

Practica 1 Repaso Ficheros.

Indice:

1. Código Fuente usado
2. Ejercicios

1. Código Fuente usado:

En esta practica o relación de ejercicios se ha creado una clase con código que se ha ido reutilizando a lo largo de los ejercicios con el objetivo de reutilizarlo, este se ha llamado "FicherosTerminal". Los mas importantes son:

Metodo 1: public static File introDirectorio(Scanner key)

Este método sirve para preguntarle al usuario por un directorio y validarlo

```
public static File introDirectorio(Scanner key)throws Exception{
    /*
    variables:
    aux objeto File que comprueba si el directorio es correcto
    control que gestiona el bucle
    */

    File aux = null;
    boolean control = false;
    // inicio del bucle
    do{
        System.out.println("introduzca un directorio: ");
        // sen introduce por terminale el directorio en la variable File
        aux = new File(key.next());
        // se comprueba si lo introducido es un directorio como tal
        if(!aux.isDirectory()){
            // en el caso que no sea asi se manda un mensaje por pantalla
            // y el bucle continua
            System.out.println("no es un directorio valido");
        }
        else{
            // en caso contrario se cambia la variable control y termian el bucle
            control = true;
        }
    }while(!control);
    return aux;
}
```

Metodo 2: public static File introFichero(Scanner key)

Este metodo hace lo mismo que el anterior con la diferencia que sirve para ficheros

```
public static File introFichero(Scanner key)throws Exception{
    /*
    variables:
    aux objeto File que comprueba si el directorio es correcto
    control que gestiona el bucle
    */
    File aux = null;
    boolean control = false;
    // inicio del bucle
    do{
        System.out.println("introduzca el directiro de un fichero: ");
        // sen introduce por terminale el directorio en la variable File
        aux = new File(key.next());
        // se comprueba si es un fichero valido

        if(!aux.isFile()){
            // en el caso que no sea asi se manda un mensaje por pantalla
            // y el bucle continua
            System.out.println("no es un directorio valido");
        }
        else{
            // en caso contrario se cambia la variable control y termian el bucle
            control = true;
        }

    }while(!control);
    return aux;
}
```

Metodo 3: public static File introFicheroEscritura(Scanner key)

igual al método 2 con la diferencia que permite decidir si se quiere crear el fichero

```
public static File introFicheroEscritura(Scanner key)throws Exception{
    /*
    variables:
    aux objeto File que comprueba si el directorio es correcto
    control que gestiona el bucle
    */
    File aux = null;
    boolean control = false;
    // inicio del bucle
    do{
        System.out.println("introduzca el directiro de un fichero: ");
        // sen introduce por terminale el directorio en la variable File
        aux = new File(key.next());
        // se comprueba si existe el fichero
        if(!aux.exists()){
            // si no existe se le pregunta al usuario si desea crearlo
        }
    }while(!control);
    return aux;
}
```

```
System.out.println("el archivo no existe, desea crearlo?(s/n)");
char opcion = ' ';
do{
    opcion = key.next().charAt(0);
    switch(opcion){
        case 's':
        case 'S':
            return aux;
        case 'n':
        case 'N':
            break;
        default:
            System.out.println("respuesta no valida");
            break;
    };
}while((opcion != 's' || opcion != 'S' ) &&
    (opcion != 'n' || opcion != 'N')
);
}else{
    if(!aux.isFile()){
        // en el caso que no sea asi se manda un mensaje por pantalla
        // y el bucle continua
        System.out.println("no es un directorio valido");
    }
    else{
        // en caso contrario se cambia la variable control y termian el bucle
        control = true;
    }
}
}while(!control);
return aux;
}
```

El Resto de métodos aparecerán en el ejercicio que se usa.

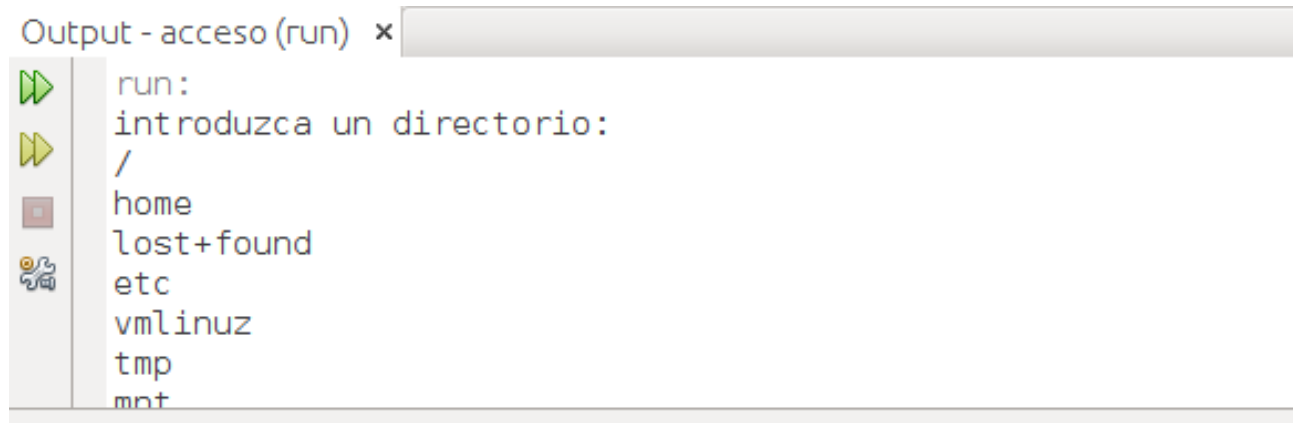
2. Ejercicios

Ejercicio 1

Realiza un programa Java que muestre los ficheros de un directorio. El nombre del directorio se pasará al programa desde la línea de comandos al ejecutarlo (muestra el primer nivel).

```
package ejercicios.ficheros;
import java.io.File;
import java.util.Scanner;
/**
 *
 * @author likendero
 */
public class PrimerEjercicio {
    // variable que sirve para almacenar el directorio
    private static File directorio = null;
    // variable de tipo scanner para la introduccion por terminal
    private static Scanner key = new Scanner(System.in);
    /**
     * metodo principal, inicio
     * @param args
     */
    public static void main(String[] args) {
        boolean control = false;
        do{
            try{
                directorio = FicherosTerminal.introDirectorio(key);
                // recorrido de los elementos encontrados en el directorio
                for(String i: directorio.list()){
                    System.out.println(i);
                }
                control = true;
            }catch(NullPointerException nu){
                System.out.println("error en la ejecucion");
            }
            // control de posibles errores
        }catch(Exception ex){
            System.out.println("error en la introduccion");
            key.next();
        }
    }while(!control);
}
```

Resultado:



```
Output - acceso (run) x
run:
introduzca un directorio:
/
home
lost+found
etc
vmlinuz
tmp
mnt
```

Ejercicio 2

Modifica el programa anterior o haz un nuevo programa que permita listar de forma recursiva los archivos y posibles subdirectorios del directorio indicado desde la línea de comandos.

