

CIFP César Manrique.

Programación 1º de Desarrollo de Aplicaciones Web

Profesor: José David Díaz Díaz

Actividades de la Unidad 10: Ficheros binarios



Esta obra está licenciada bajo la Licencia Creative Commons Atribución 4.0 Internacional.
Para ver una copia de esta licencia, visite <http://creativecommons.org/licenses/by/4.0/> o
envíe una carta a Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Índice

Actividades.....	1
Desarrollo.....	2
Actividades de comprobación.....	2
11.1. Los ficheros binarios se diferencian de los de texto en que:.....	2
11.2. Si queremos guardar una cadena de caracteres en un flujo binario de tipo ObjectOutputStream, usaremos:.....	2
11.3. Para guardar una tabla del tipo int[] en ObjectOutputStream, usaremos:.....	2
11.4. Si queremos leer una tabla de Cadenas de caracteres del flujo binario entrada de tipo ObjectInputStream, escribiremos:.....	3
11.5. Un flujo de tipo ObjectInputStream permite leer de:.....	3
11.6. Un flujo de tipo ObjectInputStream permite acceder a:.....	3
11.7. Si guardamos una cadena de caracteres usando un flujo objectOutputStream, podemos leerla directamente del archivo:.....	4
11.8. Si guardamos una serie de objetos de la clase Cliente con un flujo ObjectOutputStream, los recuperaremos:.....	4
11.9. Los flujos binarios se cierran:.....	4
11.10. Hay que cerrar los flujos binarios:.....	5
Actividades de Aplicación.....	6
11.11. Pide un valor double por consola y guardalo en un archivo binario.....	6
11.12. Abre el fichero de la Actividad de aplicación 11.11, lee el valor double contenido en él y muéstralo por pantalla.....	8
11.13. Escribe un programa que lea de un fichero binario una tabla de números double y después muestre el contenido de la tabla por consola.....	9

11.14. Introduce por teclado una frase y guardala en un archivo binario. A continuación, recuperala y muéstrala por pantalla.....	11
11.15. Implementa un programa que lea números enteros desde el fichero numeros.dat y los vaya guardando en los ficheros pares.dat e impares.dat, según su paridad.....	13
11.16. Implementa una aplicación que gestione una lista de nombres ordenada por orden alfabético. Al arrancar se leerá de un fichero los nombres insertados anteriormente y se pedirán nombres nuevos hasta que se introduzca la cadena « fin ». Cada nombre que se introduzca deberá añadirse a los que ya había, de forma que la lista permanezca ordenada. Al terminar, se guardará en el fichero la lista actualizada.....	14
11.17. Escribe un texto, línea a línea, de forma que, cada vez que se pulse Intro, se guarde la línea en un archivo binario. El proceso se termina cuando introduzcamos una línea va cía. Después el programa lee el texto completo del archivo y lo muestra por pantalla	15
11.18. Un libro de firmas es útil para recoger los nombres de todas las personas que han pasado por un determinado lugar. Crea una aplicación que permita mostrar el libro de firmas o insertar un nuevo nombre (comprobando que no se encuentre repetido) usando el fichero binario firmas.dat.....	18
11.19. Por motivos puramente estadísticos se desea llevar constancia del número de llamadas recibidas cada día en una oficina. Para ello, al terminar cada jornada laboral se guarda dicho número al final de un archivo binario. Implementa una aplicación con un menú, que nos permita añadir el número correspondiente cada día y ver la lista completa en cualquier momento	19

Actividades

Actividades de la Unidad 10: Ficheros binarios.

En este documento se detallan las actividades a realizar. Se entregará al profesor en la plataforma digital dos ficheros. Un primer fichero pdf con todas las actividades a realizar, el nombre del fichero será “unidad2 + nombre del alumno.pdf”. Añadir en el fichero pdf por cada actividad de programación dos capturas de pantalla, una del código y otra de su ejecución. También en el fichero pdf copiar todas las preguntas y las respuestas correctas de las actividades de comprobación. Además, entregar un segundo fichero comprimido con todos los códigos fuentes de cada actividad de programación realizada.

Todas las actividades resueltas se deberán de analizar y no se entregarán.

