INSTITUTO TÉCNICO
DE ESTUDIOS
PROFESIONALES

PRACTICA 1

Acceso a datos

Práctica 1.1

Objetivos

Desarrollar una aplicación que almacene datos en un fichero

Enunciado

Se debe implementar una aplicación que gestione información almacenándola y recuperándola de fichero. Se pensará en un supuesto real en el que exista un tipo de objeto y se creará la aplicación para gestionarlo.

Requisitos

- Se debe gestionar información de un tipo de objeto con, al menos, 5 atributos. Deben aparecer, al menos, datos de tipo cadena, número (entero y decimal) e imágenes.
- De cada objeto el usuario podrá dar de alta, modificar y eliminar
- La aplicación será capaz de almacenar toda la información en disco en una ubicación fija de forma transparente para el usuario. La carga de los datos se realizará durante la carga de la aplicación
- Se deberá comentar el código
- La aplicación contará con un listado para cada tipo de objeto de forma que el usuario pueda acceder a cualquiera de ellos directamente para ver su información

Otras funcionalidades

- La aplicación contará con una opción de búsqueda avanzada desde la que se podrá buscar algún objeto
- La aplicación dispondrá de una opción de *guardar como* que permitirá almacenar los datos en una ubicación alternativa
- Añadir la opción de que sea el usuario quién decida cuando realizar el guardado de los datos, en lugar de hacerlo de forma transparente
- Permitir al usuario cambiar, desde la aplicación, la ruta fija donde se almacenan los ficheros
- Añadir una opción a la aplicación que permita eliminar todos los datos del programa
- Activar/Desactivar los controles del interfaz de forma que no se permita al usuario utilizar aquellos que no deba usar en cada momento
- Añadir una opción al usuario que permita recuperar el último elemento borrado

```
public class Alumno {
	// Atributos:
	String nombre;
	Integer posicion;
	Float notaMedia;
	String foto;
	String dni;

	// Constructores:
	public Alumno() {
		this.nombre = "Sin nombre";
		this.posicion = null;
		this.notaMedia = null;
		this.foto = null;
		this.dni = "Sin DNI";
	}// Fin Constructor

	public Alumno(String nombre, Float notaMedia, String dni) {
		this.nombre = nombre;
		this.posicion = null;
		this.notaMedia = notaMedia;
		this.foto = null;
		this.dni = dni;
	}// Fin Constructor

	public Alumno(String nombre, Integer posicion, Float notaMedia, String dni, String
foto) {
		this.nombre = nombre;
		this.posicion = posicion;
		this.notaMedia = notaMedia;
		this.foto = foto;
		this.dni = dni;
	}// Fin Constructor

	// Métodos:
	public String getNombre() {
		return nombre;
	}// Fin Función

	public void setNombre(String nombre) {
		this.nombre = nombre;
	}// Fin Procedimiento

	public Integer getPosicion() {
		return posicion;
	}// Fin Función

	public void setPosicion(Integer posicion) {
		this.posicion = posicion;
	}// Fin Procedimiento

	public Float getNotaMedia() {
		return notaMedia;
	}// Fin Función

	public void setNotaMedia(Float notaMedia) {
		this.notaMedia = notaMedia;
	}// Fin Procedimiento
```

```
	public String getFoto() {
		return this.foto;
	}// Fin Función

	public void setFoto(String foto) {
		this.foto = foto;
	}// Fin Procedimiento

	public String getDni() {
		return dni;
	}// Fin Función

	public void setDni(String dni) {
		this.dni = dni;
	}// Fin Procedimiento

}
```

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;

public class GestionaAlumnos {
	// Atributos:
	File ruta, borrados;

	// Constructores:
	public GestionaAlumnos() {
		this.ruta = new File("C:\\Users\\" + System.getProperty("user.name") +
"\\Documents\\Alumnos");
		this.ruta.mkdir();
		this.borrados = new File("C:\\Users\\" + System.getProperty("user.name") +
"\\Documents\\AlumnosBorrados");
		this.borrados.mkdir();
	}// Fin Constructor

	public GestionaAlumnos(String ruta) {
		this.ruta = new File(ruta);
		this.ruta.mkdir();
		this.borrados = new File(this.ruta + ".\\Borrados");
		this.ruta.mkdir();
	}// Fin Constructor

