

# Dante Sarotti - Gestión del sistema de archivos

Usando las clases File, FileReader y FileWriter de Java, crea una clase que, dada una carpeta (establecida en el código), permita:

- Listar los archivos y carpetas.
- Ver los permisos del fichero.
- Leer el fichero.
- Escribir en el fichero.
- Crear un fichero.
- Borrar el fichero.
- Crear un directorio.
- Borrar un directorio.
- Dar la ruta absoluta al fichero.

Crea un pdf mostrando el resultado de las operaciones, las clases/funciones utilizadas y el menú de selección.

# Índice

<b>1 - Clase Ficheros.....</b>	<b>3</b>
<b>2 - Programa principal.....</b>	<b>8</b>
<b>3 - Demostración.....</b>	<b>11</b>
Menú.....	11
Opción 1: Listar archivos y carpetas.....	12
Opción 2: Ver permisos del fichero.....	12
Opción 3: Leer el fichero.....	13
Opción 4: Escribir en el fichero.....	13
Opción 5: Crear un fichero.....	14
Opción 6: Borrar un fichero.....	14
Opción 7: Crear un directorio.....	15
Opción 8: Borrar un directorio.....	15
Opción 9: Mostrar ruta absoluta de un fichero.....	16

# 1 - Clase Ficheros

Herramientas para llevar a cabo el ejercicio.

```
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Scanner;

public class Ficheros {

    /**
     * Muestra el contenido del directorio indicado.
     * @param rutaCarpeta El directorio que listar.
     */
    public static void listarArchivosCarpetas(String rutaCarpeta) {
        File carpeta = new File(rutaCarpeta);
        File[] archivos = carpeta.listFiles();
        System.out.println("Listando archivos y carpetas en " +
carpeta.getAbsolutePath() + ":");
        for (File archivo : archivos) {
            System.out.println(archivo.getName());
        }
    }

    /**
     * Muestra los permisos del fichero.
     * @param nombrefichero El fichero del cual mostrar permisos.
     */
    public static void verPermisosFichero(String rutaCarpeta, String nombrefichero) {
        File fichero = new File(rutaCarpeta + nombrefichero);
        if (fichero.exists()) {
            System.out.println("Permisos del fichero " + fichero.getAbsolutePath() +
":");

            System.out.println("Lectura: " + (fichero.canRead()?"Si":"No"));
            System.out.println("Escritura: " + (fichero.canWrite()?"Si":"No"));
            System.out.println("Ejecucion: " + (fichero.canExecute()?"Si":"No"));
        }else{
            System.out.println("No se encuentra el fichero o directorio");
        }
    }
}
```

```

/**
 * Muestra el contenido del fichero.
 * @param nombreFichero El fichero a leer.
 */
public static void leerFichero(String rutaCarpeta, String nombreFichero) {
    File fichero = new File(rutaCarpeta + nombreFichero);
    try (FileReader fr = new FileReader(fichero)) {
        int c;
        System.out.println("Contenido del fichero " + fichero.getAbsolutePath() +
":");

        while ((c = fr.read()) != -1) {
            System.out.print((char) c);
        }
    } catch (IOException e) {
        System.out.println("Error al leer el fichero.");
    }
}

/**
 * Solicita una entrada por teclado y la guarda al final del fichero.
 * @param nombreFichero El fichero a editar.
 */
public static void escribirEnFichero(String rutaCarpeta, String nombreFichero) {
    File archivo = new File(rutaCarpeta + nombreFichero);
    if (!archivo.exists() || archivo.isFile()) {
        try (FileWriter fw = new FileWriter(archivo, true)) {
            Scanner scanner = new Scanner(System.in);
            System.out.println("Introduce el texto a escribir (termina con Ctrl+D
en Linux/Unix o Ctrl+Z en Windows):");
            while (scanner.hasNextLine()) {
                String line = scanner.nextLine();
                fw.write(System.lineSeparator());
                fw.write(line);
            }
            fw.close();
            System.out.println("Contenido escrito en el fichero.");
        } catch (IOException e) {
            System.out.println("Error al escribir en el fichero.");
        }
    } else {
        System.out.println("La ruta no pertenece a un archivo.");
    }
}

/**
 * Crea un fichero nuevo.
 * @param nombreFichero el nombre del fichero a crear.
 */