A continuación, detallamos las actividades a realizar:

- **Actividades propuestas.** No realizar ninguna.
- **Actividades de comprobación.** Realizarlas todas. Copiar todas las preguntas y sus respuestas correctas.
- **Actividades de aplicación.** Realizar las siguientes **11.11, 11.12, 11.13, 11.14, 11.15, 11.16, 11.17, 11.18 y 11.19.**
- **Actividades de ampliación.** No realizar ninguna.

[Volver al índice](#)

Desarrollo

Actividades de comprobación.

11.1. Los ficheros binarios se diferencian de los de texto en que:

- a) Solo tienen ceros y unos.
- b) Sirven tanto para escribir como para leer.
- c) No sirven para guardar texto.
- d) Permiten guardar todo tipo de datos, incluidos datos primitivos y objetos.

[Volver al índice](#)

11.2. Si queremos guardar una cadena de caracteres en un flujo binario de tipo `ObjectOutputStream`, usaremos:

- a) `writeString()`.
- b) `writeChar()`.
- c) `writeObject()`.
- d) Nada, no se puede.

[Volver al índice](#)

11.3. Para guardar una tabla del tipo `int[]` en `ObjectOutputStream`, usaremos:

- a) `writeln()`.
- b) `writeArrayInt()`.
- c) `readObject()`.
- d) `writeObject()`.

[Volver al índice](#)

11.4. Si queremos leer una tabla de Cadenas de caracteres del flujo binario entrada de tipo `ObjectInputStream`, escribiremos:

- a) `String[] tabla = (String []) entrada.readObject();`
- b) `String tabla = (String) entrada.readObject();`
- c) `String[] tabla = entrada.readObject();`
- d) `String[] tabla = (Object) .readObject();`

[Volver al índice](#)

11.5. Un flujo de tipo `ObjectInputStream` permite leer de:

- a) Cualquier archivo de Windows.
- b) Archivos de imagen con extensión JPG.
- c) Archivos creados con un flujo `ObjectOutputStream`.
- d) Archivos creados con un flujo `BufferedReader`.

[Volver al índice](#)

11.6. Un flujo de tipo `ObjectInputStream` permite acceder a:

- a) Solo archivos del disco duro.
- b) Cualquier fuente de datos primitivos u objetos de Java.
- c) Únicamente a conexiones de red.
- d) Solo nos permite leer de la consola.

[Volver al índice](#)

11.7. Si guardamos una cadena de caracteres usando un flujo `objectOutputStream`, podemos leerla directamente del archivo:

- a) Usando un procesador de texto.
- b) Usando un editor de texto.
- c) Usando una hoja de cálculo.
- d) Usando un flujo `ObjectInputStream` .

[Volver al índice](#)

11.8. Si guardamos una serie de objetos de la clase `Cliente` con un flujo `ObjectOutputStream`, los recuperaremos:

- a) En el mismo orden en que se guardaron.
- b) En orden inverso.
- c) En un orden aleatorio.
- d) Nunca se pueden recuperar.

[Volver al índice](#)

11.9. Los flujos binarios se cierran:

- a) Con el método `close()`.
- b) Apagando el ordenador.
- c) Abortando el programa.
- d) Con el método `cerrar()`.

[Volver al índice](#)

11.10. Hay que cerrar los flujos binarios:

- a) Siempre.
- b) Una vez al día.
- c) Solo si no se han abierto con una estructura try-catch con recursos.
- d) Nunca .

[Volver al índice](#)

Actividades de Aplicación.