	// Métodos:
	public String getRuta() {
		return this.ruta.getAbsolutePath();
	}// Fin Función

	public void setRuta(String ruta) {
		this.ruta = new File(ruta);
```

```
            this.ruta.mkdir();
            this.borrados = new File(this.ruta + ".\\Borrados");
            this.ruta.mkdir();
      }// Fin Procedimiento

      public boolean crearAlumno(Alumno alu) {
            // Entorno:
            boolean retornar;
            FileWriter ficheroAGenerar;
            String datosAGuardar;

            // Algoritmo:
            retornar = true;

            try {
                  ficheroAGenerar = new FileWriter(this.ruta + "\\" + alu.dni + ".txt");
                  datosAGuardar = alu.getNombre() + ":" + alu.getPosicion() + ":" +
alu.getNotaMedia() + ":" + alu.getDni()
                              + ":" + alu.getFoto();
                  ficheroAGenerar.write(datosAGuardar);
                  ficheroAGenerar.close();
            } catch (IOException e) {
                  retornar = false;
            } // Fin try

            return retornar;
      }// Fin Función

      public boolean modificarAlumno(Alumno aluModificado) {
            // Entorno:
            // Algoritmo:
            return crearAlumno(aluModificado);
      }// Fin Función

      public boolean eliminarAlumno(Alumno alu) {
            // Entorno:
            File borrarEsto;

            // Algoritmo:
            borrarEsto = new File(this.ruta + "\\" + alu.dni + ".txt");

            backup(borrarEsto.toString(), this.borrados + "\\ultElem.txt");

            return borrarEsto.delete();
      }// Fin Función

      private boolean backup(String aCopiar, String aGenerar) {
            File origen = new File(aCopiar);
            File destino = new File(aGenerar);
            boolean salida = true;

            if (origen.exists()) {
                  try {
                        InputStream in = new FileInputStream(origen);
                        OutputStream out = new FileOutputStream(destino);
                        byte[] buf = new byte[1024];
                        int len;
                        while ((len = in.read(buf)) > 0) {
```

```
                                    out.write(buf, 0, len);
                              } // Fin Mientras
                              in.close();
                              out.close();
                              salida = true;
                        } catch (IOException ioe) {
                              ioe.printStackTrace();
                              salida = false;
                        } // Fin try
                  } else {
                        salida = false;
                  } // Fin Si

                  return salida;
            }// Fin Función

            public String listadoAlumnos() {
//                public Alumno[] listadoAlumnos() {
                  // Entorno:
                  String[] dnis, datos;
                  BufferedReader lecturaActual;
//                Alumno[] salida;
                  String salida;

                  // Algoritmo:
                  dnis = this.ruta.list();
//                salida = new Alumno[dnis.length];
                  salida = "";
                  for (int i = 0; i < dnis.length; i++) {
                        try {
                              lecturaActual = new BufferedReader(new
FileReader(ruta.getAbsolutePath() + "\\" + dnis[i]));
                              datos = lecturaActual.readLine().split(":");

                              salida += "Nombre: " + datos[0] + "\n";
                              salida += "Posición: " + datos[1] + "\n";
                              salida += "Nota media: " + datos[2] + "\n";
                              salida += "DNI: " + datos[3] + "\n";
                              salida += "Imagen: " + datos[4] + "\n" + "\n";

//                              salida[i] = new Alumno(datos[0], Integer.parseInt(datos[1]),
Float.parseFloat(datos[2]), datos[3], (Imagen)datos[4]); //Arreglar lo de imagen
                        } catch (FileNotFoundException e) {
                              System.out.println("No pude ver al alumno con dni: " + dnis[i]);
                        } catch (IOException e) {
                              System.err.println("Error E/S");
                        } // Fin try
                  } // Fin Para

                  return salida;
            }// Fin Función

            public Alumno busquedaAvanzada(String dni) {
                  // Entorno:
                  String[] dnis;
                  BufferedReader alumnoLeido;
                  Alumno devuelta;
                  byte i;
```