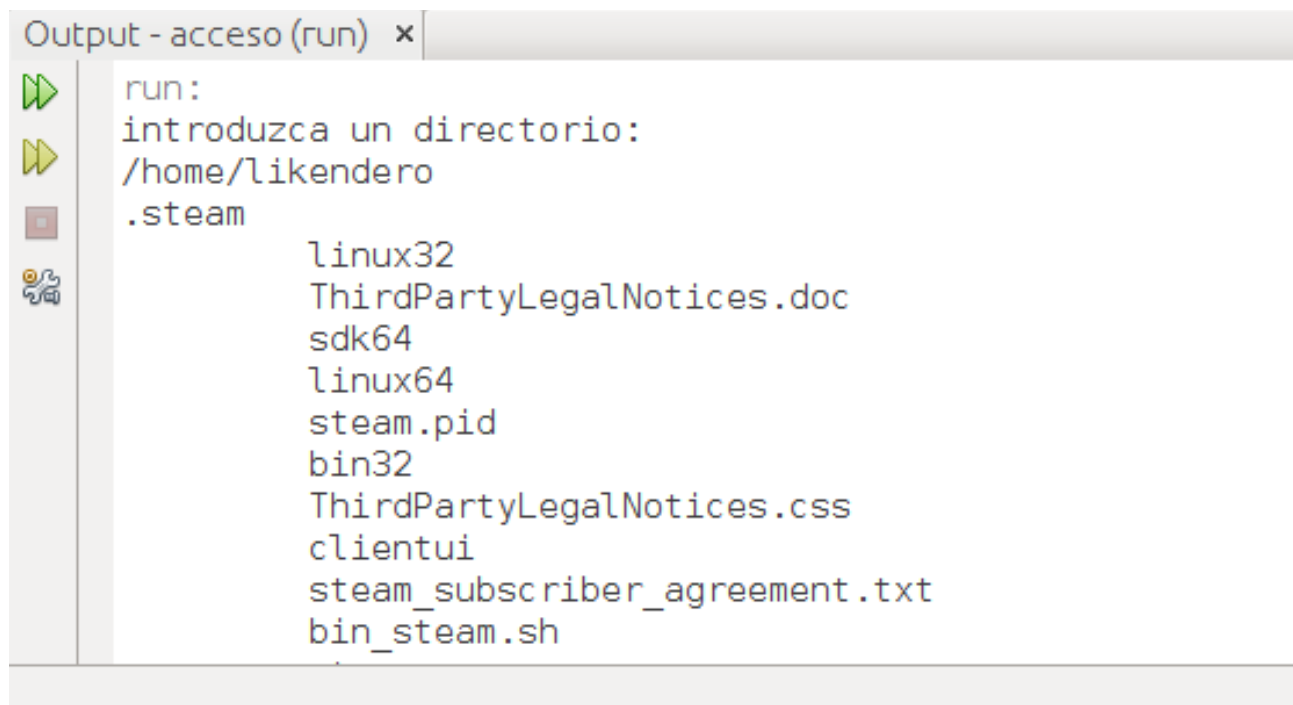
```
package ejercicios.ficheros;

import java.io.File;
import java.util.Scanner;

/**
 * listar todos los directorios y sub directorios
 * @author likendero
 */
public class SegundoEjercicio {
    // variable que sirve para almacenar el directorio
    private static File directorio = null;
    // variable de tipo scanner para la introduccion por terminal
    private static Scanner key = new Scanner(System.in);
    /**
     * metodo principal
     * @param args
     */
    public static void main(String[] args) {
        boolean control = false;
        File aux = null;
        do{
            try{
                directorio = FicherosTerminal.introDirectorio(key);
                // recorrido de los elementos encontrados en el directorio
                for(String i: directorio.list()){
                    System.out.println(i);
                    aux = new File(directorio.getAbsolutePath()+"/"+i);
                    if( aux.isDirectory()){
                        for(String j: aux.list()){
                            System.out.println("\t " + j);
                        }
                    }
                }
            }
        }
    }
}
```

```
        }  
    }  
    }  
    control = true;  
    // en el caso que el directorio sig siendo nulo  
}catch(NullPointerException nu){  
    System.out.println("error en la ejecucion");  
    nu.printStackTrace();  
    // control de posibles errores  
}catch(Exception ex){  
    System.out.println("error en la introduccion");  
    ex.printStackTrace();  
}  
}while(!control);  
}  
}
```

Resultado:



```
Output - acceso (run) x  
run:  
introduzca un directorio:  
/home/likendero  
.steam  
    linux32  
    ThirdPartyLegalNotices.doc  
    sdk64  
    linux64  
    steam.pid  
    bin32  
    ThirdPartyLegalNotices.css  
    clientui  
    steam_subscriber_agreement.txt  
    bin_steam.sh
```

Ejercicio 3

Realiza un programa que elimine de forma recursiva un directorio. Se debe contemplar la posibilidad de que el directorio contenga subdirectorios a diferentes niveles de profundidad. El nombre del directorio se pasará al programa desde la línea de comandos al ejecutarlo.