```

```

    public static void crearFichero(String rutaCarpeta, String nombreFichero) {
        if (nombreFichero.isBlank()){
            System.out.println("El nombre no puede estar vacío. No se creará el
fichero.");
        }else{
            File nuevoFichero = new File(rutaCarpeta+nombreFichero);
            try {
                if (nuevoFichero.createNewFile()) {
                    System.out.println("Archivo creado con éxito.");
                } else {
                    System.out.println(nuevoFichero.getAbsolutePath());
                    System.out.println("El archivo ya existe.");
                }
            } catch (IOException e) {
                System.out.println("Error al crear el archivo.");
            }
        }
    }

/**
 * Borra un fichero.
 * @param nombreFichero nombre del fichero a borrar.
 */
    public static void borrarFichero(String rutaCarpeta, String nombreFichero) {
        if (nombreFichero.isBlank()){
            System.out.println("El nombre no puede estar vacío.");
        }else{
            File archivoBorrar = new File(rutaCarpeta+nombreFichero);
            if (archivoBorrar.exists() && archivoBorrar.isFile()) {
                if (archivoBorrar.delete()) {
                    System.out.println("Archivo borrado exitosamente.");
                } else {
                    System.out.println("No se pudo borrar el archivo.");
                }
            } else {
                System.out.println("El archivo no existe.");
            }
        }
    }

/**
 * Crea un nuevo directorio.
 * @param directorio nombre del directorio a crear.
 */
    public static void crearDirectorio(String rutaCarpeta, String directorio) {
        if (directorio.isBlank()){

```

```

        System.out.println("El nombre no puede estar vacío. No se creará el
directorio.");
    }else{
        File nuevoDirectorio = new File(rutaCarpeta+directorio);
        try {
            if (nuevoDirectorio.mkdir()) {
                System.out.println("Directorio creado exitosamente.");
            } else {
                System.out.println("Error al crear el directorio.");
            }
        } catch (SecurityException e) {
            System.out.println("No se ha permitido crear el directorio.");
        }
    }
}

/**
 * Borra un directorio.
 * @param directorio nombre del directorio a borrar.
 */
public static void borrarDirectorio(String rutaCarpeta, String directorio) {
    if (directorio.isBlank()){
        System.out.println("El nombre no puede estar vacío.");
    }else{
        File directorioBorrar = new File(rutaCarpeta + directorio);
        if (directorioBorrar.exists() && directorioBorrar.isDirectory()) {
            try{
                if (directorioBorrar.delete()) {
                    System.out.println("Directorio borrado exitosamente.");
                } else {
                    System.out.println("Error al borrar el directorio.");
                }
            }catch(SecurityException e){
                System.out.println("No se ha permitido borrar el directorio");
            }
        } else {
            System.out.println("El directorio no existe.");
        }
    }
}

/** Muestra la ruta absoluta de un fichero.
 * @param fichero Nombre del fichero del cual mostrar la ruta absoluta.
 */
public static void mostrarRutaAbsolutaFichero(String rutaCarpeta, String fichero){
    if (fichero.isBlank()){
        System.out.println("El nombre no puede estar vacío.");
    }else{

```

```
File ficheroRuta = new File(rutaCarpeta + fichero);
if(ficheroRuta.exists() && ficheroRuta.isFile()){
    System.out.println(ficheroRuta.getAbsolutePath());
}else{
    System.out.println("No existe un fichero con el nombre \"" + fichero +
"\"");
}
}
}
}
```

## 2 - Programa principal.

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

/**
 * Muestra un menú con opciones para gestionar el directorio establecido en @param RUTA_CARPETA
 *
 * @param RUTA_CARPETA Constante de clase que indica el directorio sobre el que trabajar.
 * @author Dante Sarotti
 * @version 0.1
 */
public class App {

    //Debe ser una ruta absoluta o relativa terminada en "/"
    private static final String RUTA_CARPETA = "carpetaFicheros/";

    public static void main(String[] args) {
        File folder = new File(RUTA_CARPETA);
        if (!folder.exists() || !folder.isDirectory()) {
            System.out.println("La carpeta establecida en el código no se encuentra.");
            return;
        }

        int opcion = 0;
        while (opcion != 10) {
            MostrarMenu();
            opcion = solicitarOpcion();

            switch (opcion) {
                case 1:
                    Ficheros.listarArchivosCarpetas(RUTA_CARPETA);
                    break;
                case 2:
                    Ficheros.verPermisosFichero(RUTA_CARPETA, solicitarEntrada("Introduce el nombre del
fichero: "));
                    break;
                case 3:
                    Ficheros.leerFichero(RUTA_CARPETA, solicitarEntrada("Introduce el nombre del
fichero a leer: "));
                    break;
                case 4:
                    Ficheros.escribirEnFichero(RUTA_CARPETA, solicitarEntrada("Introduce el nombre del
fichero a escribir: "));
                    break;
                case 5:
                    Ficheros.crearFichero(RUTA_CARPETA, solicitarEntrada("Introduce el nombre del
fichero a crear: "));
                    break;
                case 6:
                    Ficheros.borrarFichero(RUTA_CARPETA, solicitarEntrada("Introduce el nombre del
fichero a borrar: "));
            }
        }
    }
}
```