11.11. Pide un valor double por consola y guardalo en un archivo binario.

```
1  package Main;
2
3  import java.io.*;
4  import java.util.*;
5
6  /**
7   * @author juancfm
8   */
9  public class Main {
10
11     final static String FILENAME = "valor.dat";
12
13     public static void main(String[] args) {
14         /**
15          * Pide un valor double por consola y guardalo en un archivo binario.
16          */
17
18         Scanner sc = new Scanner(System.in);
19         double valor;
20
21         System.out.println("Ingrese un valor double: ");
22         try {
23             valor = sc.nextDouble();
24             escribir(valor);
25         } catch (InputMismatchException e) {
26             System.out.println("Necesitamos un número. ");
27         }
28     }
29
30     /**
31     * escribe el fichero
32     * @param valor El valor que se va a escribir
33     */
34     public static void escribir (double valor){
35
36         try (ObjectOutputStream myFile
37             = new ObjectOutputStream(new FileOutputStream(name: FILENAME))) {
38
39             myFile.writeDouble(valor);
40
41             System.out.println("El valor se ha guardado correctamente en "
42                 + "el archivo binario.");
43
44         } catch (IOException e) {
45             System.out.println("Ha ocurrido un error al guardar el valor "
46                 + "en el archivo: " + e.getMessage());
47         } catch (Exception e) {
48             System.out.println("Ha ocurrido un error: " + e.getMessage());
49         }
50     }
51 }
52
```

Actividades de la Unidad 10: Ficheros binarios

```
Output - Main (run) x
run:
Ingrese un valor double:
23,8
El valor se ha guardado correctamente en el archivo binario.
BUILD SUCCESSFUL (total time: 27 seconds)
```

```
Output - Main (run) x
run:
Ingrese un valor double:
12.5
Necesitamos un número.
BUILD SUCCESSFUL (total time: 11 seconds)
```

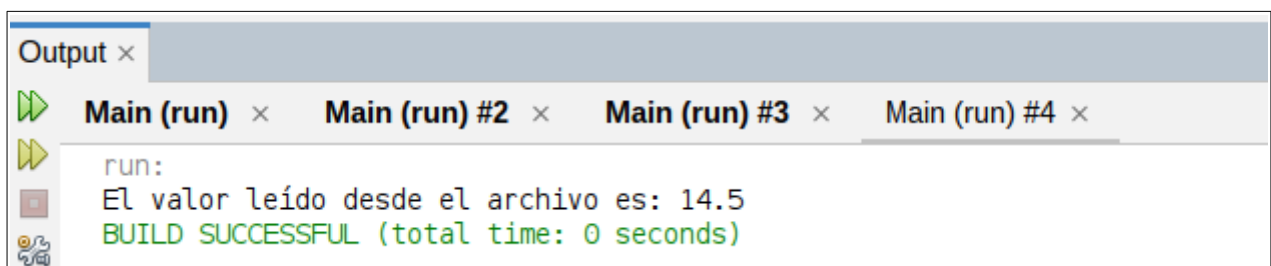
[Volver al índice](#)

11.12. Abre el fichero de la Actividad de aplicación 11.11, lee el valor double contenido en él y muéstralo por pantalla.

```

1  package Main;
2
3  import java.io.*;
4
5  /**
6   * @author juancfm
7   */
8  public class Main {
9
10     final static String FILENAME = "valor.dat";
11
12     public static void main(String[] args) {
13         /**
14          * Abre el fichero de la Actividad de aplicación 11.11, lee el valor
15          * double contenido en él y muéstralo por pantalla.
16          */
17
18         try (ObjectInputStream myFile
19              = new ObjectInputStream(new FileInputStream(name: FILENAME))) {
20
21             double valor = myFile.readDouble();
22             System.out.println("El valor leído desde el archivo es: " + valor);
23
24         } catch (IOException e) {
25             System.out.println("Ha ocurrido un error al leer el archivo: "
26                               + e.getMessage());
27         } catch (Exception e) {
28             System.out.println("Ha ocurrido un error: " + e.getMessage());
29         }
30     }
31 }
32

