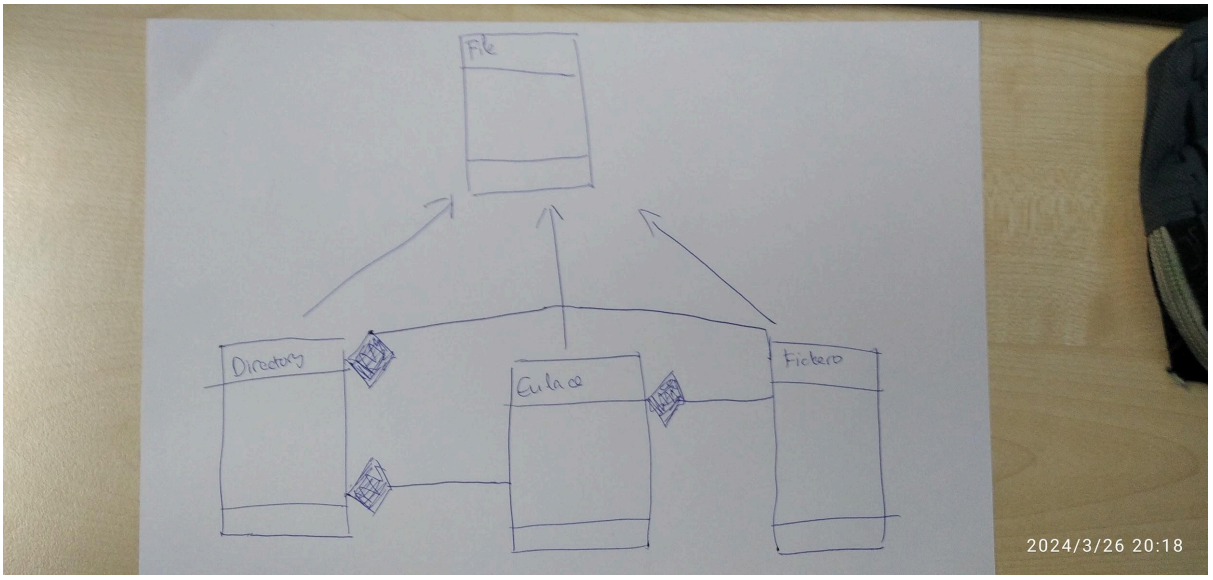
Problema 1

Diagrama Realiza un diagrama de como crees que podrían estar relacionadas las clases archivo y las que heredan de ella: fichero, directorio y enlace. Justifica brevemente tus decisiones de asociación

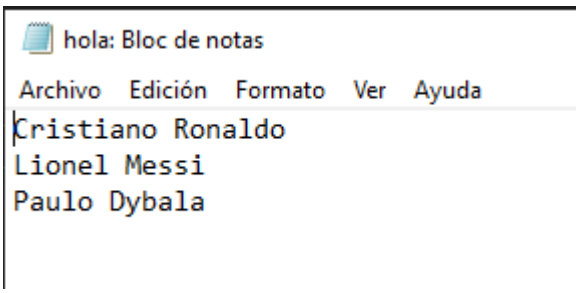


Directorio, enlace y fichero son tipos de archivos, un directorio o carpeta puede contener ficheros y enlaces dentro, pero si se elimina la carpeta estos desaparecen. Y he puesto asociación de composición entre fichero y enlace porque un enlace se crea de un fichero (digo esto porque entiendo que solo estamos trabajando con estas tres cosas, no enlaces a páginas web ni nada.)y si se elimina el fichero en enlace deja de servir.