```

        break;
    case 7:
        Ficheros.crearDirectorio(RUTA_CARPETA, solicitarEntrada("Introduce el nombre del
directorio a crear: "));
        break;
    case 8:
        Ficheros.borrarDirectorio(RUTA_CARPETA, solicitarEntrada("Introduce el nombre del
directorio a borrar: "));
        break;
    case 9:
        Ficheros.mostrarRutaAbsolutaFichero(RUTA_CARPETA, solicitarEntrada("Introduce el
nombre del fichero para mostrar su ruta absoluta: "));
        break;
    case 10:
        break;
    default:
        System.out.println("Opción inválida.");
    }
}

/**
 * Muestra un menú con las opciones disponibles.
 */
private static void MostrarMenu() {
    System.out.println("\nSelecciona una opción:");
    System.out.println("1. Listar archivos y carpetas.");
    System.out.println("2. Ver permisos del fichero.");
    System.out.println("3. Leer el fichero.");
    System.out.println("4. Escribir en el fichero.");
    System.out.println("5. Crear un fichero.");
    System.out.println("6. Borrar el fichero.");
    System.out.println("7. Crear un directorio.");
    System.out.println("8. Borrar un directorio.");
    System.out.println("9. Mostrar ruta absoluta de un fichero.");
    System.out.println("10. Salir.");
}

/**
 * Solicita que el usuario introduzca un numero para el menú.
 *
 * @return El número que el usuario ha escogido, -1 si no ha introducido un entero.
 */
private static int solicitarOpcion() {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Ingresa el número de la opción: ");
    try{
        return scanner.nextInt();
    }catch(InputMismatchException e){
        return -1;
    }
}