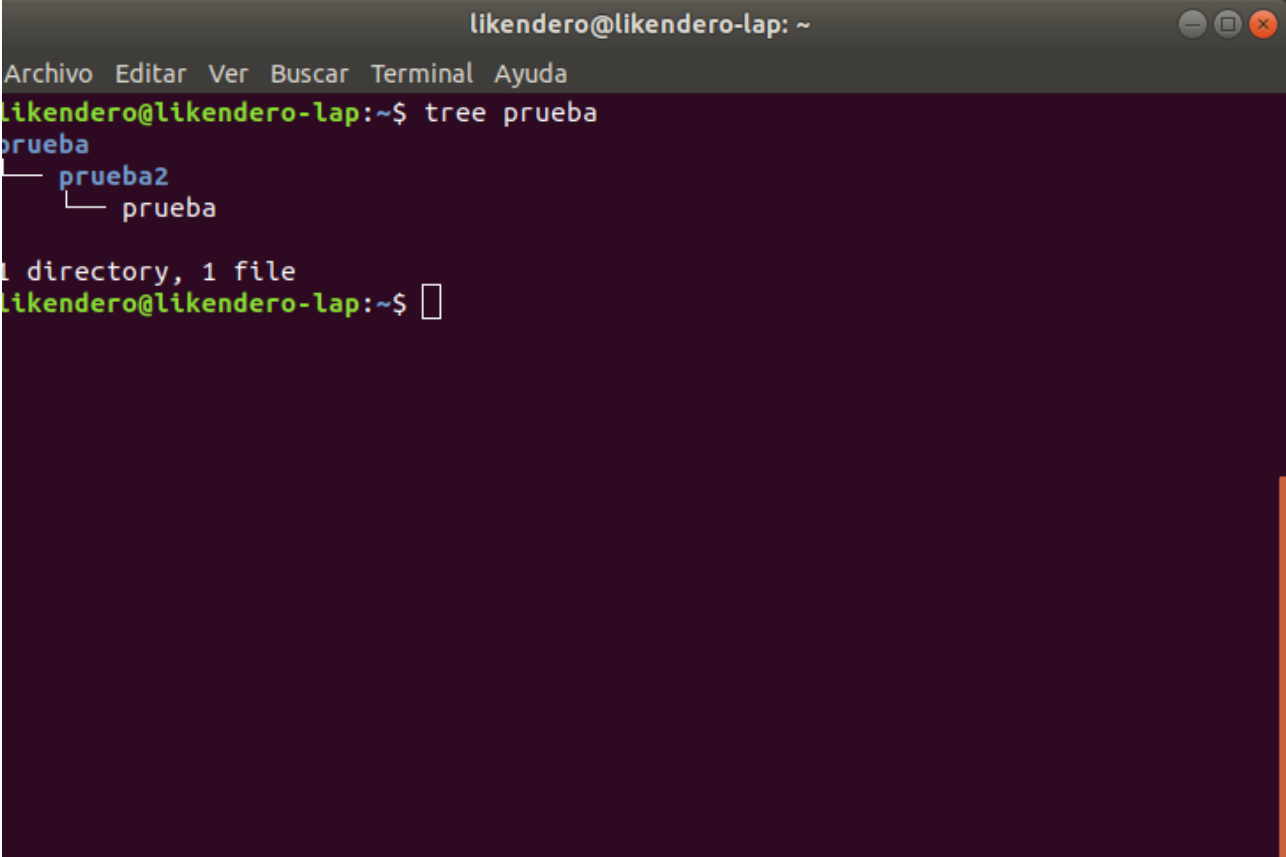
```
package ejercicios.ficheros;
import java.io.File;
import java.util.Scanner;

/**
 * programa que borra un directorio
 * @author likendero
 */
public class tercerEjercicio {
    // variable que almacena el directorio de trabajo (el que se borra)
    private static File directorio = null;
    private static Scanner key = new Scanner(System.in);
    /**
     * metodo principal
     * @param args
     */
    public static void main(String[] args) {
        boolean control = false;
        // bucle :3
        do{
            try{
                directorio = FicherosTerminal.introDirectorio(key);
                // se intenta borrar el directorio
                if(!directorio.delete()){
                    // se trata de borrar los archivos internos
                    FicherosTerminal.recorridoArchivos(directorio);
                    if(!directorio.delete()) System.out.println("no se borra");
                    else System.out.println("borrado satisfactorio");
                }else{
                    System.out.println("borrado satisfactorio");
                }
                control = true;
                // control null
            }catch(NullPointerException nu){
                System.out.println("error en la ejecucion, puede ser que falten permisos");
                nu.printStackTrace();
                key.nextLine();
                // control de excepciones inesperadas
            }catch(SecurityException se){
                System.out.println("existen archivos imborrables");
                se.printStackTrace();
                // se acaba el bucle ya que la eliminacion es imposible
                control = true;
            }catch(Exception ex){
                System.out.println("error inesperado o en la introduccion");
                key.nextLine();
            }
        } while (!control);
    }
}
```

Javier González Rives





```
        control = true;
    }
    }while(!control);
}
}
```

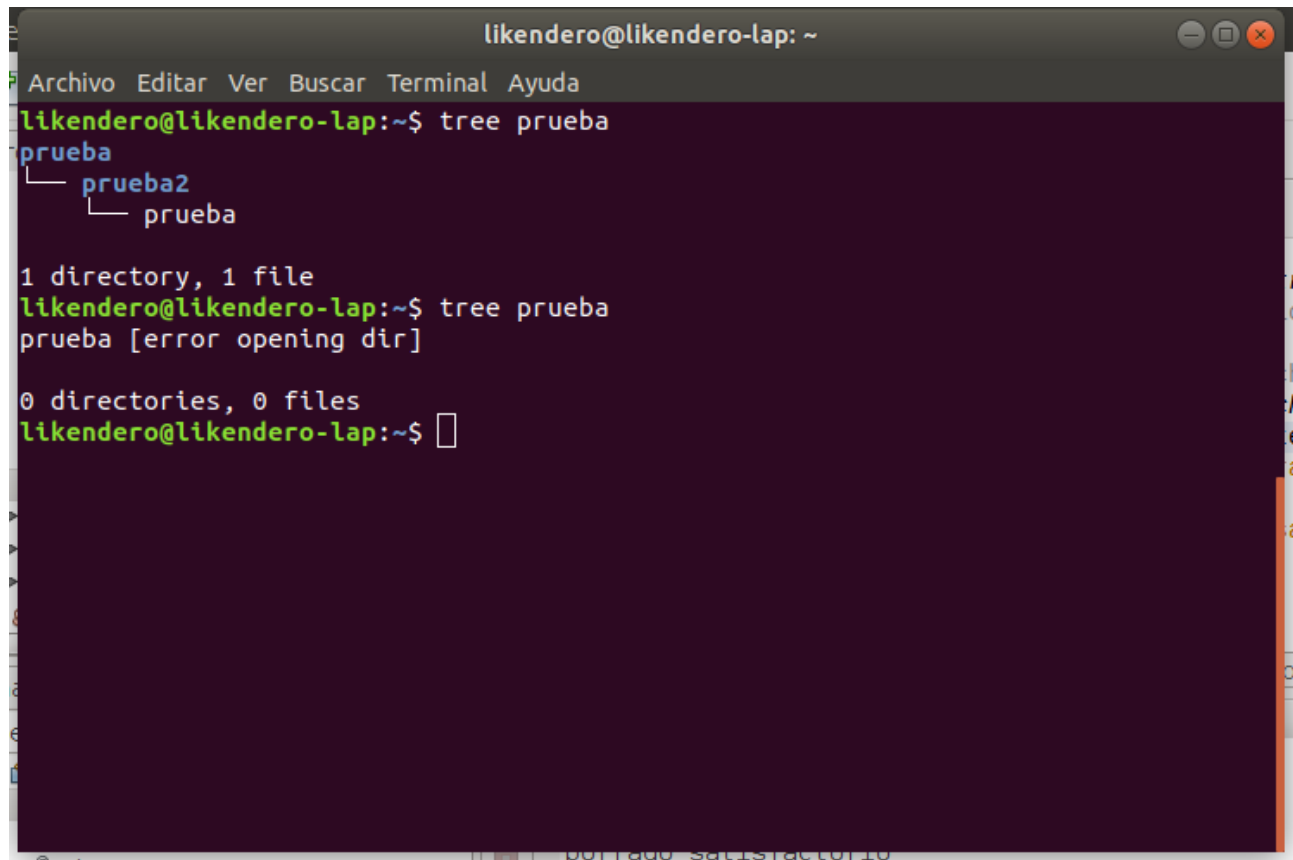
Resultado:



```
likendero@likendero-lap: ~
Archivo Editar Ver Buscar Terminal Ayuda
likendero@likendero-lap:~$ tree prueba
preuba
├── prueba2
│   └── prueba
└── 1 directory, 1 file
likendero@likendero-lap:~$
```

