

# TAREA TEMA 11 PRO

04/03/2023

Autor: Derimán Tejera Fumero

## 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.

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.

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

- a) `writeIntO-`
- b) `writeArrayIntO`.
- c) `readObjectO-`
- d) `writeObjectO`.

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();`

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`.

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.

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`.

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) Niña se pueden recuperar.

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().

11.10. Hay que cerrar los flujos binarios:

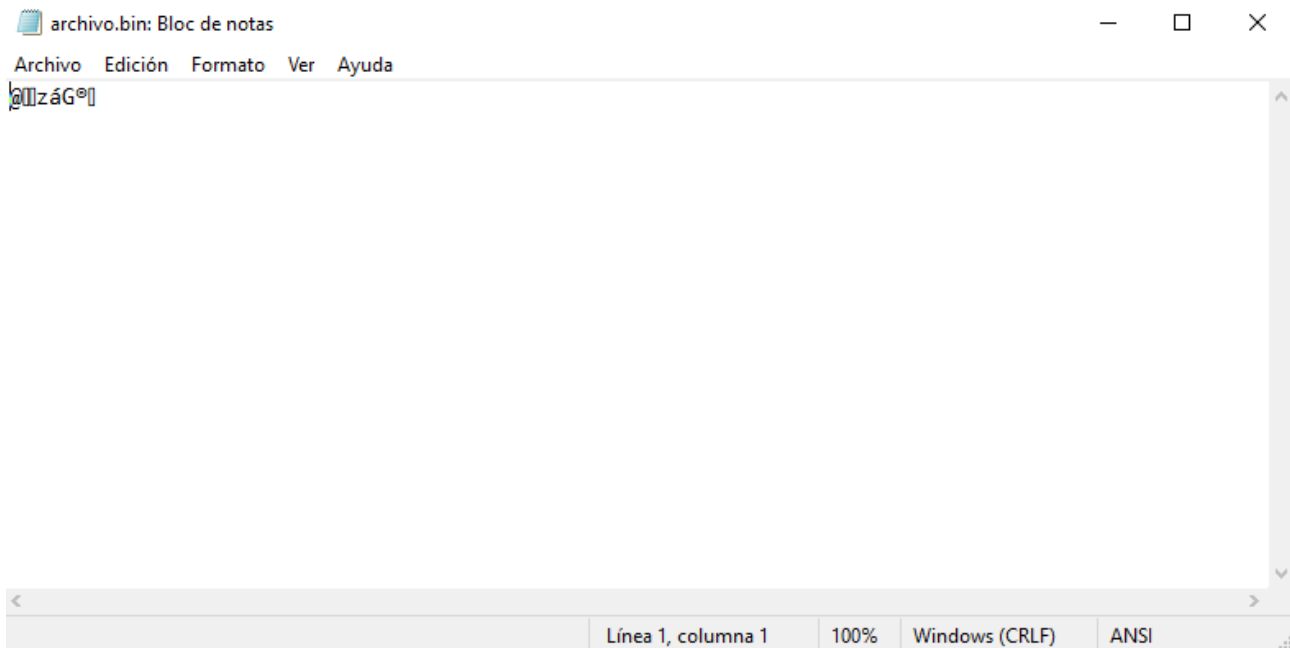
- a) Siempre.
- b) Una vez al día.
- c) Solo si no se han abierto con una estructura try-catch con recursos.
- d) Niña.