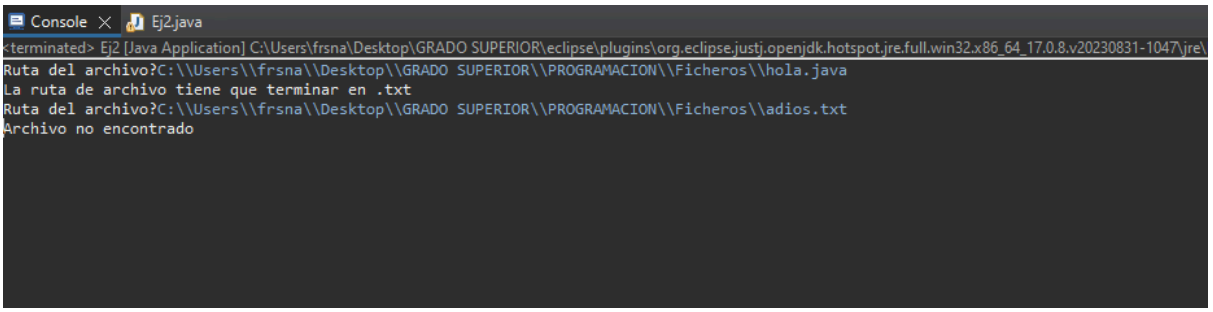
Problema 2:

Lectura Codifica un programa llamado programa 2 que realice el siguiente proceso secuencial:

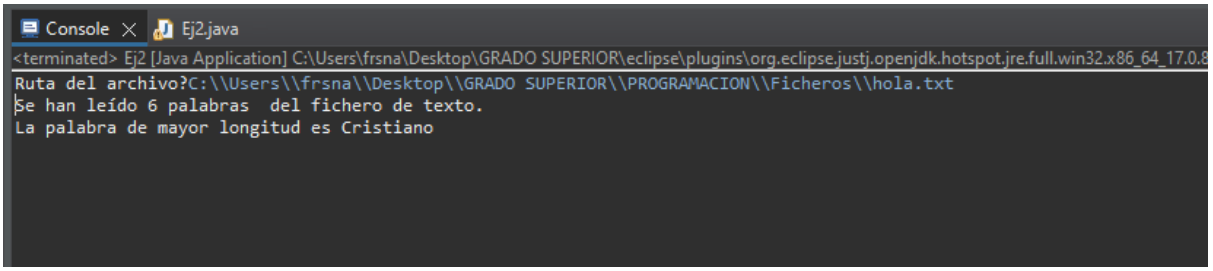
i) Recibirá una ruta (absoluta o relativa) usando Teclado de un supuesto archivo de texto y comprobará (antes de realizar la búsqueda) que el archivo realmente podría ser de texto (termina en .txt). En caso de no ser candidato a ello volverá a pedir el nombre hasta que se le dé uno válido avisando del formato adecuado por consola.



ii) Después, tratará de leer el archivo. Solamente en el caso de que no lo encuentre sacará por consola el mensaje: “Archivo no encontrado” y terminará el programa. Si lo encuentra continuará su ejecución normal.



iii) Por último, sacará por consola el número de palabras en el texto y la palabra de mayor longitud. Pista: revisar el método Split de la clase String para ello.



Console × Ej2.java ×

```java
package ejercicios;

import java.io.BufferedReader;
//C:\\Users\\frsna\\Desktop\\GRADO SUPERIOR\\PROGRAMACION\\Ficheros\\hola.txt

public class Ej2 {

    public static void main(String[] args) {
        boolean check = true;
        File ar = null;
        BufferedReader lector = null;
        String[] Vectorpalabras = null;
        int contador = 0;
        String linea;
        String palabra;
        String palabralarga = "";
        do {
            String ruta = Teclado.leerCadena("Ruta del archivo?");

            if (ruta.endsWith(".txt")) {
                check = true;
                ar = new File(ruta);
                try {
                    lector = new BufferedReader(new FileReader(ruta));
                    linea = lector.readLine();
                    while (linea != null) {
                        Vectorpalabras = linea.split("\\s");
                        for (int i = 0; i < Vectorpalabras.length; i++) {
                            palabra = Vectorpalabras[i];
                            contador++;
                            if (palabra.length() > palabralarga.length()) {
                                palabralarga = palabra;
                            }

                        }
                        linea = lector.readLine();
                    }
                    System.out.println("Se han leído " + contador + " palabras  del fichero de texto.");
                    System.out.println("La palabra de mayor longitud es " + palabralarga);
                    lector.close();

                    if (contador == 0) {
                        System.out.println("No se pudo leer el fichero porque esta vacio o corrupto");
                        lector.close();
                    }

                } catch (IOException e) {
                    System.out.println("Archivo no encontrado");
                }

            } else {
                System.out.println("La ruta de archivo tiene que terminar en .txt");
                check = false;
            }

        } while (check != true);

    }

}
```

Problema 3:

JSON y los internautas

i) Investiga qué son los archivos JSON y da un breve resumen.