Output - acceso (run) x

 run:
 introduzca un directorio:
/home/likendero/prueba
 borrado satisfactorio
 BUILD SUCCESSFUL (total time: 1 minute 28 seconds)



```
likendero@likendero-lap: ~  
Archivo  Editar  Ver  Buscar  Terminal  Ayuda  
likendero@likendero-lap:~$ tree prueba  
prueba  
├── prueba2  
│   └── prueba  
  
1 directory, 1 file  
likendero@likendero-lap:~$ tree prueba  
prueba [error opening dir]  
  
0 directories, 0 files  
likendero@likendero-lap:~$
```

Ejercicio 4

Realiza un programa que liste todos los ficheros modificados entre dos fechas en un directorio dado. Con el fin de facilitar la elaboración del programa, considera las fechas y el directorio como valores dados en el programa. Para realizar el ejercicio:

FicherosTerminal

```
/**  
 * metodo para filtrar archivos segun un intervalo de fechas  
 * @param dir  
 * @param filtro  
 * @return  
 */  
public static String directorioFechas(File dir,FiltroFecha filtro){  
    // variable para devolver la cadena  
    String salida = "";  
  
    for(File i: dir.listFiles((FileFilter) filtro)){  
        salida += i.getName() + "\n";  
    }  
    return salida;  
}  
/**  
 * metodo para preguntar al usuario por una fecha  
 * @param key  
 * @return
```

```
*/
public static GregorianCalendar fechas(Scanner key){
    int dia = 0;
    int mes = 0;
    int anno = 0;
    boolean control = false;
    // intro del dia
    do{
        try{
            // introduccion de los parametros
            System.out.println("introduce el dia");
            dia = key.nextInt();
            System.out.println("introduce mes");
            mes = key.nextInt();
            System.out.println("introduce año");
            anno = key.nextInt();
            // comprobacion de la validez de la fecha
            LocalDate.of(anno,mes,dia);
            control = true;
        }catch(DateTimeException da){
            System.out.println("la fecha no es correcta");
        }
    }while(!control);

    return new GregorianCalendar(anno, mes, dia);
}
}
```

Ejercicio

```
package ejercicios.ficheros;

import java.io.File;
import java.util.GregorianCalendar;
import java.util.Scanner;

/**
 *
 * @author likendero
 */
public class CuartoEjercicio {
    private static Scanner key = new Scanner(System.in);
    private static File directorio = null;
    /**
     * metodo principal
     * @param args
     */
    public static void main(String[] args) {
        GregorianCalendar fecha1 = null;
        GregorianCalendar fecha2 = null;
        try{
            directorio = FicherosTerminal.introDirectorio(key);
        }
```

```
        // fecha de inicio
        System.out.println("fecha inicio de intervalo");
        fecha1 = FicherosTerminal.fechas(key);
        // fecha de fin
        System.out.println("fecha fin de intervalo");
        fecha2 = FicherosTerminal.fechas(key);
        System.out.println("archivos:");
        // creacion del filtro
        FiltroFecha filtro = new FiltroFecha(fecha1, fecha2);
        // se imprimen los ficheros
        System.out.println(FicherosTerminal.directorioFechas(directorio, filtro));
    }catch(NullPointerException nu){
        System.out.println("error en la ejecucion");
        nu.printStackTrace();
    }catch(Exception ex){
        System.out.println("error");
        ex.printStackTrace();
    }
}
}
```

Este ejercicio no he sido capaz de terminarlo

Ejercicio 5

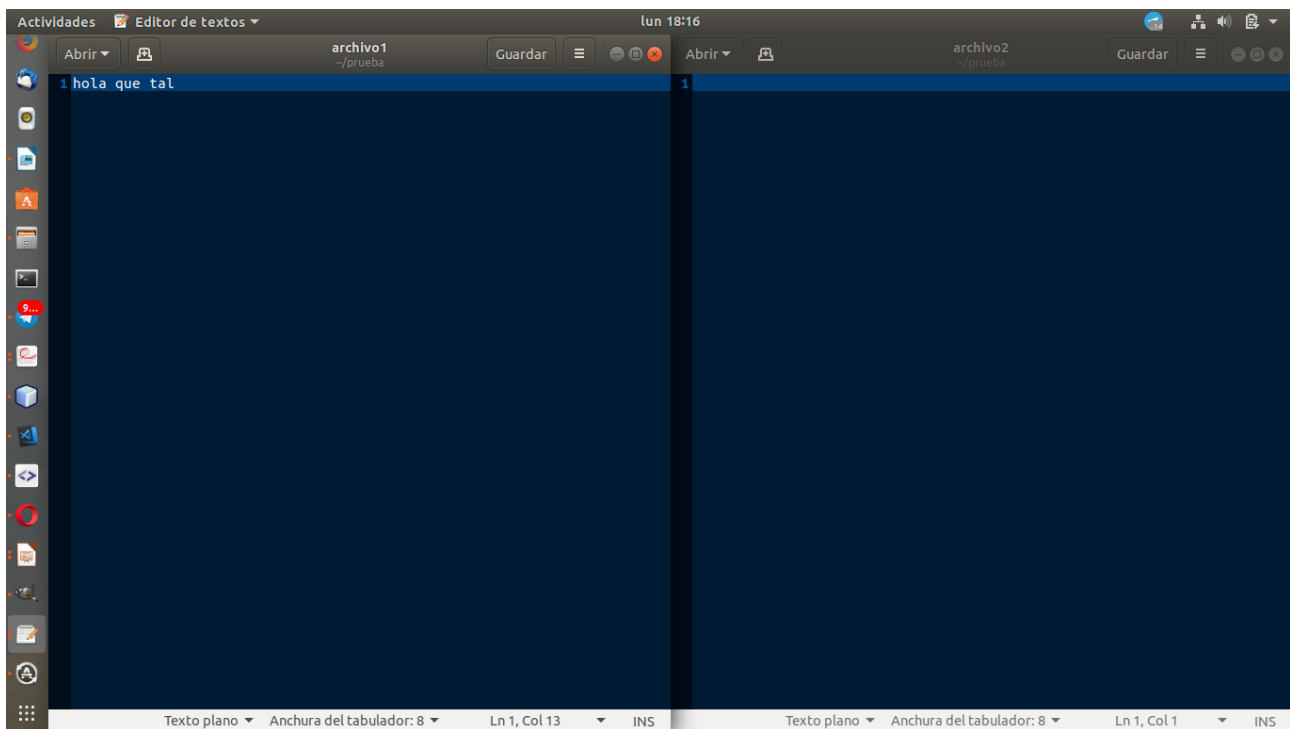
Realiza un programa que añada a un fichero de texto, el contenido de otro fichero de texto.

```
package ejercicios.ficheros;
import java.io.File;
import java.util.Scanner;

/**
 *
 * @author likendero
 */
public class QuintoEjercicio {
    private static File directorio = null;
    private static File directorioDest = null;
    private static Scanner key = new Scanner(System.in);
    /**
     * metodo principal
     * @param args
     */
    public static void main(String[] args) {
        boolean control = true;
        do{
            try{
                // introduccion del usuario
                // salida
                System.out.println("directorio de primer fichero");
```