```



Output ×

Main (run) × Main (run) #2 × Main (run) #3 × Main (run) #4 ×

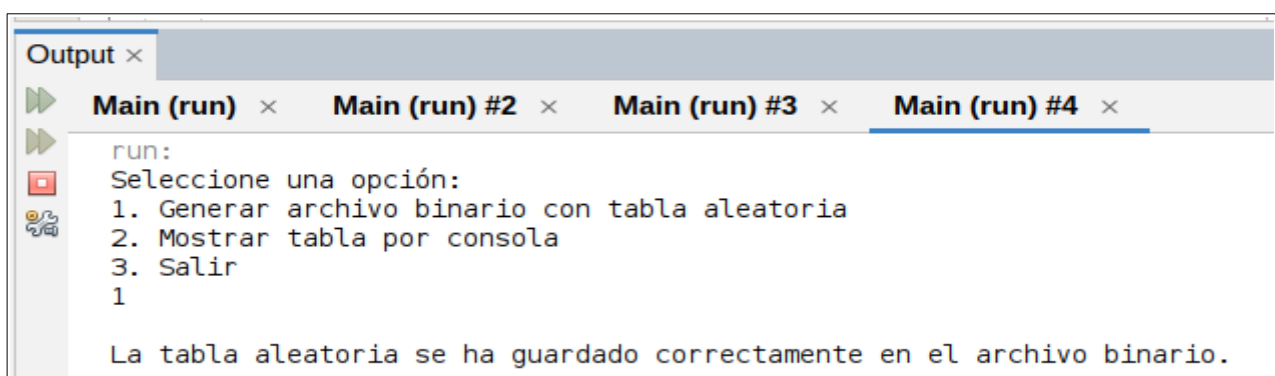
run:
El valor leído desde el archivo es: 14.5
BUILD SUCCESSFUL (total time: 0 seconds)

[Volver al índice](#)

11.13. Escribe un programa que lea de un fichero binario una tabla de números double y después muestre el contenido de la tabla por consola.

```
1 package Main;
2
3 import java.io.*;
4 import java.util.*;
5
6 /**
7  * @author juancfm
8  */
9 public class Main {
10
11     final static String FILENAME = "frase.dat";
12     final static int TABLA_SIZE = 10;
13
14     public static void main(String[] args) {
15         /**
16          * 11.13. Escribe un programa que lea de un fichero binario una tabla de
17          * números double y después muestre el contenido de la tabla por
18          * consola.
19          */
20
21         Scanner sc = new Scanner(System.in);
22         int opcion = 0;
23
24         while (opcion != 3) {
25             System.out.println("Seleccione una opción:");
26             System.out.println("1. Generar archivo binario con tabla "
27                 + "aleatoria");
28             System.out.println("2. Mostrar tabla por consola");
29             System.out.println("3. Salir");
30
31             try {
32                 opcion = sc.nextInt();
33             } catch (InputMismatchException e) {
34                 System.out.println("Seleccione una opción válida.");
35                 sc.nextLine();
36                 continue;
37             }
38
39             switch (opcion) {
40                 case 1 -> generarTablaAleatoria();
41                 case 2 -> mostrarTablaPorConsola();
42                 case 3 -> System.out.println("\nHasta luego.");
43                 default -> System.out.println("\nSeleccione una opción...
44                     + "válida.");
45             }
46         }
47     }
48 }
49
50
```

```
51
52 /**
53  * Genera una tabla de 10 números double aleatorios y la guarda en
54  * un archivo binario FILENAME.
55  */
56 public static void generarTablaAleatoria() {
57     try (ObjectOutputStream myFile
58         = new ObjectOutputStream(new FileOutputStream(FILENAME))) {
59
60         double[] tabla = new double[TABLA_SIZE];
61         Random rand = new Random();
62
63         for (int i = 0; i < TABLA_SIZE; i++) {
64             tabla[i] = rand.nextDouble() * 100;
65         }
66
67         myFile.writeObject(tabla);
68
69         System.out.println("\nLa tabla aleatoria se ha guardado...
70             + "correctamente en el archivo binario.\n");
71     } catch (IOException e) {
72         System.out.println("\nHa ocurrido un error al guardar la tabla...
73             + "en el archivo: " + e.getMessage() + "\n");
74     } catch (Exception e) {
75         System.out.println("\nHa ocurrido un error: "
76             + e.getMessage() + "\n");
77     }
78 }
79
80 /**
81  * Lee el archivo binario FILENAME, muestra la tabla por consola y
82  * cierra el archivo.
83  */
84 public static void mostrarTablaPorConsola() {
85     try (ObjectInputStream myFile
86         = new ObjectInputStream(new FileInputStream(FILENAME))) {
87
88         double[] tabla = (double[]) myFile.readObject();
89         System.out.println("\nLa tabla es la siguiente: \n");
90
91         for (double valor : tabla) {
92             System.out.println(valor);
93         }
94         System.out.println("\n");
95     } catch (FileNotFoundException e) {
96         System.out.println("\nNo se ha encontrado el archivo: "
97             + e.getMessage() + "\n");
98     } catch (IOException e) {
99         System.out.println("\nHa ocurrido un error al leer el archivo: "
100             + e.getMessage() + "\n");
101     } catch (ClassNotFoundException e) {
102         System.out.println("\nNo se ha encontrado la clase necesaria...
103             + "para leer el archivo: " + e.getMessage() + "\n");
104     } catch (Exception e) {
105         System.out.println("\nHa ocurrido un error: " + e.getMessage()
106             + "\n");
107     }
108 }
109
110
111
```



```
Output x
Main (run) x Main (run) #2 x Main (run) #3 x Main (run) #4 x
run:
Seleccione una opción:
1. Generar archivo binario con tabla aleatoria
2. Mostrar tabla por consola
3. Salir
1

La tabla aleatoria se ha guardado correctamente en el archivo binario.
```

