Ejercicio 1.

Abre el proyecto ADo1_02_03 ListaNumeros UT8 AL y completa la clase ListaNumeros. Esta clase guarda en una colección List una serie de números leídos desde un fichero. Completa los siguientes métodos:

- public int leerDeFicheroV1(String nombre) lee del fichero de texto cuyo nombre se especifica como parámetro una serie de números y los añade a la lista en memoria. El fichero de texto contiene una serie de líneas con formato: n1:n2:n3:...., es decir, en cada línea hay una serie de números separados por el carácter:
 - El fichero se lee línea a línea (con BufferedReader) y con ayuda del método parserLinea() se extrae cada no v se añade a la lista.
 - Hay que capturar aquí todas las posibles excepciones.
 - El método devuelve la cantidad total de valores erróneos leídos entre todas las líneas.
- public int parsearLinea(String linea) A partir de una línea de texto extrae los números y los añade a la lista. Las excepciones de formato se capturarán contabilizando los nos erróneos que se detectan. El programa no para cuando se encuentra un n.º erróneo.

Ejercicio 2.

Siguiendo con el proyecto anterior:

- public int leerDeFicheroV2(String nombre) Haz lo mismo que en leerDeFicheroV1() pero ahora usando la clase Scanner
- public int maximo() Calcula y devuelve el valor máximo de la colección (usa la clase Collections)
- public List<Integer> ordenarDescendente() Devuelve una colección ordenada de mayor a menor sin modificar la colección original

Ejercicio 3.

Añade ahora a la clase ListaNumeros el método public void salvarEnFichero(String nombre) que guarda en el fichero cuyo nombre indica el parámetro el valor máximo de la colección en una línea y la colección ordenada descendentemente en otra línea. Se propagarán todas las posibles excepciones.

Completa el método main():

- crea un objeto ListaNumeros Х
- lee del fichero "numeros.txt" los nos y guárdalos en la lista (usa leerDeFicheroV1()) Х
- muestra la cantidad de valores erróneos leídos Х
- guarda en el fichero "resultado1.txt" el máximo y la lista en orden descendente Х
- vacia la lista Х
- lee del fichero "numeros.txt" los nºs y guárdalos en la lista (usa leerDeFicheroV2()) Х
- guarda en el fichero "resultado2.txt" el máximo y la lista en orden descendente

Propaga todas las posibles excepciones.

Con el fichero *numeros.txt* que se proporciona el resultado será:

```
Nos erróneos leídos 2
```

```
resultado1.txt

Máximo de la lista: 125

[125, 89, 89, 89, 77, 23, 21, 19, 12, 7, 4, 4, 3, 3, -32, -67]

resultado2.txt

Máximo de la lista: 125

[125, 89, 89, 89, 77, 23, 21, 19, 12, 7, 4, 4, 3, 3, -32, -67]
```

Ejercicio 4.

Crea una clase ComandoType que simule el funcionamiento del comando *type* del DOS (muestra el contenido de un fichero de texto) pero numerando además las líneas a partir de 1.

```
1: línea
2: línea
3: línea
```

La clase incluirá como único atributo el nombre del fichero a mostrar que se recibirá como parámetro del constructor. El método public void mostrarFichero() throws IOException mostrará el contenido del fichero.

Añade una clase con el método main() y elige con JFileChooser un fichero de texto (puedes seleccionar uno que contenga un programa fuente *java*) y muestra su contenido. Trata aquí las posibles excepciones.

Utilizando JFileChooser y FileNameExtensionFilter

- JFileChooser es una clase del paquete javax.swing.JFileChooser (consulta la ayuda)
- Abre un cuadro de diálogo modal (ventana que pertenece a la ventana padre, ésta queda deshabilitada hasta cerrar la ventana de diálogo) permitiendo que un usuario de una aplicación pueda seleccionar un fichero a procesar.
- Un objeto JFileChooser se abre llamando a showOpenDialog() o showSaveDialog()
- El valor APPROVE_OPTION se devuelve si el botón *Open / Save* se ha pulsado
- El método getSelectedFile() devuelve un objeto File asociado al fichero seleccionado
- Por defecto JFileChooser se abre en el directorio home del usuario. Se puede elegir el directorio en el que queremos que se abra llamando al método setCurrentDirectory() pasando el directorio en cuestión.
 - → fc.setCurrentDirectory(new File("."));

 ó
 - → elector.setCurrentDirectory(new File(System.getProperty("user.dir")));
- La clase FileNameExtensionFilter permite establecer un filtro para una serie de extensiones

```
FileFilter filtro = new FileNameExtensionFilter("JPEG ficheros", "jpg", "jpeg");
JFileChooser elector = ...;
elector.setFileFilter(filtro);
```