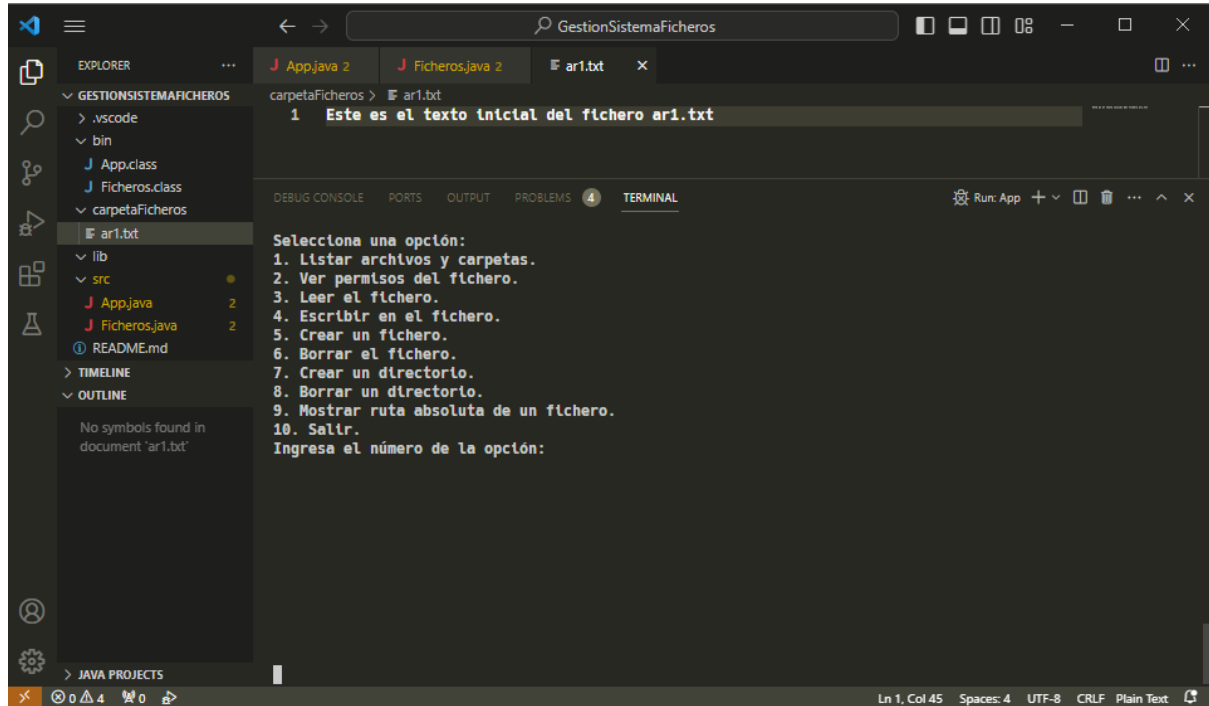
```

```
/**
 * Solicita que el usuario intrduzca una entrada para la opción escogida.
 *
 * @param mensaje El mensaje que indica al usuario qué debe introducir.
 * @return El String que ha introducido el usuario.
 */
private static String solicitarEntrada(String mensaje) {
    Scanner scanner = new Scanner(System.in);
    System.out.print(mensaje);
    return scanner.nextLine();
}
}
```

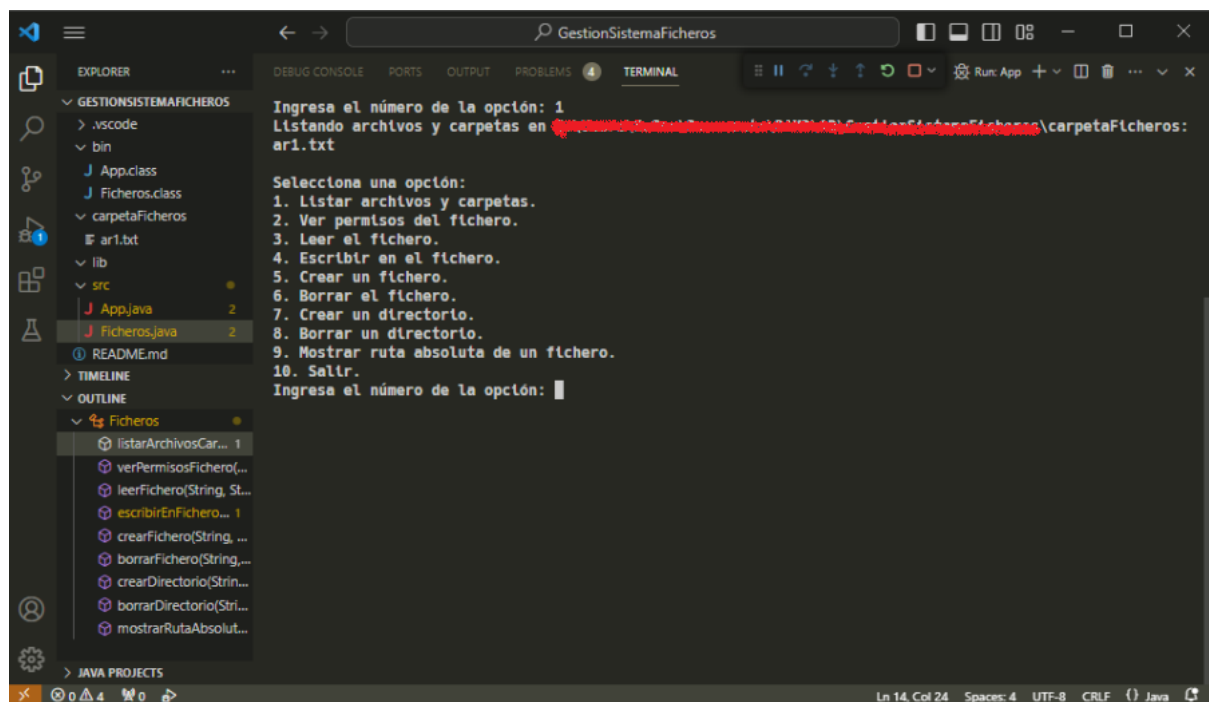
### 3 - Demostración

Partimos del escenario en que tenemos creado el directorio “carpetaFicheros” en el directorio de ejecución del programa

