Investigación sobre clases del paquete java.io

1. ¿Para qué se utilizan las siguientes clases?

- **File**: Representa una ruta de acceso abstracta a un archivo o carpeta en el sistema de archivos. Sirve para comprobar propiedades del archivo (saber si existe, consultar su tamaño, permisos, etc.) o para realizar acciones como crear, eliminar o renombrar directorios/archivos. No lee ni escribe el contenido del archivo directamente.
- **FileReader**: Flujo de lectura de caracteres básico. Se utiliza para leer archivos de texto directamente desde el disco carácter por carácter.
- **BufferedReader**: Lee texto de un flujo de entrada almacenando previamente los datos en un búfer (memoria intermedia). Permite leer eficientemente datos y proporciona el método readLine() para leer líneas completas de texto de una sola vez.
- **FileWriter**: Flujo de escritura de caracteres básico. Sirve para escribir caracteres o cadenas de texto directamente en un archivo.
- **BufferedWriter**: Guarda el texto en un búfer en memoria antes de escribirlo físicamente en el archivo. Optimiza las operaciones de escritura reduciendo los accesos continuos al disco.
- **PrintWriter**: Clase avanzada de escritura que permite imprimir representaciones formateadas de objetos y texto. Ofrece métodos cómodos como println(), print() y printf().

2. ¿Qué diferencia hay entre FileReader y BufferedReader? ¿Cuál es más eficiente en el tratamiento de información? ¿Por qué?

- **Diferencia principal:** FileReader lee los datos directamente del disco carácter por carácter. En cambio, BufferedReader lee bloques grandes de texto del disco y los almacena en la memoria RAM (búfer), permitiendo además la lectura cómoda de líneas completas mediante readLine().

- **Clase más eficiente: BufferedReader**.
- **¿Por qué?:** Reduzca drásticamente el número de accesos físicos al disco duro (las operaciones de Entrada/Salida son muy lentas). En lugar de pedir al disco cada letra individualmente, lee un conjunto entero de datos a la memoria RAM de una sola vez.

3. ¿Qué diferencia hay entre FileWriter, BufferedWriter y PrintWriter?

- **FileWriter**: Escribe caracteres directamente en el archivo de texto de bajo nivel sin utilizar un búfer.
- **BufferedWriter**: Añade una memoria intermedia sobre un flujo de escritura (como FileWriter), agrupando los datos en memoria para escribirlos en el disco en un solo viaje.
- **PrintWriter**: Añade funcionalidades superiores de formato (métodos println(), printf()) y gestiona internamente las excepciones sin interrumpir la ejecución con errores obligatorios.

4. Preguntas sobre FileWriter y PrintWriter:

- **¿Cuál de las dos clases lanza excepciones de tipo IOException?**
  FileWriter obliga a capturar o declarar excepciones comprobadas de tipo IOException.
- **¿Cómo gestionará los errores la otra clase?**
  PrintWriter no lanza explícitamente IOException. Captura las excepciones de forma interna de modo que no bloqueen el código, y permite consultar si ha ocurrido algún error llamando al método **checkError()**, el cual devuelve true si ocurrió un fallo.

- **¿Cuál de las dos clases usarías para añadir/escribir texto al final de un fichero?**
  Se utiliza **FileWriter** mediante su constructor que acepta un parámetro booleano: new FileWriter(archivo, true). Para usar PrintWriter manteniendo este comportamiento, se envuelve dicha instancia:

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