## Actividades de aplicación

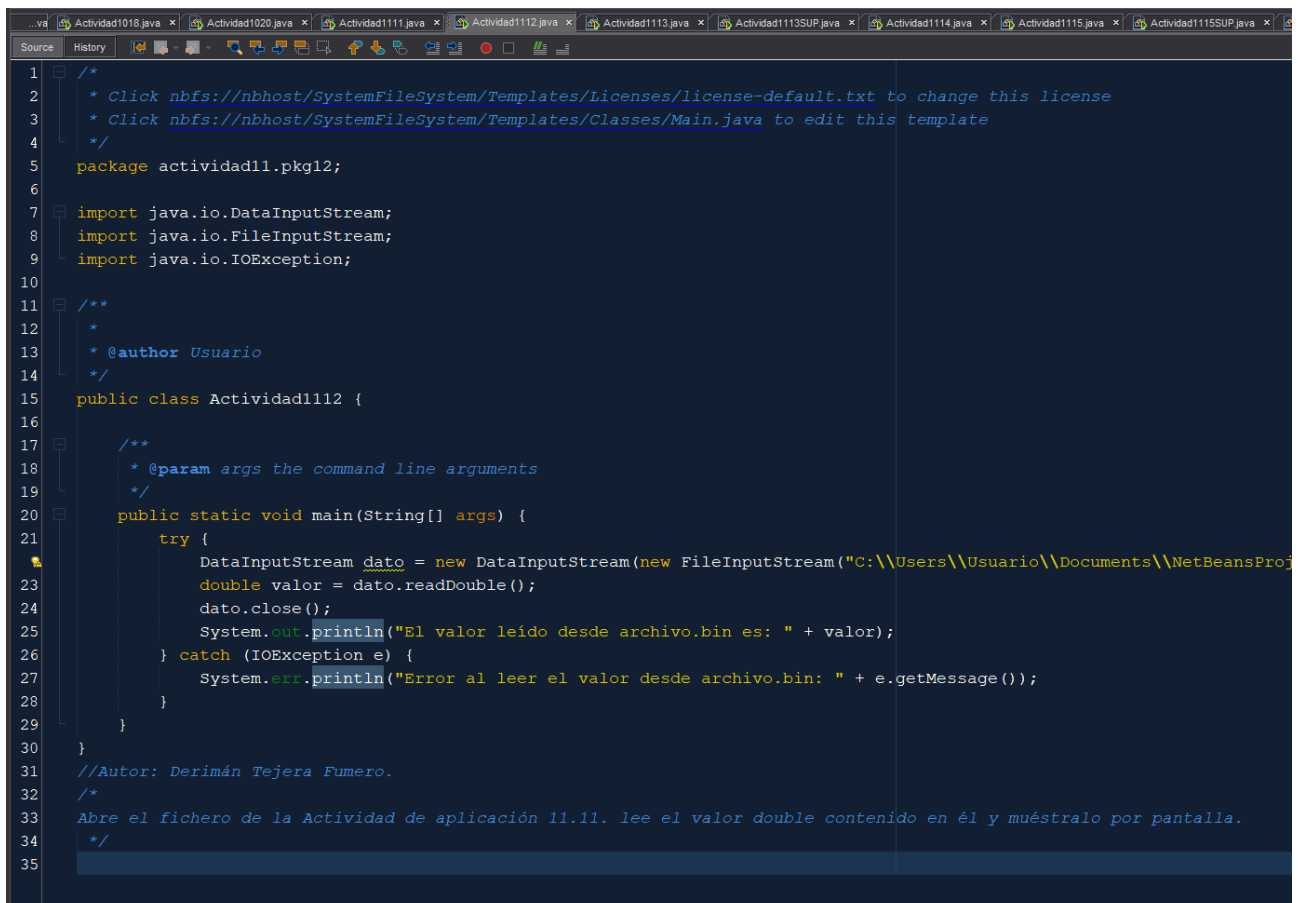
### 11.11. Pide un valor double por consola y guárdalo en un archivo binario.

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
4   */
5  package actividad11.pkg11;
6
7  import java.io.DataOutputStream;
8  import java.io.FileOutputStream;
9  import java.io.IOException;
10 import java.util.Scanner;
11 import java.util.Locale;
12
13 /**
14  *
15  * @author Usuario
16  */
17 public class Actividad1111 {
18
19     /**
20      * @param args the command line arguments
21      */
22     public static void main(String[] args) {
23         Scanner sc = new Scanner(System.in).useLocale(Locale.US);
24         System.out.print("Introduce un valor: ");
25         double valor = sc.nextDouble();
26         sc.close();
27
28         try {
29             DataOutputStream dato = new DataOutputStream(new FileOutputStream("archivo.bin"));
30             dato.writeDouble(valor);
31             dato.close();
32             System.out.println("El valor " + valor + " ha sido guardado en archivo.bin");
33         } catch (IOException e) {
34             System.err.println("Error al guardar el valor en archivo.bin: " + e.getMessage());
35         }
36     }
37 }
38 //Autor: Derimán Tejera Fumero.
39 /*
40 Pide un valor double por consola y guárdalo en un archivo binario.
41 */
```

```
Output - Actividad11.11 (run) X
run:
Introduce un valor: 5.02
El valor 5.02 ha sido guardado en archivo.bin
BUILD SUCCESSFUL (total time: 4 seconds)
```



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