```
            // Algoritmo:
            devuelta = null;
            dnis = this.ruta.list();

            i = 0;
            while (!dnis[i].equalsIgnoreCase(dni + ".txt") && i < dnis.length) {
                  i++;
            } // Fin Mientras

            try {
                  alumnoLeido = new BufferedReader(new
FileReader(this.ruta.getAbsolutePath() + "\\" + dnis[i]));
                  dnis = alumnoLeido.readLine().split(":");

                  devuelta = new Alumno(dnis[0], Integer.parseInt(dnis[1]),
Float.parseFloat(dnis[2]), dnis[3], dnis[4]);
            } catch (FileNotFoundException e) {
                  System.out.println("No se ha encontrado el Alumno con dni: " + dni);
            } catch (IOException e) {
                  System.out.println("Error E/S");
            } // Fin Si

            return devuelta;
      }// Fin Función

      public boolean guardarDatosActuales() {
            return true;
      }// Fin Función

      public boolean borrarTodosLosDatos() {
            // Entorno:
            // Algoritmo:
            String[] archivos = this.ruta.list();
            for (int i = 0; i < archivos.length; i++) {
                  File aDeletear = this.ruta;
                  aDeletear = new File(aDeletear + "\\" + archivos[i]);
                  aDeletear.delete();
            } // Fin Para

            archivos = this.borrados.list();
            for (int i = 0; i < archivos.length; i++) {
                  File aDeletear = this.ruta;
                  aDeletear = new File(aDeletear + "\\" + archivos[i]);
                  aDeletear.delete();
            } // Fin Para

            return this.ruta.delete();
      }// Fin Función

      public Alumno recuperarUltimoElementoBorrado() {
            // Entorno:
            Alumno devolucion;
            BufferedReader fichero;
            String[] archivo;

            // Algoritmo:
            devolucion = null;
```

```
            try {
                  fichero = new BufferedReader(new FileReader(this.borrados +
"\\ultElem.txt"));

                  archivo = fichero.readLine().split(":");
                  devolucion = new Alumno(archivo[0], Integer.parseInt(archivo[1]),
Float.parseFloat(archivo[2]), archivo[3],
                              archivo[4]);
            } catch (FileNotFoundException e) {
                  System.out.println("No se pudo econtrar el último elemento");
            } catch (IOException e) {
                  System.err.println("Error E/S");
            } // Fin try

            return devolucion;
      }// Fin Función
}
```

```
import java.util.Scanner;

public class Ejecutable {

      public static void menuPrincipal() {
            System.out.println("1.-  Crear Alumno.");
            System.out.println("2.-  Modificar Alumno.");
            System.out.println("3.-  Eliminar Alumno.");
            System.out.println("4.-  Mostrar lista de Alumnos.");
            System.out.println("5.-  Búsqueda Avanzada.");
            System.out.println("6.-  Guardar datos.");
            System.out.println("7.-  Borrar todos los datos.");
            System.out.println("8.-  Recuperar el último elemento borrado.");
            System.out.println("9.-  Cambiar ruta.");
            System.out.println("10.-  Salir.");
            System.out.print("Elija una opción: ");
      }// Fin Procedimiento

      @SuppressWarnings("resource")
      public static void main(String[] args) {
            // Entorno:
            Scanner sc = new Scanner(System.in);
            GestionaAlumnos principal = new GestionaAlumnos();
            int opcion = 0;
            boolean esSalir = false;
            Alumno alu;
            String datosAlumno = "";

            // Algoritmo:
            do {
                  menuPrincipal();

                  try {
                        opcion = Integer.parseInt(sc.next());
                  } catch (NumberFormatException nfe) {
                        opcion = 0;
                  } // Fin try

                  switch (opcion) {
                  case 1: {
```

```
                        alu = new Alumno();
                        int posicion = 0;
                        float media = -1;