#### Menú



## Opción 1: Listar archivos y carpetas.

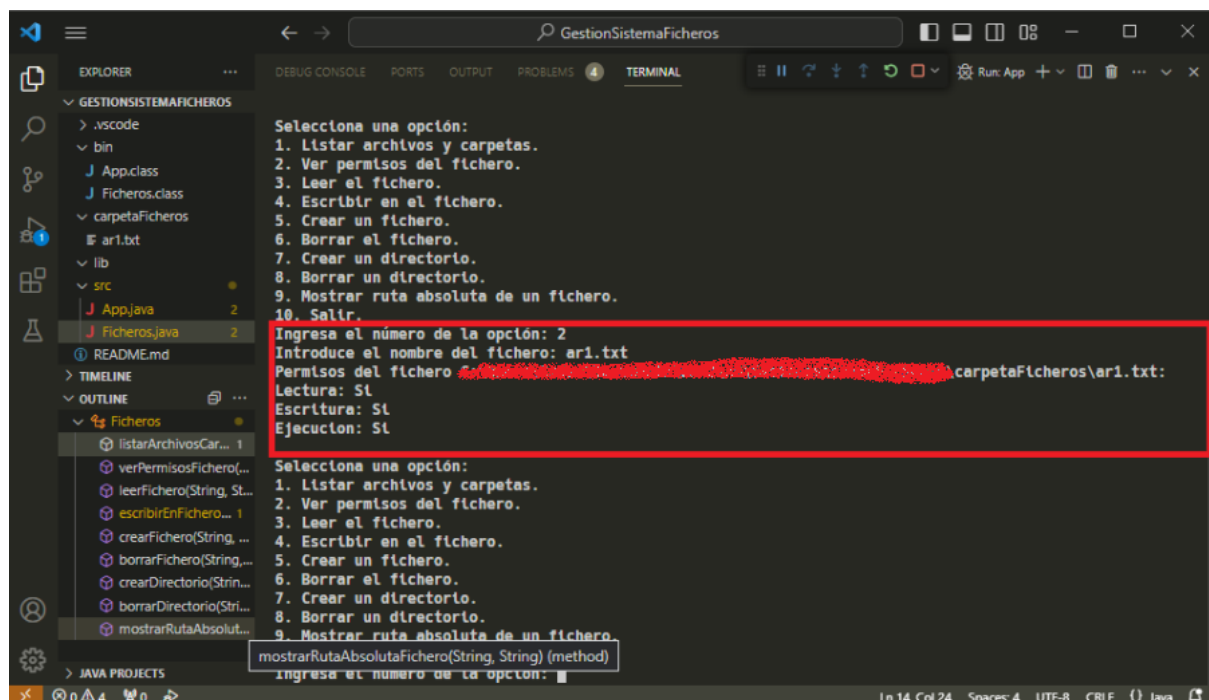


The screenshot shows the Visual Studio Code interface with a Java project named 'GestionSistemaFicheros'. The Explorer panel on the left shows the project structure, including a 'carpetaFicheros' directory. The Terminal panel on the right displays the application's output. The application prompts the user to enter an option number. The user has entered '1', and the application responds by listing the files and directories in 'carpetaFicheros\ar1.txt'.

