David Torres Aranha

Actividad 1.1:
Configuración del entorno y tratamiento de ficheros

Clases de `java.io`

- **File:** representa un fichero o directorio y permite comprobar si existe, obtener su ruta, crear o eliminarlo, etc. No lee ni escribe contenido.
- **FileReader:** permite leer caracteres de un fichero de texto.
- **BufferedReader:** permite leer texto usando un búfer, haciendo la lectura más eficiente. Incluye **`readLine()`**.
- **FileWriter:** permite escribir caracteres en un fichero.
- **BufferedWriter:** permite escribir usando un búfer, mejorando el rendimiento. Incluye **`newLine()`**.
- **PrintWriter:** facilita la escritura de texto mediante **`print()`**, **`println()`** y **`printf().`**

Diferencia entre FileReader y BufferedReader

`FileReader` lee directamente los caracteres del fichero.

`BufferedReader` utiliza un búfer, por lo que es más eficiente para muchas lecturas y permite leer líneas completas con **`readLine().`**

BufferedReader br = new BufferedReader(new FileReader("fichero.txt"));

Diferencia entre FileWriter, BufferedWriter y PrintWriter

- **FileWriter:** escribe directamente en el fichero.
- **BufferedWriter:** utiliza un búfer para hacer la escritura más eficiente.
- **PrintWriter:** facilita la escritura de texto con `print()`, `println()` y `printf()`.

FileWriter vs PrintWriter y gestión de errores

FileWriter puede lanzar **`IOException`**, por lo que hay que gestionarla.

PrintWriter no lanza normalmente **`IOException`** durante la escritura; los errores se pueden comprobar con **`checkError()`.**

Para añadir texto al fina**l** de un fichero se utiliza **`FileWriter`** con **`true`**:

```
FileWriter fw = new FileWriter("fichero.txt", true);
```

El **`true`** indica que el contenido se añadirá al final en lugar de sobrescribir el fichero.

También se puede combinar:

```
BufferedWriter bw = new BufferedWriter(
    new FileWriter("fichero.txt", true)
);
```

Así se consigue añadir texto y utilizar búfer para mejorar la escritura.