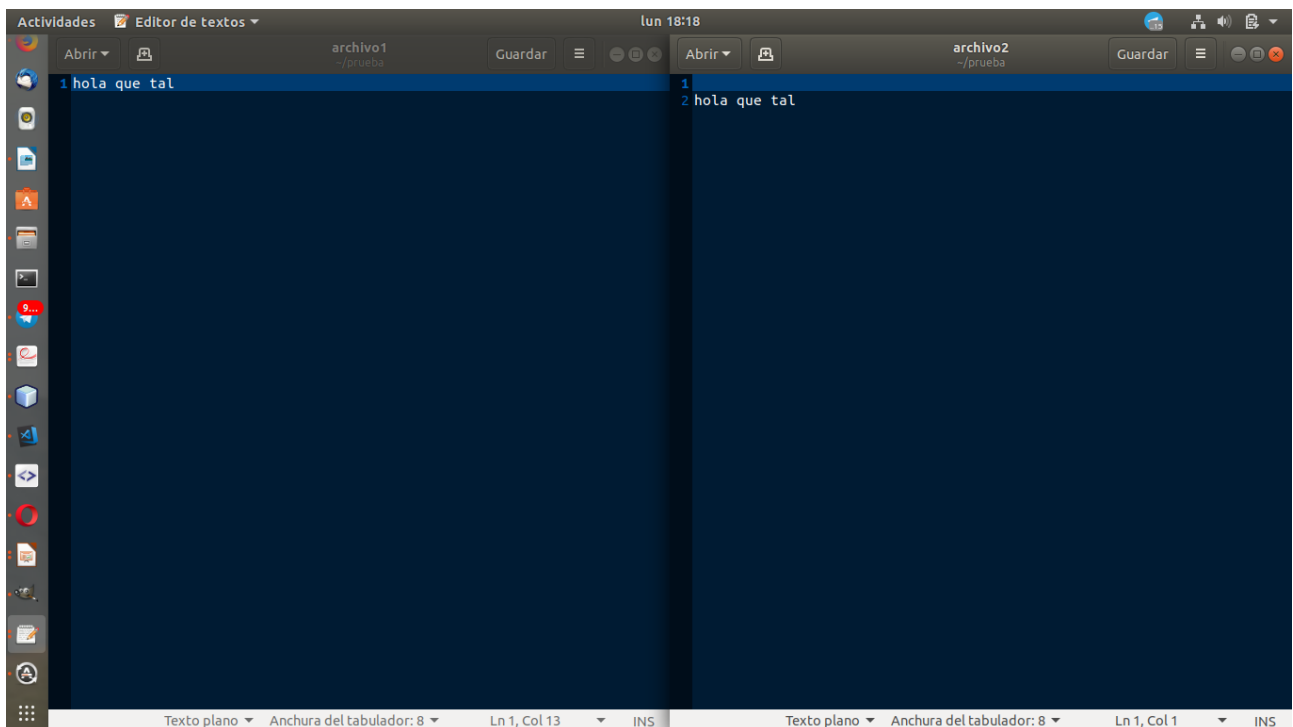
```
    directorio = FicherosTerminal.introFichero(key);
    // destino
    System.out.println("directorio de fichero destino");
    directorioDest = FicherosTerminal.introFichero(key);
    // comprobacion de que los directorios existen
    if(directorio.exists() && directorioDest.exists()){
        // proceso de lectura
        String txt = FicherosTerminal.lectorTxt(directorio);
        // proceso de escritura en el nuevo directorio
        FicherosTerminal.escriptorTxt(directorioDest, txt, true);
        System.out.println("fin del programa");
    }else{
        System.out.println("uno de los directorios no existe");
        control = false;
    }
} catch (NullPointerException nu){
    nu.printStackTrace();
} catch (Exception ex){
    ex.printStackTrace();
}
} while (!control);
}
```

Resultado



Output - acceso (run) x

```
run:
directorio de primer fichero
introduzca el directiro de un fichero:
/home/likendero/prueba/archivo1
directorio de fichero destino
introduzca el directiro de un fichero:
/home/likendero/prueba/archivo2
fin esxcritura
fin del programa
BUILD SUCCESSFUL (total time: 30 seconds)
```



Ejercicio 6

Crea una aplicación donde pidamos la ruta de un fichero por teclado y un texto que queramos a escribir en el fichero. Deberás mostrar por pantalla el mismo texto pero variando entre mayúsculas y minúsculas, es decir, si escribo “Bienvenido” deberá devolver “BIENVENIDO”. Si se escribe cualquier otro carácter, se quedara tal y como se escribió. Deberás crear un método para escribir en el fichero el texto introducido y otro para mostrar el contenido en mayúsculas.

FicherosTerminal

```
/**
 * metodo que escribe una cadena de texto en un fichero
 * @param dir directorio del fichero
 * @param txt cadena de texto
 * @param sobreEscritua indica si se quiere sobre escribir
 * o escribir a continuacion
 */
public static void escritorTxt(File dir,String txt,boolean sobreEscritua){
    try{
        FileWriter escritor = new FileWriter(dir,sobreEscritua);
        BufferedWriter flujoEscr = new BufferedWriter(escritor);
        flujoEscr.write(txt);
        System.out.println("fin esxcritura");
        // cerrado de flujo
        flujoEscr.close();
        escritor.close();
    }catch(IOException io){
        System.out.println("error en la escritura");
        io.printStackTrace();
    }catch(Exception ex){
        System.out.println("error");
        ex.printStackTrace();
    }
}

/**
 * metodo que convierte las letras en minuscula a mayusculas
 * y viceversa
 * @param entrada cadena de texto que se quiere
 * @return
 */
public static String invertir(String entrada){
    // combierto el string en un array de caracteres
    char caracteres[] = entrada.toCharArray();
    String salida = "";
    // recorrido del array de caracteres
    for(int i = 0; i < caracteres.length;i++){
        // comprobacion de si es una letra
        if(Character.isLetter(caracteres[i])){
            // comprobacion de minuscula o mayuscula
            if(Character.isLowerCase(caracteres[i])){
                // se convierte en mayuscula al mismo tiempo que se annade
                // a la cadena
            }
        }
    }
}
```

```
        salida += Character.toUpperCase(caracteres[i]);
    }else{
        // se suma en minuscula
        salida += Character.toLowerCase(caracteres[i]);
    }
    }else{
        // en caso de no ser letra el elemento se annade tal cual
        salida += caracteres[i];
    }
}
return salida;
}
```

