------------------------------
linea 1: aparece 1 veces
linea 2: aparece 1 veces
linea 3: aparece 3 veces
------------------------------
Aparece un total de 5 veces
```