                        do {
                                System.out.print("Introduzca el nombre del alumno: ");
                                datosAlumno = sc.next();
                        } while (datosAlumno.isEmpty());
                        alu.setNombre(datosAlumno);

                        do {
                                System.out.print("Introduzca la posición del alumno >0:
");
                                try {
                                        posicion = Integer.parseInt(sc.next());
                                } catch (NumberFormatException nfe) {
                                        System.out.println("No ha introducido un número
correcto.");
                                        posicion = 0;
                                } // Fin try
                        } while (posicion < 0);
                        alu.setPosicion(posicion);

                        do {
                                System.out.print("Introduzca la nota media del alumno >=0:
");
                                try {
                                        media = Float.parseFloat(sc.next());
                                } catch (NumberFormatException nfe) {
                                        System.out.println("No ha introducido un formato
numérico correcto.");
                                        media = -1;
                                } // Fin try
                        } while (media < 0);
                        alu.setNotaMedia(media);

                        datosAlumno = "";
                        do {
                                System.out.print("Introduzca la ruta completa de la foto:
");
                                datosAlumno = sc.next();
                        } while (datosAlumno.isEmpty());
                        alu.setFoto(datosAlumno);

                        datosAlumno = "";
                        do {
                                System.out.print("Introduzca el DNI del alumno: ");
                                datosAlumno = sc.next();
                        } while (datosAlumno.isEmpty());
                        alu.setDni(datosAlumno);

                        principal.crearAlumno(alu);

                        break;
                } // Crear Alumno
                case 2: {
                        datosAlumno = "";
                        do {
```

```
                                System.out.print("¿Qué alumno desea modificar? (DNI): ");
                                datosAlumno = sc.next();
                        } while (datosAlumno.isEmpty());

                        String dniOriginal = datosAlumno;
                        alu = principal.busquedaAvanzada(datosAlumno);
                        do {
                                System.out.print("¿Desea modificarle el nombre? (S/N): ");
                                datosAlumno = sc.next();
                        } while (!datosAlumno.equalsIgnoreCase("s") &&
!datosAlumno.equalsIgnoreCase("n"));
                        if (datosAlumno.equalsIgnoreCase("s")) {
                                datosAlumno = "";
                                do {
                                        System.out.print("¿Qué nombre desea ponerle?: ");
                                        datosAlumno = sc.next();
                                } while (datosAlumno.isEmpty());

                                alu.setNombre(datosAlumno);
                        } // Fin Si

                        do {
                                System.out.print("¿Desea modificarle la posición? (S/N):
");
                                datosAlumno = sc.next();
                        } while (!datosAlumno.equalsIgnoreCase("s") &&
!datosAlumno.equalsIgnoreCase("n"));
                        if (datosAlumno.equalsIgnoreCase("s")) {
                                datosAlumno = "";
                                int posicion = 0;

                                do {
                                        System.out.print("¿Qué posición desea ponerle? >0:
");
                                        datosAlumno = sc.next();
                                        try {
                                                posicion = Integer.parseInt(datosAlumno);
                                        } catch (NumberFormatException nfe) {
                                                System.out.println("No ha introducido un
número válido");
                                                posicion = 0;
                                        } // Fin try
                                } while (posicion <= 0);

                                alu.setPosicion(posicion);
                        } // Fin Si

                        do {
                                System.out.print("¿Desea modificarle la nota media? (S/N):
");
                                datosAlumno = sc.next();
                        } while (!datosAlumno.equalsIgnoreCase("s") &&
!datosAlumno.equalsIgnoreCase("n"));
                        if (datosAlumno.equalsIgnoreCase("s")) {
                                datosAlumno = "";
                                float notaMedia = -1;