ejercicio

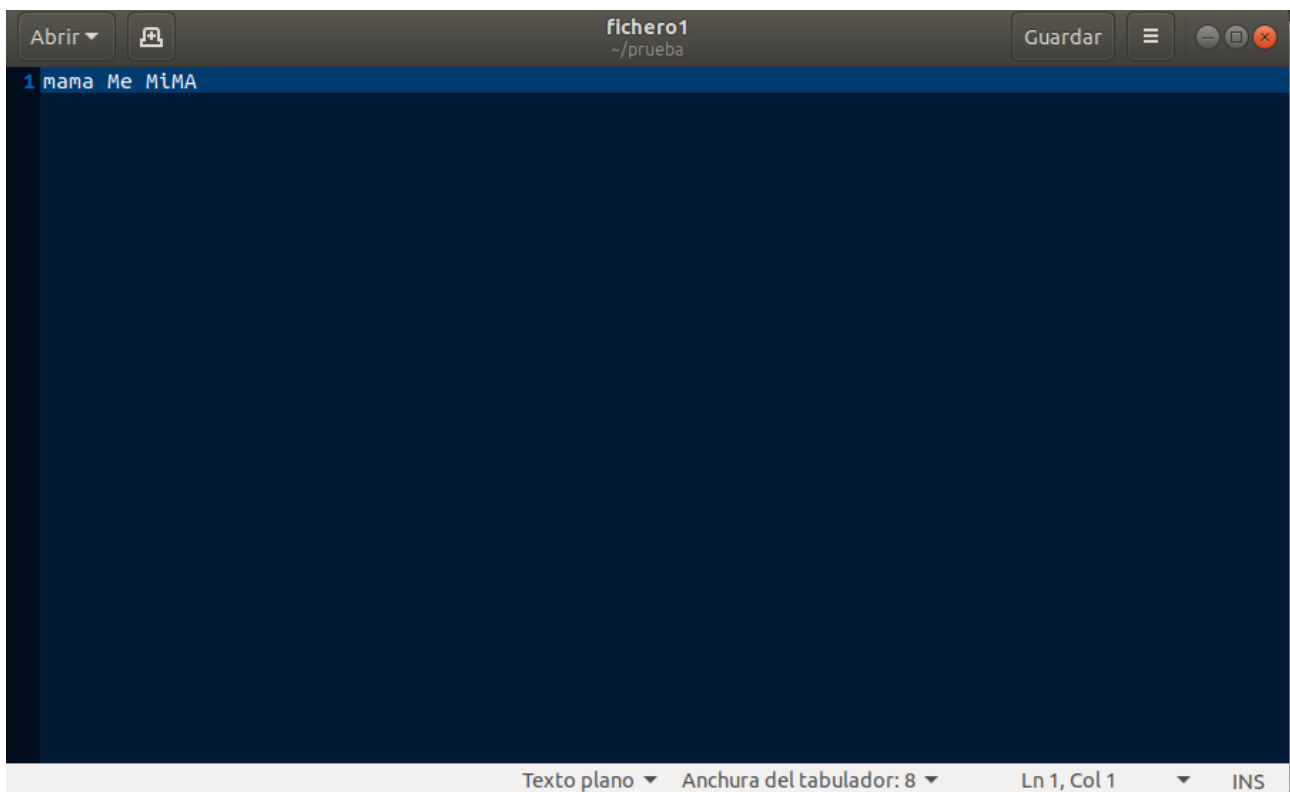
```
package ejercicios.ficheros;
import java.io.File;
import java.util.Scanner;
/**
 *
 * @author likendero
 */
public class SextoEjercicio {
    private static File directorio = null;
    private static Scanner key = new Scanner(System.in);
    /**
     * metodo priuncipal
     * @param args
     */
    public static void main(String[] args) {
        try{
            // introduccion del directorio
            directorio = FicherosTerminal.introFicheroEscritura(key);
            System.out.println("introduzca la frase que desea introducir");
            // guarado de un texto desde teclado
            key.nextLine();
            String frase = key.nextLine();
            // escritura del fichero
            FicherosTerminal.escriptorTxt(directorio, frase, false);
            // salida por pantalla invertido del fichero
            System.out.println(
                FicherosTerminal.invertir(
                    FicherosTerminal.lectorTxt(directorio)
                )
            );
        }catch(NullPointerException nu){
            System.out.println("error");
            nu.printStackTrace();
        }catch(Exception ex){
            System.out.println("error en el programa");
            ex.printStackTrace();
        }
    }
}
```

```
}  
}
```

Output - acceso (run) x

```
run:  
introduzca el directiro de un fichero:  
/home/likendero/prueba/fichero1  
el archivo no existe, desea crearlo?(s/n)  
s  
introduzca la frase que desea introducir  
mama Me MiMA  
fin esxcritura  
  
MAMA mE mIma  
BUILD SUCCESSFUL (total time: 28 seconds)
```

2



The screenshot shows a code editor window with the title 'fichero1' and the path '~/prueba'. The editor contains a single line of text: '1 mama Me MiMA'. The status bar at the bottom indicates 'Texto plano', 'Anchura del tabulador: 8', 'Ln 1, Col 1', and 'INS'.

Ejercicio 7

Crea una aplicación que pida por teclado un número de números aleatorios enteros positivos y la ruta de un fichero. Este fichero contendrá la cantidad de números aleatorios enteros positivos que se ha pedido por teclado. El rango de los números aleatorios estará entre 0 y 100, incluyendo el 100. Cada vez que ejecutemos la aplicación añadiremos números al fichero sin borrar los anteriores, es decir, si cuando creo el fichero añado 10 números y después añado otros 10 al mismo, en el fichero habrá 20 números que serán mostrados por pantalla

```
package ejercicios.ficheros;
import java.io.File;
import java.util.InputMismatchException;
import java.util.Scanner;
/**
 *
 * @author likendero
 */
public class SeptimoEjercicio {
    private static File directorio = null;
    private static Scanner key = new Scanner(System.in);

    public static void main(String[] args) {
        try{
            // se pide al usuario el directorio
            directorio = FicherosTerminal.introFicheroEscritura(key);
            // pregunta al usuario y se crea el array de numeros
            int[] numeros = numerosAleatorios(cantidad());
            // se comprueba si el directorio existe
            if(directorio.exists()){
                /*
                 en el caso que exista ya el fichero annade numeros
                */
                FicherosTerminal.escribirNumeros(directorio, numeros, true);
            }else{
                /*
                 en el caso que no exista lo crea y escribe los numeros
                */
                FicherosTerminal.escribirNumeros(directorio, numeros, false);
            }
            System.out.println("los numeros del fichero son");
            System.out.println(FicherosTerminal.leerFicheroNumeros(directorio));
        }catch(NullPointerException nu){
            System.out.println("error nulo");
            nu.printStackTrace();
        }catch(Exception ex){
            System.out.println("error");
            ex.printStackTrace();
        }
    }
}
/**
 * metodo que genera un array de numeros enteros aleatorio
 * @param cantidad numero de numeros aleatorios deseado
 */
```

```
* @return array con los numeros
*/
public static int[] numerosAleatorios(int cantidad){
    int numeros[] = new int[cantidad];
    // bucle para generar los numeros
    for(int i = 0; i < numeros.length; i++){
        // generacion de numeros aleatorios entre 0 y 100 incluidos
        numeros[i] = (int)(Math.random()*101);
    }
    return numeros;
}
/**
 * metodo que pide al usuario el numero de datos a crear
 * @return
 */
public static int cantidad(){
    // variable de control para el bucle
    boolean control = false;
    // variable entera que almacena el numero a devolver
    int vuelt = 0;
    do{
        try{
            System.out.println("que cantidad de variables enteras desea crear?");
            // introduccion del usuario
            vuelt = key.nextInt();
            // cambio control
            control = true;
            // excepcion que controla si lo introducido es un numero
        }catch(InputMismatchException in){
            System.out.println("se ha introducido un elemento no valido");
            key.nextLine();
        }
    }while(!control);
    return vuelt;
}
}
```

```
Output - acceso (run) x
introduzca el directiro de un fichero:
/home/likendero/prueba/ficheroNumeros
el archivo no existe, desea crearlo?(s/n)
s
que cantidad de variables enteras desea crear?
10
los numeros del fichero son
fin lectura

79
90
21
14
88
0
```