```
Ingresa el número de la opción: 1
Listando archivos y carpetas en carpetaFicheros\ar1.txt

Selecciona una opción:
1. Listar archivos y carpetas.
2. Ver permisos del fichero.
3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.
Ingresa el número de la opción: 
```

## Opción 2: Ver permisos del fichero.

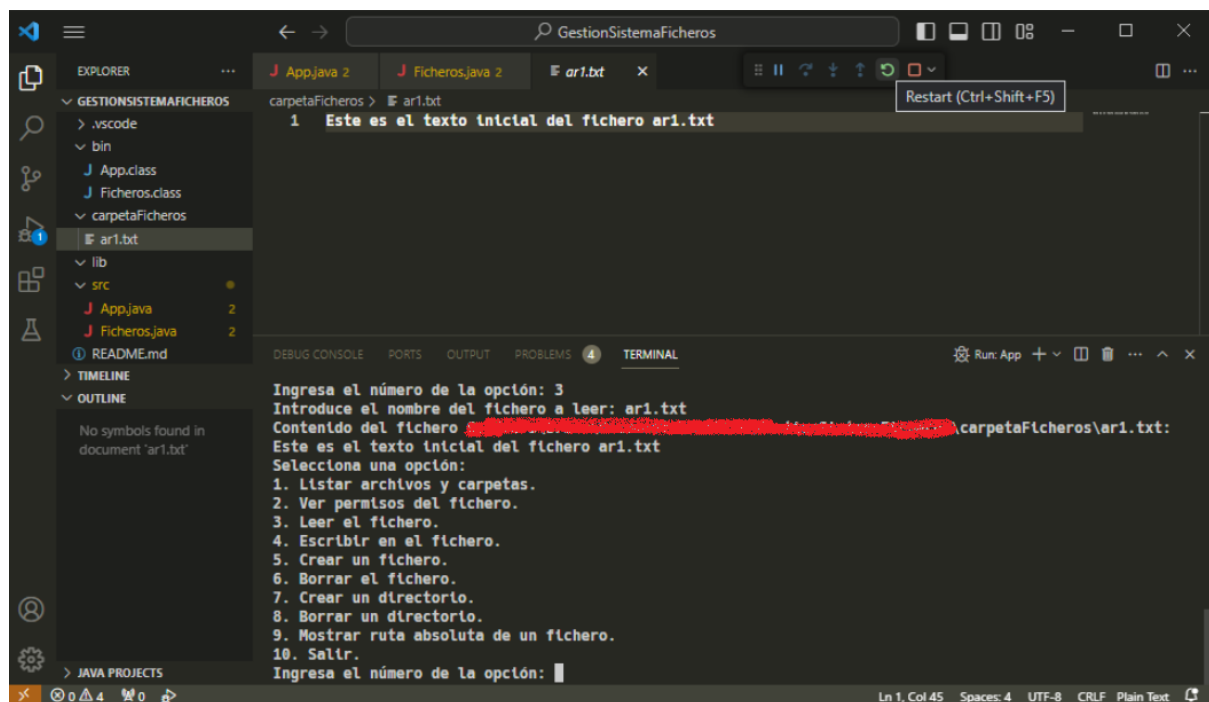


The screenshot shows the Visual Studio Code interface with the same Java project. The application prompts the user to enter an option number. The user has entered '2', and the application responds by asking for the file name. The user has entered 'ar1.txt', and the application displays the permissions for the file 'carpetaFicheros\ar1.txt'.

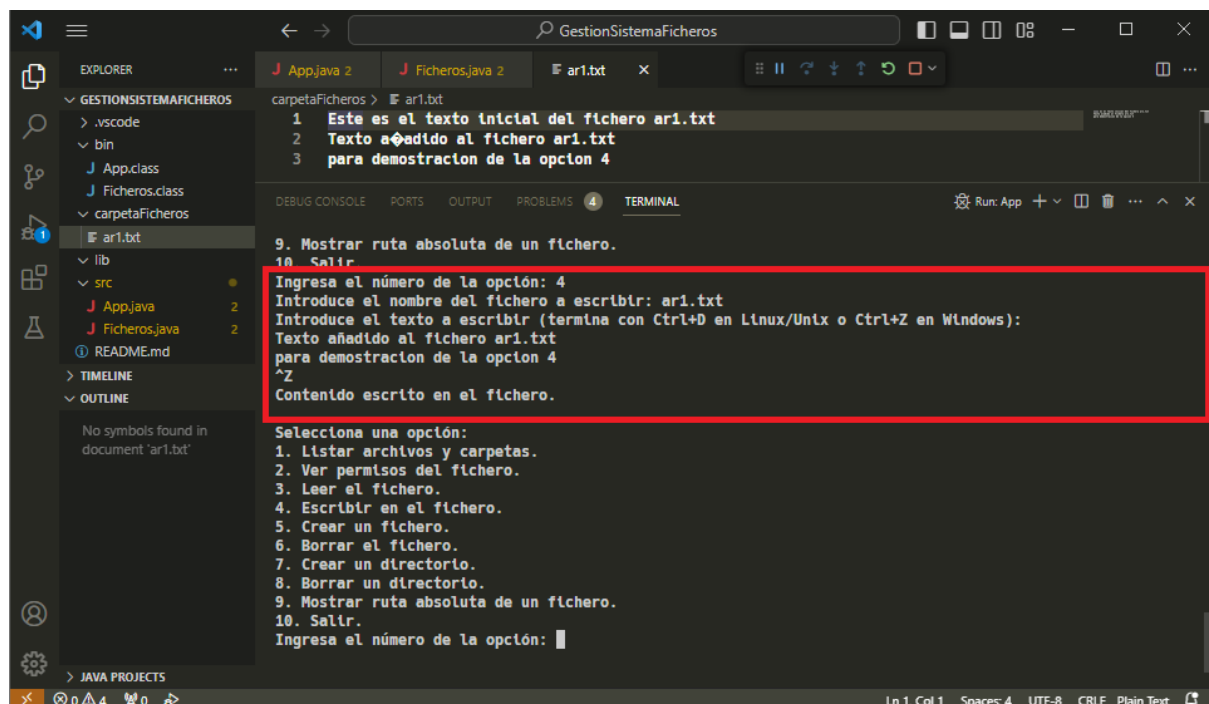
```
Selecciona una opción:
1. Listar archivos y carpetas.
2. Ver permisos del fichero.
3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.
Ingresa el número de la opción: 2
Introduce el nombre del fichero: ar1.txt
Permisos del fichero carpetaFicheros\ar1.txt:
Lectura: Sl
Ejecución: Sl