Actividades de la Unidad 10: Ficheros binarios

4/4

Seleccione una opción:

1. Generar archivo binario con tabla aleatoria
2. Mostrar tabla por consola
3. Salir

2

La tabla es la siguiente:

49.29045257064809
99.42303599185394
99.96306647443768
48.41786534949782
93.54586283676117
60.45408217992353
31.442346255950483
39.26991569212072
19.45311470785498
48.68236689461172

Seleccione una opción:

1. Generar archivo binario con tabla aleatoria
2. Mostrar tabla por consola
3. Salir

3

Hasta luego.

BUILD SUCCESSFUL (total time: 2 minutes 31 seconds)

[Volver al índice](#)

11.14. Introduce por teclado una frase y guárdala en un archivo binario. A continuación, recupérala y muéstrala por pantalla.

```

1 package Main;
2
3 import java.io.*;
4 import java.util.Scanner;
5
6 /**
7  * @author juancfr
8  */
9 public class Main {
10
11     /**
12      * 11.14. Introduce por teclado una frase y guárdala en un archivo binario.
13      * A continuación, recupérala y muéstrala por pantalla.
14      */
15
16     final static String FILENAME = "frase.dat";
17
18     public static void main(String[] args) {
19
20         Scanner sc = new Scanner(System.in);
21         int option = 0;
22
23         while (option != 3) {
24             System.out.println("\n--- Menú ---");
25             System.out.println("\nSeleccione una opción: ");
26             System.out.println("1. Escribir y guardar frase en archivo");
27             System.out.println("2. Leer frase y mostrar por pantalla");
28             System.out.println("3. Salir");
29
30             option = sc.nextInt();
31             sc.nextLine();
32
33             switch (option) {
34                 case 1:
35                     System.out.println("Ingrese una frase:");
36                     String frase = sc.nextLine();
37                     escribirFraseEnArchivo(frase);
38                     break;
39                 case 2:
40                     String fraseRecuperada = leerFraseDeArchivo();
41                     System.out.println("La frase recuperada del archivo es: "
42                                     + fraseRecuperada);
43                     System.out.println("#####");
44                     break;
45                 case 3:
46                     System.out.println("¡Hasta luego!");
47                     break;
48                 default:
49                     System.out.println("Opción inválida, por favor "
50                                     + "seleccione una opción del menú.");
51                     break;
52             }
53         }
54     }
55 }