Ejercicio 8:

A partir de un fichero binario de acceso aleatorio de empleados AleatorioEmple.dat y con estructura de registros como la que se muestra a continuación:

Apellido

Dep

Salario

Realiza un programa que introduzca datos en dicho fichero. Además realiza un programa que reciba un identificador desde la línea de comandos y visualice sus datos. Si el empleado no existe debe visualizar un mensaje indicándolo.

```
package ejercicios.ficheros;
import java.io.*;
import java.util.InputMismatchException;
import java.util.Scanner;
/**
 *
 * @author likendero
 */
public class OctavoEjercicio {
    private static File directorio = new File("AleatorioEmple.dat");
    private static Scanner key = new Scanner(System.in);
    /**
```

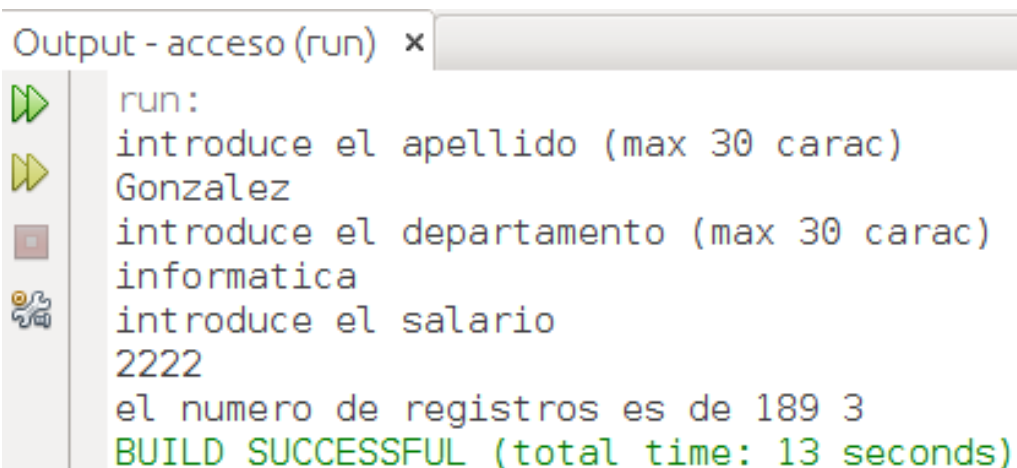
```
* metodo principal
* @param args
*/
public static void main(String[] args) {
    try{
        // introduccion del apellido
        System.out.println("introduce el apellido (max 30 carac)");
        String apellido = cadenas();
        // introduccion del departamento
        System.out.println("introduce el departamento (max 30 carac)");
        String departamento = cadenas();
        // introduccion del salario
        System.out.println("introduce el salario");
        int sala = salario();
        // escritura en el fichero
        escribir(apellido, departamento, sala);
    }catch(Exception ex){
        System.out.println("error");
        ex.printStackTrace();
    }
}
/**
 * metodo que escribe la informacion de un empleado al final
 * de un archivo de acceso aleatorio
 * @param apellido
 * @param departamento
 * @param salario
 */
private static void escribir(String apellido,String departamento,int salario){
    try{
        int id = 1;
        // creacion del flujo de lectura escritura
        RandomAccessFile escritor = new RandomAccessFile(directorio, "rw");
        // guardado del tamanno original
        long tamano = escritor.length();
        // posicionamiento al final del del documeto
        escritor.seek(escritor.length());
        // se comprueba si el fichero tiene registros
        if(tamano != 0){
            // si tiene se posiciona en el anterior
            escritor.seek(tamano-63);
            // se guarda el id mas 1
            id = escritor.readInt() + 1;
            // se mueve el puntero al final del documento
            escritor.seek(tamano);
            // se escribe el id
            escritor.writeInt(id);
        }else{
            // se escribe el primer id
            escritor.writeInt(id);
        }
    }
}
```

```
// annadir apellido
//escritor.writeUTF(cadenaTreinta(apellido));
escritor.writeUTF(apellido);
// escribir departamento
//escritor.writeUTF(cadenaTreinta(departamento));
escritor.writeUTF(departamento);
// escribir salario
escritor.write(salario);
escritor.setLength(tamano+63);
// numero de registros
    System.out.println("el numero de registros es de " + escritor.length() + " " +
(escritor.length()/63));
    // cierre de flujo
    escritor.close();

} catch(IOException io){
    System.out.println("error en la escritura");
}
}
/**
 * metodo que permite introducir una cadena
 * @return
 */
private static String cadenas(){
    boolean control = false;
    String salida = "";
    do{
        try{
            // introduccion del apellido
            salida = key.next();
            // se comprueba el rango de la cadena
            if(salida.length()>30){
                // si es demasiado larga se indica y se vuelve a intro
                System.out.println("es demasiado largo");
            }else{
                // si esta en rango se termian el bucle
                control = true;
            }
        } catch(InputMismatchException in){
            System.out.println("error en la introduccion");
            key.nextLine();
        }
    }while(!control);
    // se devuelve la cadena
    return salida;
}
/**
 * metodo que permite introducir un numero entero
 * @return
 */
private static int salario(){
    int salarioVu = 0;
    boolean control = false;
```

```
do{
    try{
        salarioVu = key.nextInt();
        control = true;
    }catch(InputMismatchException in){
        System.out.println("error en la introduccion");
        key.nextLine();
    }
}while(!control);
return salarioVu;
}
/**
 * metodo que ajustas las cadenas a 30 elementos
 * @param cadenaOriginal
 * @return
 */
private static String cadenaTreinta(String cadenaOriginal){
    // se comprueba si es necesario ajustar la cadena
    if(cadenaOriginal.length()<30){
        // si lo fuese se añaden los espacios necesarios
        for(int i = 0; i < 30 - cadenaOriginal.length(); i++){
            cadenaOriginal += " ";
        }
    }
    return cadenaOriginal;
}
}
```