```
Output - Actividad11.12 (run) X
run:
El valor leido desde archivo.bin es: 5.02
BUILD SUCCESSFUL (total time: 0 seconds)
```

**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.**

```
4  package actividad11.pkg13;
5
6
7  import java.io.DataInputStream;
8  import java.io.FileInputStream;
9  import java.io.IOException;
10
11  /**
12   *
13   * @author Usuario
14   */
15  public class Actividad1113 {
16
17      /**
18       * @param args the command line arguments
19       */
20      public static void main(String[] args) {
21          try {
22              DataInputStream dato = new DataInputStream(new FileInputStream("C:\\Users\\Usuario\\Documents\\NetBeansProjects\\Actividad11.13SUP\\tabla.bin"));
23
24              int n = dato.readInt();
25              double[] tabla = new double[n];
26
27              for (int i = 0; i < n; i++) {
28                  tabla[i] = dato.readDouble();
29              }
30
31              dato.close();
32
33              System.out.println("Tabla leida desde tabla.bin:");
34              for (int i = 0; i < n; i++) {
35                  System.out.println(tabla[i]);
36              }
37
38          } catch (IOException e) {
39              System.err.println("Error al leer la tabla desde tabla.bin: " + e.getMessage());
40          }
41      }
42  }
43  //Autor: Derimán Tejera Fumero.
44  /*
45  Escribe un programa que lea de un fichero binario una tabla de números double y después muestre el contenido de la tabla por consola.
46  */
```

```

Archivo  Edición  Formato  Ver  Ayuda
| e?ð      ?ð(ðÃ\)?ðQë... R?ðzágG@{ ?ðÉx
=p#?ðïïïïïf?ððÃ\ (ð?ñ .Që...?ñG@l|z áH?ñpÉx
=q?ñ"~~~~~$?ñÃ\ (ðÃ?ñë... Q1?ðl|z áG@l|?ð=pÉx
=?ðfffff?ð\ (ðÃ?ð .Që... , ?ðágG@l|z á?ð
=pÉx
?ð333333?ð\ (ðÃ\?ó... Që...?ó@l|z áG@?óx
=pÉx?ð      ?ð(ðÃ\)?ðQë... R?ðzágG@l|{ ?ðÉx
=p#?ðïïïïïf?ððÃ\ (ð?ð .Që...?ðG@l|z áH?ðpÉx
=q?ð"~~~~~$?ðÃ\ (ðÃ?ðë... Q1?ðl|z áG@l|?ð=pÉx
=?ðfffff?ð\ (ðÃ?ð .Që... , ?ðágG@l|z á?ð
=pÉx
?+333333?+\ (ðÃ\?+... Që...?+@l|z áG@?+x
=pÉx?ø      ?ø(ðÃ\)?øQë... R?øzágG@l|{ ?øÉx
=p#?øïïïïïf?øðÃ\ (ø?ù .Që...?ùG@l|z áH?ùpÉx
=q?ù"~~~~~$?ùÃ\ (ðÃ?ùë... Q1?ùl|z áG@l|?ù=pÉx
=?ùfffff?ù\ (ðÃ?ù .Që... , ?ùágG@l|z á?ù
=pÉx
?ù333333?ù\ (ðÃ\?ù... Që...?ù@l|z áG@?ùx
=pÉx?ù      ?ù(ðÃ\)?ùQë... R?ùzágG@l|{ ?ùÉx
=p#?ùïïïïïf?ùðÃ\ (ø?ý .Që...?ýG@l|z áH?ýpÉx
=q?ý"~~~~~$?ýÃ\ (ðÃ?ýë... Q1?ùl|z áG@l|?ù=pÉx
=?ùfffff?ù\ (ðÃ?ù .Që... , ?ùágG@l|z á?ù
=pÉx
?ý333333?ý\ (ðÃ\?ý... Që...?ý@l|z áG@?ýx
=pÉx@

```

```
Output - Actividad11.13 (run) X
run:
Tabla leída desde tabla.bin:
1.0
1.01
1.02
1.03
1.04
1.05
1.06
1.07
1.08
1.09
1.1
1.11
1.12
1.13
1.14
1.15
1.16
1.17
1.18
1.19
1.2
1.21
1.22
1.23
1.24
1.25
1.26
1.27
1.28
1.29
1.3
1.31
1.32
1.33
1.34
1.35
1.36
1.37
1.38
1.39
1.4
1.41
1.42
1.43
1.44
1.45
1.46
1.47
1.48
1.49
1.5
1.51
1.52
```

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