Selecciona una opción:
1. Listar archivos y carpetas.
2. Ver permisos del fichero.
3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.
Ingresa el número de la opción: 
```

### Opción 3: Leer el fichero.



Opción 4: Escribir en el fichero.



## Opción 5: Crear un fichero.

The screenshot shows the Visual Studio Code interface for a project named 'GestionSistemaFicheros'. The Explorer sidebar on the left shows the file structure with 'carpetaFicheros' expanded, containing 'ar1.txt' and 'ar2.txt'. The 'ar1.txt' file is selected and its content is displayed in the editor: 'Este es el texto inicial del fichero ar1.txt', 'Texto añadido al fichero ar1.txt', and 'para demostracion de la opcion 4'. The Terminal panel at the bottom shows the execution of a Java application. It prompts the user to 'Ingresa el número de la opción: 5' and 'Introduce el nombre del fichero a crear: ar2.txt', followed by the confirmation 'Archivo creado con éxito.'.

```
carpetaFicheros > ar1.txt
1 Este es el texto inicial del fichero ar1.txt
2 Texto añadido al fichero ar1.txt
3 para demostracion de la opcion 4

3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.

Ingresa el número de la opción: 5
Introduce el nombre del fichero a crear: ar2.txt
Archivo creado con éxito.

Selecciona una opción:
1. Listar archivos y carpetas.
2. Ver permisos del fichero.
3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.
Ingresa el número de la opción: 
```

## Opción 6: Borrar un fichero.

This screenshot shows the same VS Code environment as the previous one, but now the 'ar2.txt' file in the 'carpetaFicheros' directory is selected in the Explorer. The editor still shows the content of 'ar1.txt'. The Terminal panel shows the application prompting the user to 'Ingresa el número de la opción: 6' and 'Introduce el nombre del fichero a borrar: ar1.txt', followed by the confirmation 'Archivo borrado exitosamente.'.

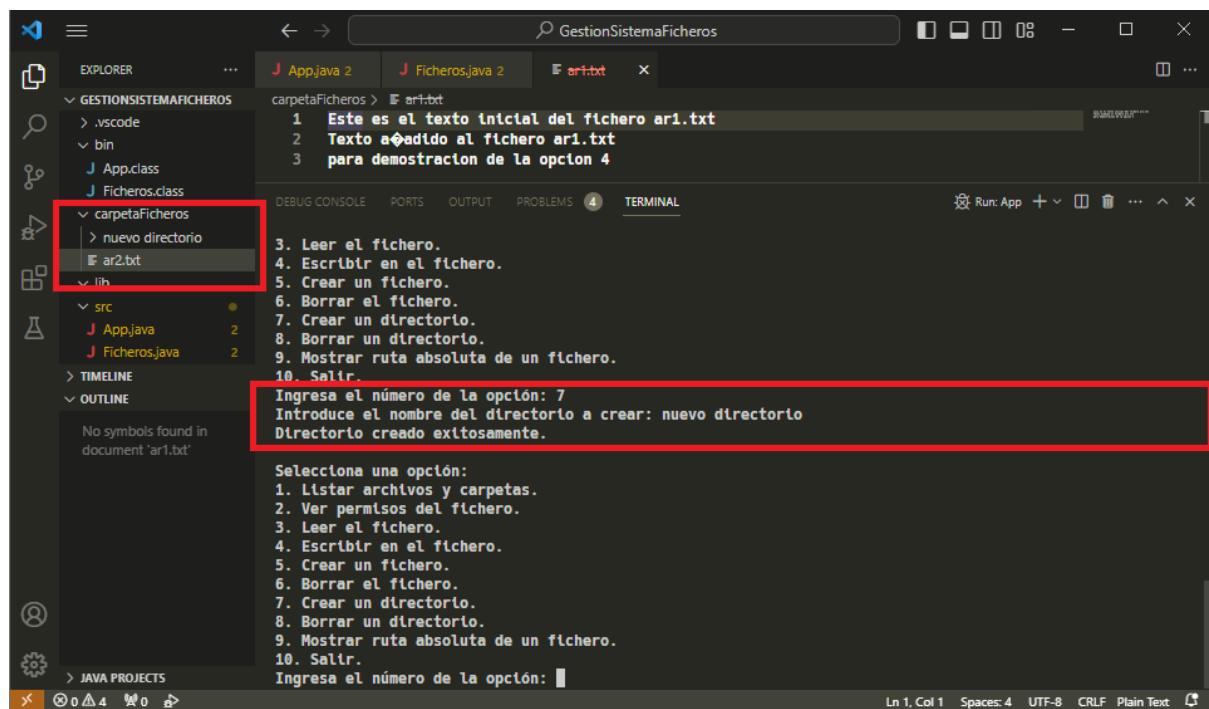
```
carpetaFicheros > ar1.txt
1 Este es el texto inicial del fichero ar1.txt
2 Texto añadido al fichero ar1.txt
3 para demostracion de la opcion 4

3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.

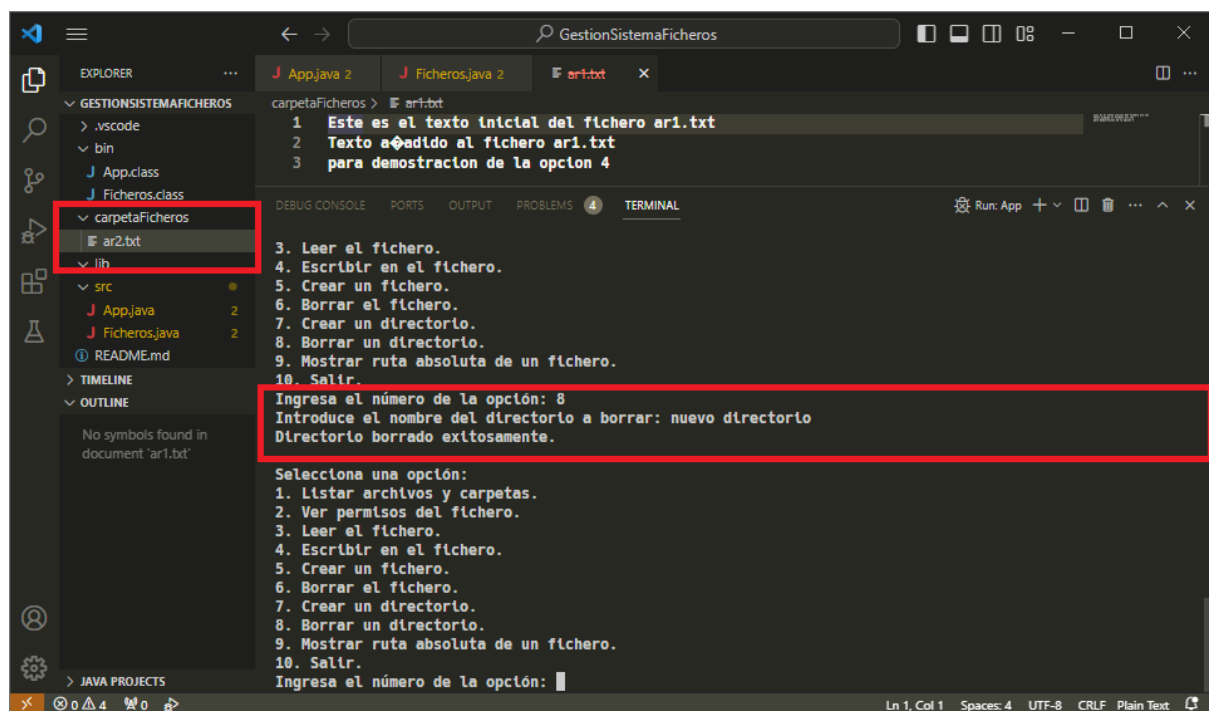
Ingresa el número de la opción: 6
Introduce el nombre del fichero a borrar: ar1.txt
Archivo borrado exitosamente.