                                do {
```

```
                                        System.out.print("¿Qué nota media desea ponerle?
>=0: ");
                                        datosAlumno = sc.next();
                                        try {
                                              notaMedia = Float.parseFloat(datosAlumno);
                                        } catch (NumberFormatException nfe) {
                                              System.out.println("No ha introducido un
número válido");
                                              notaMedia = -1;
                                        } // Fin try
                                  } while (notaMedia < 0);

                                  alu.setNotaMedia(notaMedia);
                            } // Fin Si

                            do {
                                  System.out.print("¿Desea modificarle la foto? (S/N): ");
                                  datosAlumno = sc.next();
                            } while (!datosAlumno.equalsIgnoreCase("s") &&
!datosAlumno.equalsIgnoreCase("n"));
                            if (datosAlumno.equalsIgnoreCase("s")) {
                                  datosAlumno = "";

                                  do {
                                        System.out.print("¿Qué foto desea ponerle? >=0: ");
                                        datosAlumno = sc.next();
                                  } while (datosAlumno.isEmpty());

                                  alu.setFoto(datosAlumno);
                            } // Fin Si

                            do {
                                  System.out.print("¿Desea modificarle el DNI? (S/N): ");
                                  datosAlumno = sc.next();
                            } while (!datosAlumno.equalsIgnoreCase("s") &&
!datosAlumno.equalsIgnoreCase("n"));
                            if (datosAlumno.equalsIgnoreCase("s")) {
                                  datosAlumno = "";

                                  do {
                                        System.out.print("¿Qué DNI desea ponerle? >=0: ");
                                        datosAlumno = sc.next();
                                  } while (datosAlumno.isEmpty());

                                  alu.setDni(datosAlumno);
                            } // Fin Si


principal.eliminarAlumno(principal.busquedaAvanzada(dniOriginal));
                            principal.crearAlumno(alu);
                            break;
                      } // Modificar Alumno
                      case 3: {
                            datosAlumno = "";
                            do {
                                  System.out.print("¿A qué alumno desea elimiar (dni): ");
                                  datosAlumno = sc.next();
                            } while (datosAlumno.isEmpty());
```

```
                        alu = principal.busquedaAvanzada(datosAlumno);
                        principal.eliminarAlumno(alu);

                        break;
                } // Eliminar Alumno
                case 4: {
                        System.out.println(principal.listadoAlumnos());
                        break;
                } // Mostrar lista de Alumnos
                case 5: {
                        datosAlumno = "";
                        do {
                                System.out.print("Introduzca el DNI de quien desea
conseguir: ");
                                datosAlumno = sc.next();
                        } while (datosAlumno.isEmpty());
                        alu = principal.busquedaAvanzada(datosAlumno);

                        System.out.println("Nombre: " + alu.getNombre() + "\n");
                        System.out.println("Posición: " + alu.getPosicion() + "\n");
                        System.out.println("Nota Media: " + alu.getNombre() + "\n");
                        System.out.println("DNI: " + alu.getDni() + "\n");
                        System.out.println("Imagen: " + alu.getFoto() + "\n");
                        break;
                } // Búsqueda Avanzada
                case 6: {
                        if (principal.guardarDatosActuales()) {
                                System.out.println("Guardados existosamente");
                        } else {
                                System.out.println("No se han guardado los datos");
                        } // Fin Si
                        break;
                } // Guardar datos
                case 7: {
                        principal.borrarTodosLosDatos();
                        break;
                } // Borrar todos los datos
                case 8: {
                        alu = principal.recuperarUltimoElementoBorrado();
                        principal.crearAlumno(alu);
                        break;
                } // Recuperar el último elemento borrado
                case 9: {
                        datosAlumno = "";
                        do {
                                System.out.print("Introduzca la nueva ruta: ");
                                datosAlumno = sc.next();
                        } while (datosAlumno.isEmpty());
                        principal.setRuta(datosAlumno);
                        break;
                } // Salir
                case 10: {
                        esSalir = true;
                        break;
                } // Salir
                default:
                        System.out.println("No ha introducido un número válido.");
```

```
                    }// Fin Según Sea
            } while (opcion > 0 && opcion < 9 || esSalir == false);

        }// Fin Programa
}
```