```
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
4   */
5   package actividad11.pkg14;
6
7   import java.io.DataInputStream;
8   import java.io.DataOutputStream;
9   import java.io.FileInputStream;
10  import java.io.FileOutputStream;
11  import java.io.IOException;
12  import java.util.Scanner;
13
14  /**
15   *
16   * @author Usuario
17   */
18  public class Actividad1114 {
19
20      /**
21       * @param args the command line arguments
22       */
23      public static void main(String[] args) {
24          Scanner scanner = new Scanner(System.in);
25          System.out.print("Introduce una frase: ");
26          String frase = scanner.nextLine();
27
28          try (DataOutputStream dato = new DataOutputStream(new FileOutputStream("frase.bin"))) {
29              dato.writeUTF(frase);
30              System.out.println("Frase guardada en frase.bin");
31          } catch (IOException e) {
32              System.err.println("Error al escribir la frase en frase.bin: " + e.getMessage());
33          }
34
35          try (DataInputStream dato = new DataInputStream(new FileInputStream("frase.bin"))) {
36              String fraseRecuperada = dato.readUTF();
37              System.out.println("Frase recuperada: " + fraseRecuperada);
38          } catch (IOException e) {
39              System.err.println("Error al leer la frase de frase.bin: " + e.getMessage());
40          }
41      }
42  }
43  //Autor: Derimán Tejera Fumero.
44  /*
45  Introduce por teclado una frase y guárdala en un archivo binario. A continuación, recupérala y muéstrala por pantalla.
46  */
```

```
Output - Actividad11.14 (run) X
run:
Introduce una frase: El gato con botas perdió las botas
Frase guardada en frase.bin
Frase recuperada: El gato con botas perdi  las botas
BUILD SUCCESSFUL (total time: 14 seconds)
```

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

```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
4  */
5  package actividad11.pkg15;
6
7  import java.io.BufferedReader;
8  import java.io.FileReader;
9  import java.io.FileWriter;
10 import java.io.IOException;
11 import java.io.PrintWriter;
12
13 /**
14 *
15 * @author Usuario
16 */
17 public class Actividad1115 {
18
19     /**
20     * @param args the command line arguments
21     */
22     public static void main(String[] args) {
23         try (BufferedReader leerNumeros = new BufferedReader(new FileReader("C:\\Users\\Usuario\\Documents\\NetBeansProjects\\Actividad11.15SUF\\numeros.dat")); PrintWriter pares = new PrintW
24
25             String linea;
26             while ((linea = leerNumeros.readLine()) != null) {
27                 int num = Integer.parseInt(linea.trim());
28                 if (num % 2 == 0) {
29                     pares.println(num);
30                 } else {
31                     impares.println(num);
32                 }
33             }
34             System.out.println("Números escritos en los archivos pares.dat e impares.dat.");
35         } catch (IOException e) {
36             System.out.println("Error al leer o escribir archivo: " + e.getMessage());
37         }
38     }
39 }
40
41 //Autor: Derimán Tejera Fumero.
42 /*
43 Implemento un programa que lea números enteros desde el fichero numeros.dat y los vaya guardando en los ficheros par.dat e impar.dat, según su paridad.
44 */
```

numeros.dat: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
```

Línea 1, columna 1 100% Windows (CRLF) UTF-8

impares.dat: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Línea 1, columna 1 100% Windows (CRLF) UTF-8

pares.dat: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Línea 1, columna 1 100% Windows (CRLF) UTF-8