```

```

56
57 /**
58  * Escribe una frase en un archivo binario.
59  *
60  * @param frase La frase a escribir en el archivo.
61  */
62 public static void escribirFraseEnArchivo(String frase) {
63     try (ObjectOutputStream out = new ObjectOutputStream(
64         new FileOutputStream(FILENAME))) {
65         out.writeObject(frase);
66         System.out.println("Frase guardada en el archivo correctamente.");
67     } catch (IOException e) {
68         System.out.println("Ha ocurrido un error al guardar la frase en "
69                             + "el archivo: " + e.getMessage());
70     }
71 }
72
73 /**
74  * Lee una frase de un archivo binario y la devuelve como una cadena.
75  *
76  * @return La frase leída del archivo.
77  */
78 public static String leerFraseDeArchivo() {
79     String fraseRecuperada = null;
80     try (ObjectInputStream in = new ObjectInputStream(
81         new FileInputStream(FILENAME))) {
82         fraseRecuperada = (String) in.readObject();
83     } catch (FileNotFoundException e) {
84         System.out.println("El archivo no existe: " + e.getMessage());
85     } catch (IOException e) {
86         System.out.println("Ha ocurrido un error al leer el archivo: "
87                             + e.getMessage());
88     } catch (ClassNotFoundException e) {
89         System.out.println("La clase requerida no se encuentra "
90                             + "en el classpath: " + e.getMessage());
91     }
92     return fraseRecuperada;
93 }
94 }

```

```

Seleccione una opción:
1. Escribir y guardar frase en archivo
2. Leer frase y mostrar por pantalla
3. Salir
1
Ingrese una frase:
Hello World!!!!
Frase guardada en el archivo correctamente.

```

Actividades de la Unidad 10: Ficheros binarios



---- Menú ----

Seleccione una opción:

1. Escribir y guardar frase en archivo
2. Leer frase y mostrar por pantalla
3. Salir

2

#####

La frase recuperada del archivo es: Hello World!!!!

#####

---- Menú ----

Seleccione una opción:

1. Escribir y guardar frase en archivo
2. Leer frase y mostrar por pantalla
3. Salir

3

¡Hasta luego!

BUILD SUCCESSFUL (total time: 1 minute 59 seconds)

[Volver al índice](#)

11.15. Implementa un programa que lea números enteros desde el fichero numeros.dat y los vaya guardando en los ficheros pares.dat e impares.dat, según su paridad.

[Volver al índice](#)

11.16. Implementa una aplicación que gestione una lista de nombres ordenada por orden alfabético. Al arrancar se leerá de un fichero los nombres insertados anteriormente y se pedirán nombres nuevos hasta que se introduzca la cadena « fin ». Cada nombre que se introduzca deberá añadirse a los que ya había, de forma que la lista permanezca ordenada. Al terminar, se guardará en el fichero la lista actualizada.

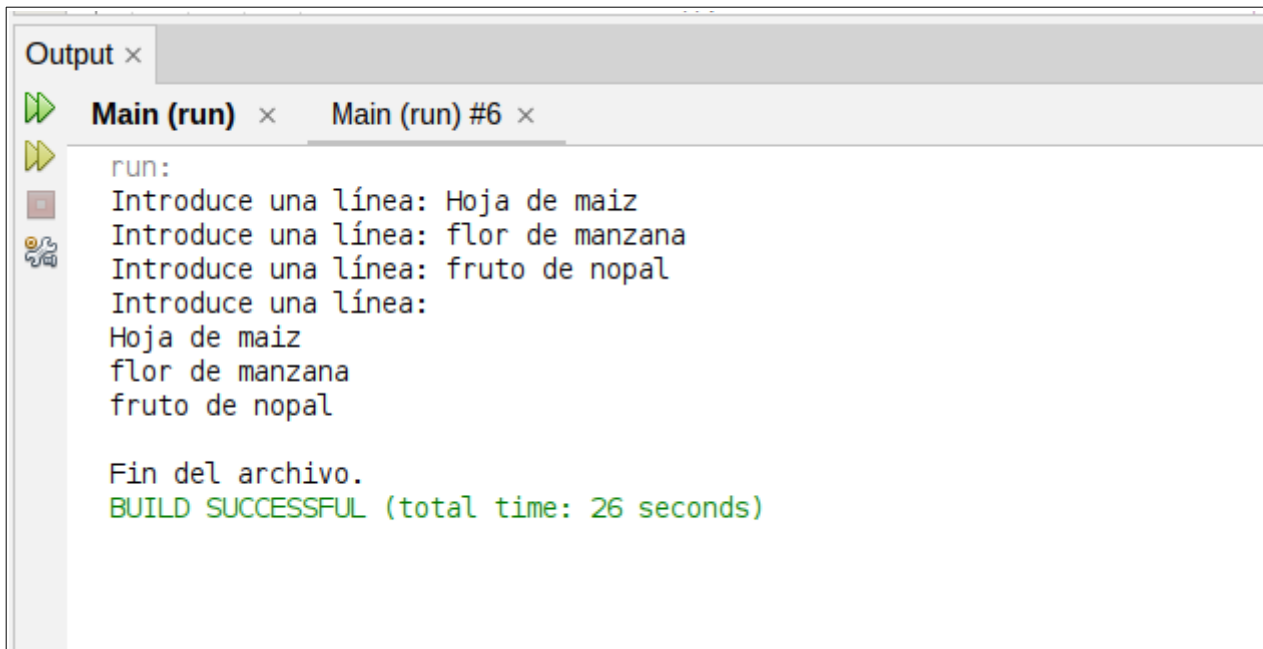
[Volver al índice](#)