Resultado



```
Output - acceso (run) x
run:
introduce el apellido (max 30 carac)
Gonzalez
introduce el departamento (max 30 carac)
informatica
introduce el salario
2222
el numero de registros es de 189 3
BUILD SUCCESSFUL (total time: 13 seconds)
```

Ejercicio 8 parte 2

```
package ejercicios.ficheros;
import java.io.*;
import java.util.InputMismatchException;
import java.util.Scanner;

/**
 *
 * @author likendero
 */
public class OctavoEjercicioLectura {
    private static File directorio = new File("AleatorioEmple.dat");
    private static Scanner key = new Scanner(System.in);
    private static final int TAMANNO = 63;
    private static int numRegistros = 0;

    /**
     * metodo principal
     * @param args
     */
    public static void main(String[] args) {

        try{
            System.out.println(leerRegistro(seleccion()));
        }catch(Exception ex){
            System.out.println("error");
            ex.printStackTrace();
        }
    }

    /**
     * metodo que lee un registro indicado por el usuario
     * @return
     */
    private static String leerRegistro(int registro){
        String salida = "";
        try{
            // creacion del flujo
            RandomAccessFile lector = new RandomAccessFile(directorio, "r");
            // al registro se le resta 1 ya que el primero empieza en 0
            registro -= 1;
            // movimiento del puntero
            lector.seek(TAMANNO*registro);
            // lectura
            salida = lector.readInt() + " " + lector.readUTF() + " " + lector.readUTF()
                    + " " + lector.readInt();
        }catch(IOException io){
            System.out.println("error en la lectura");
        }
        return salida;
    }

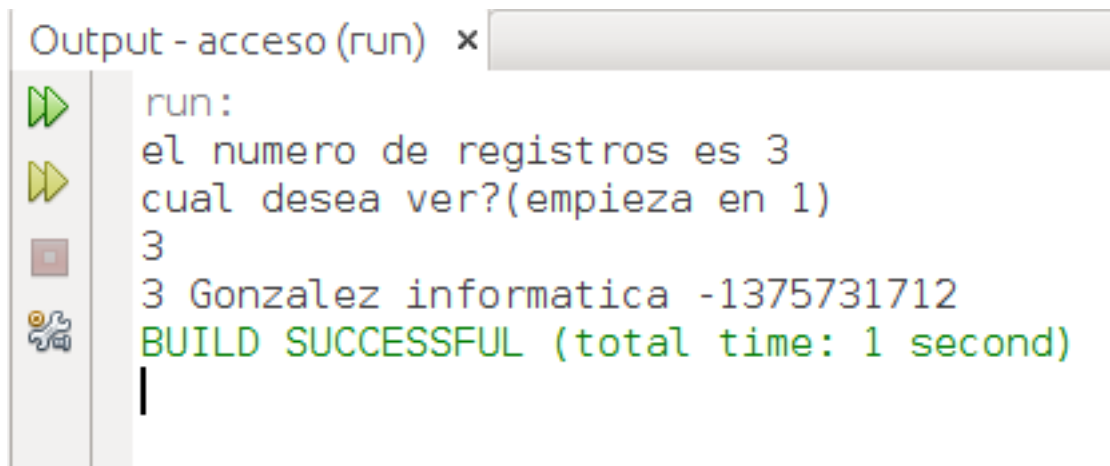
    /**
     * metodo que calcula el numero de registros en el fichero
     * @return
     */
}
```

```
*/
public static int numeroRegistros(){
    int vuelta = 0;
    try{
        // creacion del flujo
        RandomAccessFile lector = new RandomAccessFile(directorio, "r");
        // guardado del numero de registros
        vuelta = (int)lector.length()/TAMANNO;
        lector.close();
    }catch(IOException io){
        System.out.println("error lectura");
        io.printStackTrace();
    }
    return vuelta;
}
/**
 * metodo que pregunta al usuario que desea ver
 * @return
 */
public static int seleccion(){
    // variable para bucle
    boolean control = false;
    int seleccion = 0;
    numRegistros =numeroRegistros();
    do{
        try{
            // mensaje al usuario
            System.out.println("el numero de registros es " + numRegistros);
            System.out.println("cual desea ver?(empieza en 1)");
            // introduccion del usuario
            seleccion = key.nextInt();
            // validacion de la introduccion
            if(seleccion > 0 && seleccion <= numRegistros){
                // fin del bucle
                control = true;
            }else{
                // ensaje indicando el error
                System.out.println("eleccion no valida");
            }
        }catch(InputMismatchException in){
            System.out.println("error en la introduccion");
            key.nextLine();
        }
    }while(!control);
    return seleccion;
}
}
```


Javier González Rives

Resultado

El sueldo no sale correctamente



```
Output - acceso (run) x
run:
el numero de registros es 3
cual desea ver?(empieza en 1)
3
3 Gonzalez informatica -1375731712
BUILD SUCCESSFUL (total time: 1 second)
|
```

Ejercicio 9

A partir del fichero binario de acceso aleatorio de empleados EmpleAleatorio.dat, realiza un programa Java que permita eliminar empleados. Para ello se hará un borrado lógico que consistirá en poner a valor -1 el id del empleado.

```
package ejercicios.ficheros;
import java.io.*;
import java.util.Scanner;
/**
 *
 * @author likendero
 */
public class NovenoEjercicio {
    private static File directorio = new File("AleatorioEmple.dat");
    private static Scanner key = new Scanner(System.in);
    private static final int TAMANNO = 63;

    /**
     * metodo principal
     * @param args
     */
    public static void main(String[] args) {
        try{
            eliminar(OctavoEjercicoLectura.seleccion());
        }catch(Exception ex){
            System.out.println("error");
            ex.printStackTrace();
        }
    }
    /**
     *
     * @param registro
     */
    private static void eliminar(int registro){
```

```
try{
    registro -= 1;
    // creacion de flujo
    RandomAccessFile escritor = new RandomAccessFile(directorio, "rw");
    // se posiciona en el registro
    escritor.seek(TAMANNO*registro);
    // se salta el id
    escritor.readInt();
    // se guarda la informacion del registro
    String apellido = escritor.readUTF();
    String departamento = escritor.readUTF();
    int sueldo = escritor.readInt();
    // se reescribe con el id -1
    escritor.seek(TAMANNO*registro);
    escritor.writeInt(-1);
    escritor.writeUTF(apellido);
    escritor.writeUTF(departamento);
    escritor.writeInt(sueldo);
    // se cierra el flujo
    escritor.close();
}catch(IOException io){
    System.out.println("error de escritura");
    io.printStackTrace();
}
}
```

Resultado

```
run:
el numero de registros es 3
cual desea ver?(empieza en 1)
1
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
Output - acceso (run) x
run:
el numero de registros es 3
cual desea ver?(empieza en 1)
1
-1 javier infor -1207959552
BUILD SUCCESSFUL (total time: 2 seconds)
```