Selecciona una opción:
1. Listar archivos y carpetas.
2. Ver permisos del fichero.
3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.
Ingresa el número de la opción: 
```

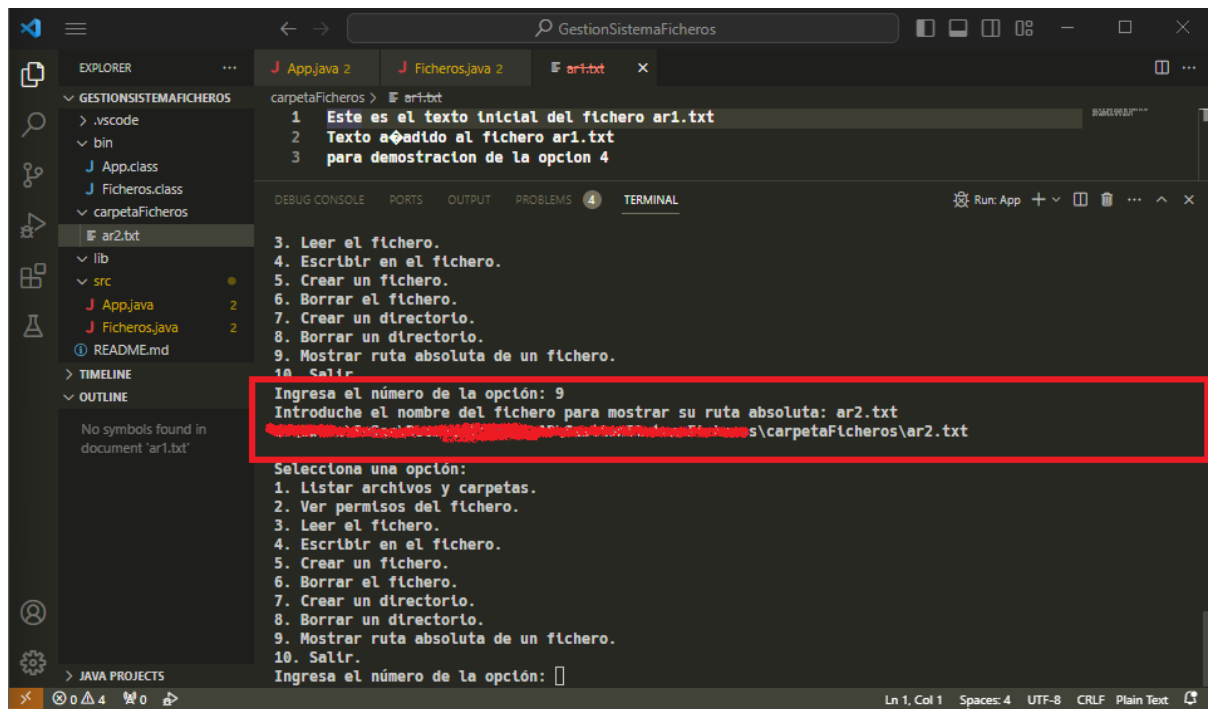
## Opción 7: Crear un directorio.



## Opción 8: Borrar un directorio.



## Opción 9: Mostrar ruta absoluta de un fichero.



```
1 Este es el texto inicial del fichero ar1.txt
2 Texto añadido al fichero ar1.txt
3 para demostración de la opción 4

3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.

Ingresar el número de la opción: 9
Introduce el nombre del fichero para mostrar su ruta absoluta: ar2.txt
C:\\Users\\User\\AppData\\Local\\Programs\\Microsoft VS Code\\carpetaFicheros\\ar2.txt

Selecciona una opción:
1. Listar archivos y carpetas.
2. Ver permisos del fichero.
3. Leer el fichero.
4. Escribir en el fichero.
5. Crear un fichero.
6. Borrar el fichero.
7. Crear un directorio.
8. Borrar un directorio.
9. Mostrar ruta absoluta de un fichero.
10. Salir.

Ingresar el número de la opción: 
```

Todo el código disponible en github:

<https://github.com/dsarotti/GestionSistemaFicheros>