Los archivos JSON o por sus siglas en inglés JavaScript Object Notation son una notación para expresar datos con el formato de los literales de objeto de JavaScript. Los archivos JSON son simples archivos de texto que contienen datos que se pueden consumir desde las aplicaciones.

ii) Sin usar ninguna librería nueva, crea una clase empleado con los atributos DNI, edad y salario y los método toString y obtenerJSON. El método obtenerJSON devolverá un String con el formato adecuado del objeto. Por ejemplo, si un empleado tiene los atributos DNI=11111111A, edad=50 y salario=1200, devolverá la siguiente cadena:

{DNI:11111111A, edad:50, salario:1200}

iii) Crea dos constructores en la clase empleado: uno que reciba los atributos y los instancie y otro que reciba una cadena con el formato JSON indicado arriba y cree el objeto de manera adecuada.

iv) Codifica una clase GestorEmpleados que use Teclado para leer por consola y gestione un menú de opciones que permita:

a) Crear un nuevo empleado y añadirlo a una colección

.

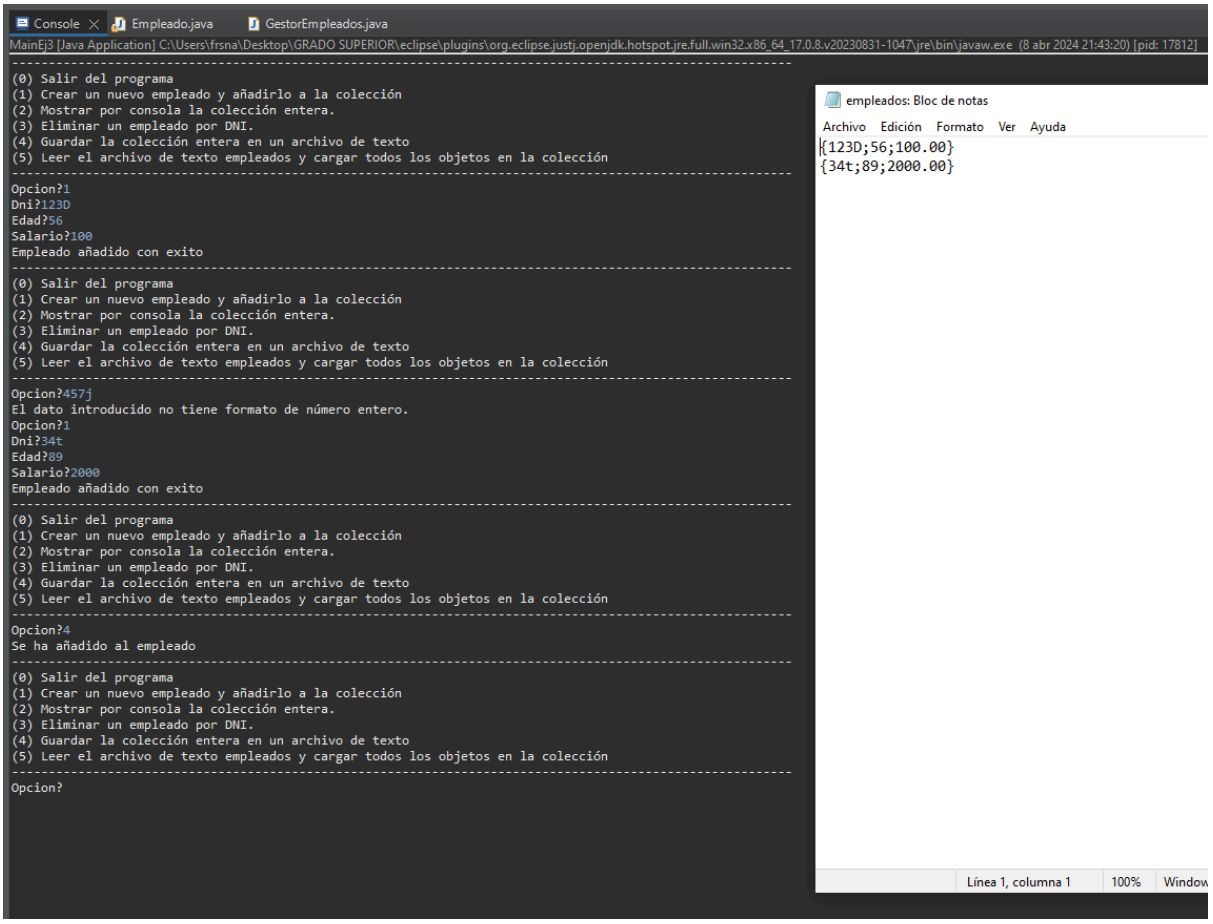
b) Mostrar por consola la colección entera.