**11.16. Implemento 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.**

```
...va Actividad1018.java x Actividad1020.java x Actividad1111.java x Actividad1112.java x Actividad1113.java x Actividad1113SUP.java x Actividad1114.java x Actividad1115.java x Actividad1115
Source History
99         indiceLista++;
100     }
101     while (indice2 < nombres2.length) {
102         listaNombres[indiceLista] = nombres2[indice2];
103         indice2++;
104         indiceLista++;
105     }
106     return listaNombres;
107 }
108
109 public static void guardarNombres(String[] nombres) {
110     try {
111         BufferedWriter bw = new BufferedWriter(new FileWriter("nombres.dat"));
112         for (String nombre : nombres) {
113             bw.write(nombre + "\n");
114         }
115         bw.close();
116     } catch (IOException e) {
117         System.out.println("Error al escribir en el archivo");
118     }
119 }
120
121 public static String[] insertarNombre(String[] nombres, String nombre) {
122     String nuevosNombres[] = new String[nombres.length + 1];
123     int indice = 0;
124     while (indice < nombres.length && nombre.compareTo(nombres[indice]) > 0) {
125         nuevosNombres[indice] = nombres[indice];
126         indice++;
127     }
128     nuevosNombres[indice] = nombre;
129     while (indice < nombres.length) {
130         nuevosNombres[indice + 1] = nombres[indice];
131         indice++;
132     }
133     return nuevosNombres;
134 }
135
136 }
137
138 //Autor: Derimán Tejera Fumero.
139 /*
140 Implementa una aplicación que gestione una lista de nombres ordenada por orden alfabético.
141 Al arrancar se leerá de un fichero los nombres insertados anteriormente y se pedirán nombres
142 nuevos hasta que se introduzca la cadena <<fin>>. Cada nombre que se introduzca deberá añadirse
143 a los que ya habrá, de forma que la lista permanezca ordenada. Al terminar, se guardará en
144 el fichero la lista actualizada.
145 */
```

Output X

Actividad 1116 (run) x Actividad 1116 (run) #2 x

run:

Los nombres contenidos en el archivo nombres.dat son:

Ana Falcó

Antonio Dominguez

Fernando Sánchez

Gallon Ful

Julio Maiz

Luisa García

Luís Hierro

Pablo Medina

Pedro López

Pedro Ortiz

Pedro Pastor

Pilar Jabalón

Zul Zol

Zur Zur

Introduce un nombre (introduce 'fin' para terminar):

```
Introduce un nombre (introduce 'fin' para terminar): Nicolás Caminos
Introduce otro nombre (introduce 'fin' para terminar): Manolo Peral
Introduce otro nombre (introduce 'fin' para terminar): Ana Glz
Introduce otro nombre (introduce 'fin' para terminar): fin
```

La lista actualizada de nombres es la siguiente:

Los nombres contenidos en el archivo nombres.dat son:

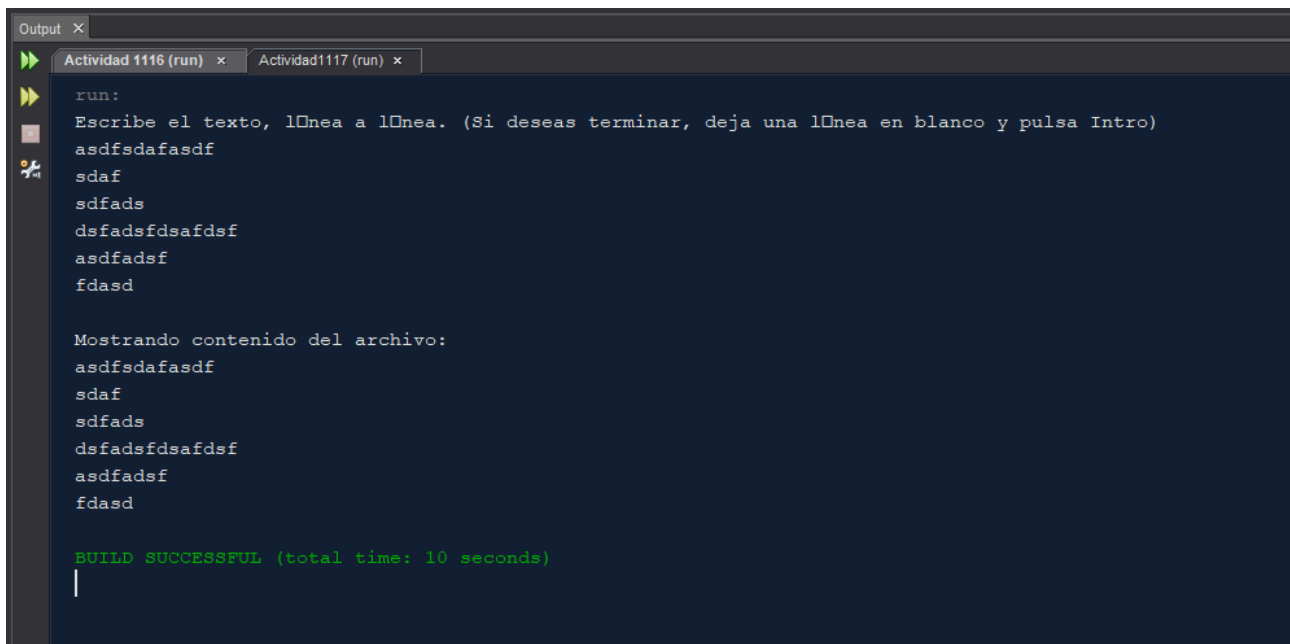
```
Ana Falc 
Ana Glz
Antonio Dominguez
Fernando S nchez
Gallon Ful
Julio Maiz
Luisa Garc a
Lu s Hierro
Manolo Peral
Nicol s Caminos
Pablo Medina
Pedro L pez
Pedro Ortiz
Pedro Pastor
Pilar Jabal 
Zul Zol
Zur Zur
```