11.17. Escribe un texto, línea a línea, de forma que, cada vez que se pulse Intro, se guarde la línea en un archivo binario. El proceso se termina cuando introduzcamos una línea vacía. Después el programa lee el texto completo del archivo y lo muestra por pantalla .

```

1  package Main;
2
3  import java.io.*;
4  import java.util.Scanner;
5
6  /**
7   * @author juancfr
8   */
9  public class Main {
10     /**
11      * 11.17. Escribe un texto, línea a línea, de forma que, cada vez que se
12      * pulse Intro, se guarde la línea en un archivo binario. El proceso se
13      * termina cuando introduzcamos una línea vacía. Después el programa lee el
14      * texto completo del archivo y lo muestra por pantalla .
15      */
16     final static String FILENAME = "texto.dat";
17
18     public static void main(String[] args) {
19         guardarTexto();
20         leerTexto();
21     }
22
23     public static void guardarTexto() {
24         try (ObjectOutputStream outputFile = new ObjectOutputStream(
25             new FileOutputStream(FILENAME))) {
26             Scanner scanner = new Scanner(System.in);
27             String line;
28             do {
29                 System.out.print("Introduce una línea: ");
30                 line = scanner.nextLine();
31                 outputFile.writeObject(line);
32             } while (!line.isEmpty());
33         } catch (IOException e) {
34             System.out.println("Error: " + e.getMessage());
35         }
36     }
37
38     public static void leerTexto() {
39         try (ObjectInputStream inputFile = new ObjectInputStream(
40             new FileInputStream(FILENAME))) {
41             Object obj;
42             while ((obj = inputFile.readObject()) != null) {
43                 String line = (String) obj;
44                 System.out.println(line);
45             }
46         } catch (EOFException e) {
47             System.out.println("Fin del archivo.");
48         } catch (IOException | ClassNotFoundException e) {
49             System.out.println("Error: " + e.getMessage());
50         }
51     }
52 }

```



The screenshot shows an IDE's output window with a tab labeled "Output x". Below the tab, there are two sub-tabs: "Main (run) x" and "Main (run) #6 x". The "Main (run) #6 x" tab is active. The output text is as follows:

```
run:
Introduce una línea: Hoja de maíz
Introduce una línea: flor de manzana
Introduce una línea: fruto de nopal
Introduce una línea:
Hoja de maíz
flor de manzana
fruto de nopal

Fin del archivo.
BUILD SUCCESSFUL (total time: 26 seconds)
```

[Volver al índice](#)

11.18. Un libro de firmas es útil para recoger los nombres de todas las personas que han pasado por un determinado lugar. Crea una aplicación que permita mostrar el libro de firmas o insertar un nuevo nombre (comprobando que no se encuentre repetido) usando el fichero binario firmas.dat.

[Volver al índice](#)

11.19. Por motivos puramente estadísticos se desea llevar constancia del número de llamadas recibidas cada día en una oficina. Para ello, al terminar cada jornada laboral se guarda dicho número al final de un archivo binario. Implementa una aplicación con un menú, que nos permita añadir el número correspondiente cada día y ver la lista completa en cualquier momento .

[Volver al índice](#)