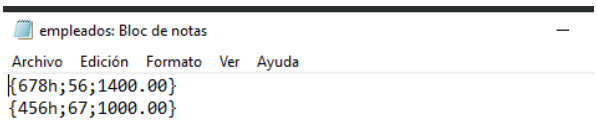
c) Eliminar un empleado por DNI.

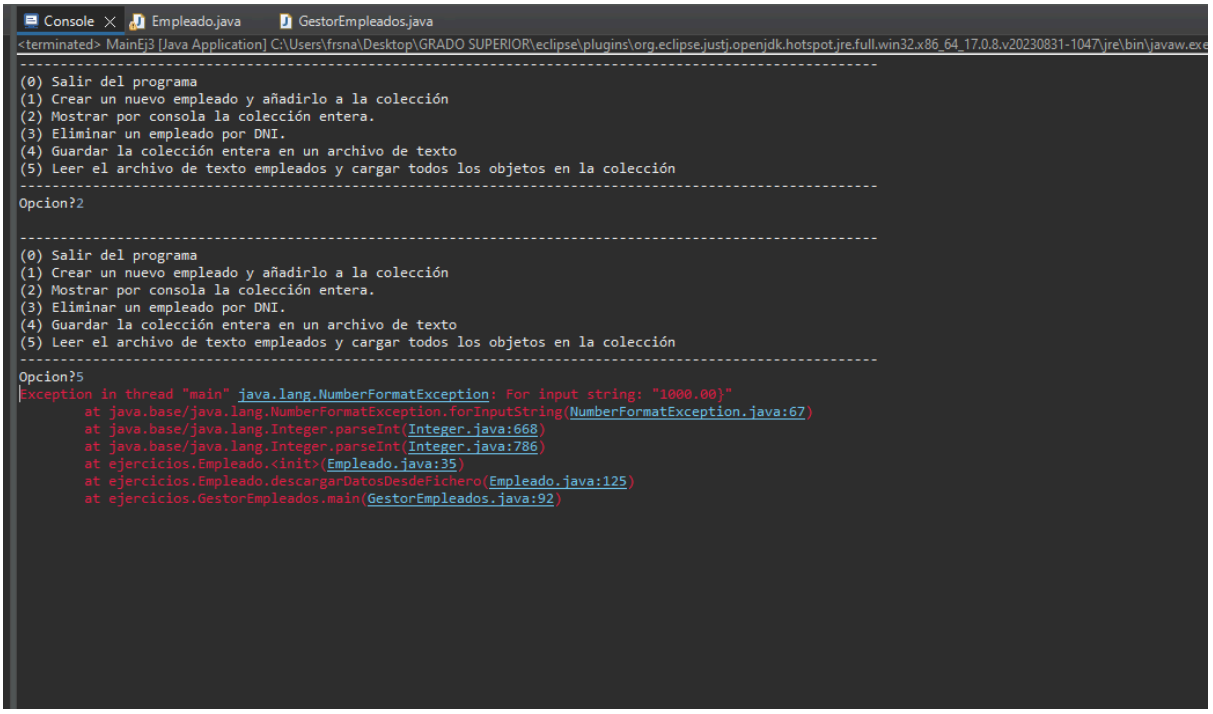
d) Guardar la colección entera en un archivo de texto llamado empleados sobreescribiendo donde se separarán los JSON de objetos distintos usando el carácter “;”. Por ejemplo, si tenemos en la colección a dos empleados con los datos: - Dni = 11111111A, edad = 22 y salario = 1000 - Dni = 22222222B, edad = 43 y salario = 1100 En el archivo debería leerse: {DNI:11111111A, edad:22, salario:1000};{DNI:22222222B, edad:43, salario:1100}