**BUILD SUCCESSFUL** (total time: 1 minute 31 seconds)

|

**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.**

```
...va Actividad1018.java x Actividad1020.java x Actividad1111.java x Actividad1112.java x Actividad1113.java x Actividad1113SUP.java x Actividad1114.java x Actividad1115.java x Actividad1115SUP.java x Actividad1116.java
Source History
26 try {
27     FileInputStream fis = new FileInputStream("texto.bin");
28     DataInputStream dis = new DataInputStream(fis);
29     String linea;
30     System.out.println("Mostrando contenido del archivo:");
31     while (dis.available() > 0) {
32         linea = dis.readUTF();
33         System.out.println(linea);
34     }
35     dis.close();
36     fis.close();
37 } catch (IOException e) {
38     System.out.println("Error al leer el archivo");
39 }
40 }
41
42 public static void guardarTexto() {
43     try {
44         FileOutputStream fos = new FileOutputStream("texto.bin");
45         DataOutputStream dos = new DataOutputStream(fos);
46         Scanner scanner = new Scanner(System.in);
47         String linea;
48
49         do {
50             linea = scanner.nextLine();
51             dos.writeUTF(linea);
52         } while (!linea.isEmpty());
53         dos.close();
54         fos.close();
55         scanner.close();
56     } catch (IOException e) {
57         System.out.println("Error al guardar el texto en el archivo");
58     }
59 }
60 }
61
62 //Autor: Derimán Tejera Fumero.
63 /*
64  Escribe un texto, línea a línea, de forma que cada vez que se pulse Intro. se guarde la línea en un archivo binario.
65  El proceso se termina cuando introduzcamos una línea vacía. Después el programa lee el texto completo del archivo y
66  lo muestra por pantalla.
67  */
68
```



```
Output X
Actividad 1116 (run) x Actividad1117 (run) x
run:
Escribe el texto, lOnea a lOnea. (Si deseas terminar, deja una lOnea en blanco y pulsa Intro)
asdfsdafeasdf
sdafe
sdfads
dsfadsfdafeasdf
asdfsdafe
fdafe

Mostrando contenido del archivo:
asdfsdafeasdf
sdafe
sdfads
dsfadsfdafeasdf
asdfsdafe
fdafe

BUILD SUCCESSFUL (total time: 10 seconds)
```

**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.**

```

78
79     private static void guardarFirmas() {
80         try (ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(FILENAME))) {
81             out.writeObject(firmas);
82         } catch (IOException e) {
83             System.out.println("Error al guardar el fichero de firmas: " + e.getMessage());
84         }
85     }
86
87     private static void mostrarFirmas() {
88         System.out.println("");
89         System.out.println("");
90         System.out.println("Las firmas almacenadas son: ");
91         for (int i = 0; i < MAX_FIRMAS; i++) {
92             if (firmas[i] != null) {
93                 System.out.println("[ " + (i + 1) + " ] " + firmas[i]);
94             }
95         }
96         System.out.println("");
97         System.out.println("");
98     }
99
100    private static void agregarFirma(String nombre) {
101        for (int i = 0; i < MAX_FIRMAS; i++) {
102            if (firmas[i] == null) {
103                firmas[i] = nombre;
104                break;
105            }
106        }
107    }
108
109    private static int buscarFirma(String nombre) {
110        for (int i = 0; i < MAX_FIRMAS; i++) {
111            if (firmas[i] != null && firmas[i].equalsIgnoreCase(nombre)) {
112                return i;
113            }
114        }
115        return -1;
116    }
117
118 }
119
120 //Autor: Derimán Tejera Fumero.
121 /*
122  Un libro de firmas es útil para recoger los nombres de todas las personas que han pasado por un determinado lugar.
123  Crea una aplicación que permita mostrar el libro de firmas o insertar un nuevo nombre (comprobando que no se encuentre
124  repetido) usando el fichero binario firmas.dat.
125  */

```

```

Output - Actividad 1118 (run) #2
run:
MENU DE OPCIONES
1. Mostrar firmas
2. Añadir firma
3. Salir
Seleccione una opción:2
Introduzca un nombre:Marta Puente
Nombre añadido a la lista.

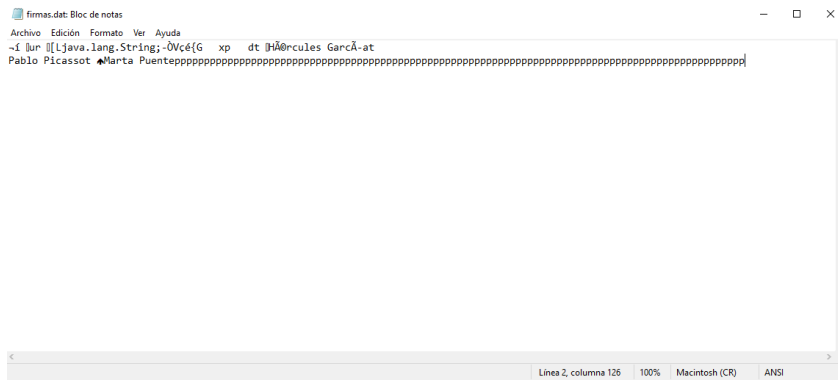
MENU DE OPCIONES
1. Mostrar firmas
2. Añadir firma
3. Salir
Seleccione una opción:1

Las firmas almacenadas son:
[1] Hércules García
[2] Pablo Picasso
[3] Marta Puente

MENU DE OPCIONES
1. Mostrar firmas
2. Añadir firma
3. Salir
Seleccione una opción:

```





**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. Implemento una aplicación con un menú, que nos permita añadir el número correspondiente cada día y ver la lista completa en cualquier momento.**

```

83 try {
84     DataOutputStream out = new DataOutputStream(new FileOutputStream("telefonos.bin", true));
85     LocalDate fechaActual = LocalDate.now();
86     DateTimeFormatter formato = DateTimeFormatter.ofPattern("dd-MM-yyyy");
87     String fechaFormateada = fechaActual.format(formato);
88     out.writeUTF(fechaFormateada);
89     out.writeInt(numllamadas);
90     out.close();
91 } catch (IOException e) {
92     System.out.println("Error de E/S.");
93 }
94 }
95
96 public static void mostrarLista() {
97     try {
98         System.out.println("");
99         System.out.println("");
100         System.out.println("Los números almacenados son: ");
101         DataInputStream in = new DataInputStream(new FileInputStream("telefonos.bin"));
102         while (in.available() > 0) {
103             String fecha = in.readUTF();
104             int numllamadas = in.readInt();
105             System.out.println(fecha + " - " + numllamadas);
106         }
107         in.close();
108         System.out.println("");
109         System.out.println("");
110     } catch (IOException e) {
111         System.out.println("Error de E/S.");
112     }
113 }
114
115 public static void borrarDatosAlmacenados() {
116     try {
117         new FileOutputStream("telefonos.bin").close();
118     } catch (IOException e) {
119         System.out.println("Error de E/S.");
120     }
121 }
122 }
123
124 //Autor: Derimán Tejera Fumero.
125 /*
126 Por motivos pusemente estadísticos se desea llevar constancia del número de llamadas recibidas cada día en una oficina.
127 Para ello, al terminar cada jornada laboral se guarda dicho número al final de un archivo binario. Implementa una
128 aplicación con un menú, que nos permita añadir el número correspondiente cada día y ver la lista completa en cualquier momento.
129 */

```



```
run:
MENU DE OPCIONES
1. Añadir número de llamadas
2. Mostrar lista completa
3. Borrar contenido del archivo
0. Salir
Seleccione una opción: 2
```

```
Los números almacenados son:
01-03-2023 - 5289
03-03-2023 - 8052
```

```
MENU DE OPCIONES
1. Añadir número de llamadas
2. Mostrar lista completa
3. Borrar contenido del archivo
0. Salir
Seleccione una opción: |
```