e) Leer el archivo de texto empleados y cargar todos los objetos en la colección

LAS CAPTURAS ESTÁN ABAJO


Console
Empleado.java
GestorEmpleados.java
MainEj3 [Java Application] C:\Users\frsna\Desktop\GRADO SUPERIOR\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe (8 abr 2024 21:43:20) [pid: 17812]
(0) Salir del programa
(1) Crear un nuevo empleado y añadirlo a la colección
(2) Mostrar por consola la colección entera.
(3) Eliminar un empleado por DNI.
(4) Guardar la colección entera en un archivo de texto
(5) Leer el archivo de texto empleados y cargar todos los objetos en la colección
Opcion?1
Dni?123D
Edad?56
Salario?100
Empleado añadido con exito
(0) Salir del programa
(1) Crear un nuevo empleado y añadirlo a la colección
(2) Mostrar por consola la colección entera.
(3) Eliminar un empleado por DNI.
(4) Guardar la colección entera en un archivo de texto
(5) Leer el archivo de texto empleados y cargar todos los objetos en la colección
Opcion?457j
El dato introducido no tiene formato de número entero.
Opcion?1
Dni?34t
Edad?89
Salario?2000
Empleado añadido con exito
(0) Salir del programa
(1) Crear un nuevo empleado y añadirlo a la colección
(2) Mostrar por consola la colección entera.
(3) Eliminar un empleado por DNI.
(4) Guardar la colección entera en un archivo de texto
(5) Leer el archivo de texto empleados y cargar todos los objetos en la colección
Opcion?4
Se ha añadido al empleado
(0) Salir del programa
(1) Crear un nuevo empleado y añadirlo a la colección
(2) Mostrar por consola la colección entera.
(3) Eliminar un empleado por DNI.
(4) Guardar la colección entera en un archivo de texto
(5) Leer el archivo de texto empleados y cargar todos los objetos en la colección
Opcion?
empleados: Bloc de notas
Archivo Edición Formato Ver Ayuda
{123D;56;100.00}
{34t;89;2000.00}
Línea 1, columna 1
100%
Windows


empleados: Bloc de notas
Archivo Edición Formato Ver Ayuda
{678h;56;1400.00}
{456h;67;1000.00}


Console
Empleado.java
GestorEmpleados.java
<terminated> MainEj3 [Java Application] C:\Users\frsna\Desktop\GRADO SUPERIOR\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe
(0) Salir del programa
(1) Crear un nuevo empleado y añadirlo a la colección
(2) Mostrar por consola la colección entera.
(3) Eliminar un empleado por DNI.
(4) Guardar la colección entera en un archivo de texto
(5) Leer el archivo de texto empleados y cargar todos los objetos en la colección
Opcion?2
(0) Salir del programa
(1) Crear un nuevo empleado y añadirlo a la colección
(2) Mostrar por consola la colección entera.
(3) Eliminar un empleado por DNI.
(4) Guardar la colección entera en un archivo de texto
(5) Leer el archivo de texto empleados y cargar todos los objetos en la colección
Opcion?5
Exception in thread "main" java.lang.NumberFormatException: For input string: "1000.00}"
at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:67)
at java.base/java.lang.Integer.parseInt(Integer.java:668)
at java.base/java.lang.Integer.parseInt(Integer.java:786)
at ejercicios.Empleado.<init>(Empleado.java:35)
at ejercicios.Empleado.descargarDatosDesdeFichero(Empleado.java:125)
at ejercicios.GestorEmpleados.main(GestorEmpleados.java:92)