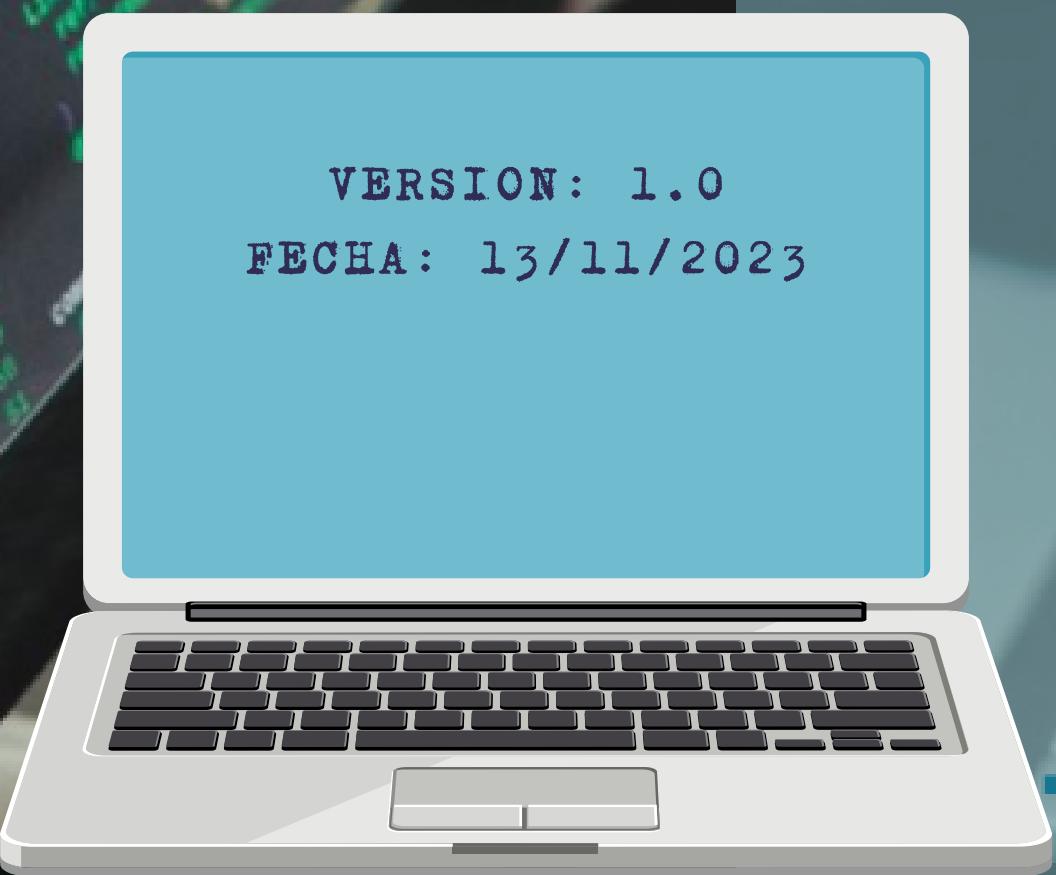


MANUAL DEL PROGRAMADOR

AUTOR: JOSE LUIS OBIANG ELA NANGUANG

PROFE: JUAN ARIAS MASA

ASIGNATURA: SEGURIDAD DE LA INFORMACIÓN



CIFRADO HILL



INDICE

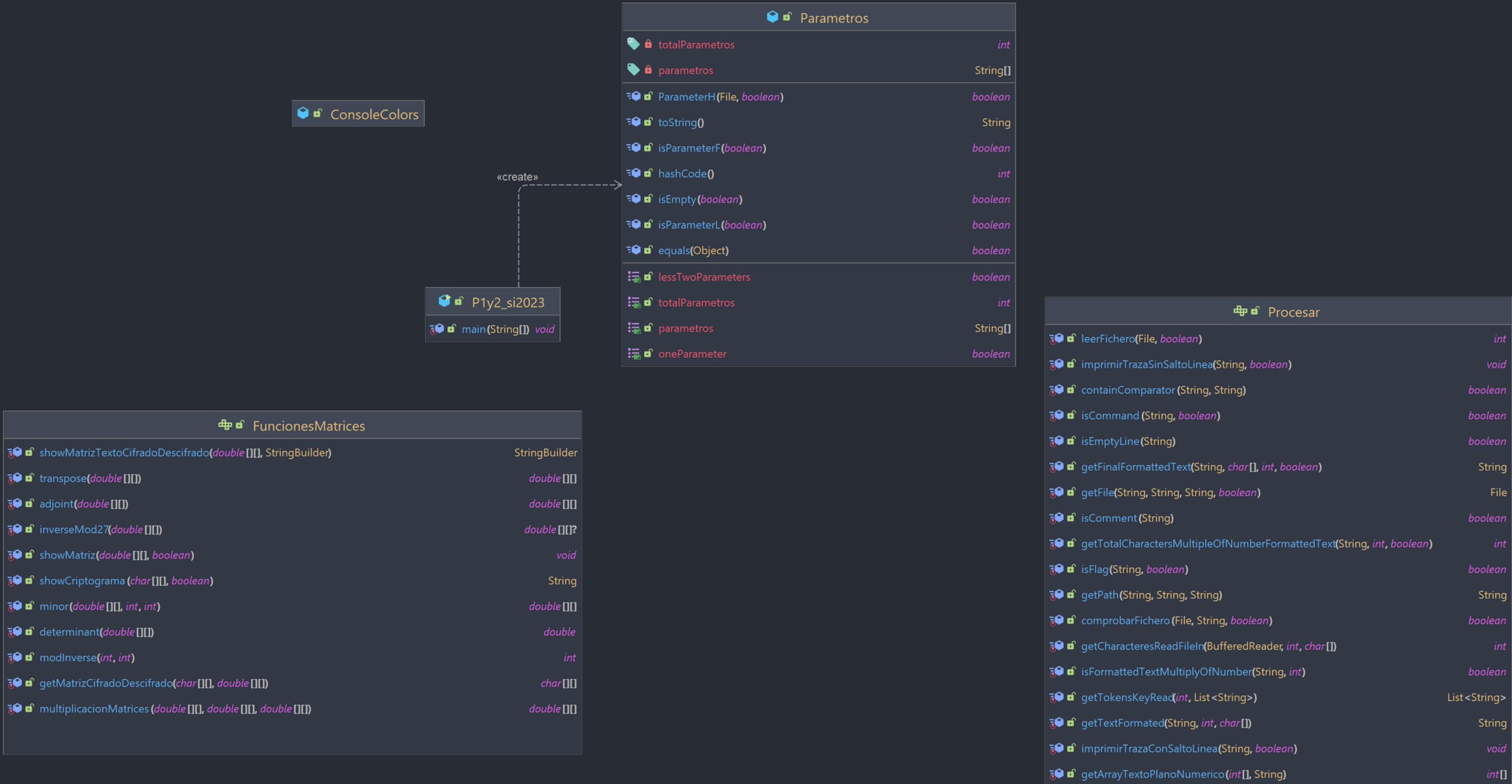
1. JAVA DOC	3
2. DIAGRAMA DEL DISEÑO DEL PROGRAMA	4
3. FUENTES	5

A dark-themed screenshot of a computer monitor. On the left, a file explorer sidebar shows several files and folders, including 'app/controllers', 'app/models', 'app/views', 'config', 'db', 'lib', 'public', 'spec', and 'tmp'. In the center, a terminal window displays a Ruby script, likely a test file, with syntax highlighting for code. The code includes require statements for 'spec_helper', 'rspec/rails', 'capybara/rspec', and 'capybara/rails'. It also contains configuration for Capybara's javascript_driver, Category.delete_all, Shoulda::Matchers.configuration, config.integrate, and test framework setup. A note at the bottom suggests adding additional requirements. The right side of the screen is mostly black, indicating a dark environment.

```
4  # Prevent database truncation if the transaction fails
5  abort("The Rails environment is running in production mode!")
6  require 'spec_helper'
7  require 'rspec/rails'
8
9  require 'capybara/rspec'
10 require 'capybara/rails'
11
12 Capybara.javascript_driver = :webkit
13 Category.delete_all; Category.create!(name: "Default")
14 Shoulda::Matchers.configure do |config|
15   config.integrate do |with|
16     with.test_framework :rspec
17     with.library :rails
18   end
19 end
20
21 # Add additional requires below this line if you need them
22
23 # Requires supporting ruby files with custom matchers and helpers
24 # in spec/support/ and its subdirectories. This directory also
25 # contains supporting files for this command:
26 # - spec/runner/runner.rb (which wraps the RSpec command)
27 # - spec/doublespy/doublespy.rb (which redefines double for spies)
28 # - spec/rspec/mocks/rspec_mocks.rb (which redefines some RSpec methods)
29
30 # Note: mongoid is not required by default. You can include it in your
31 # application's Gemfile if you want to use Mongoid's features. If you do,
32 # you must run bundle install before running rspec, as Bundler must
33 # know about the Mongoid gem in order for its features to be available
34 # in your tests.
```



2. DIAGRAMA



3. Fuentes

P1y2_si2023{

3 . Fuentes

FuncionesMatrices{



```
/*
 * Interfaz que contiene todas las operaciones de las matrices
 * @author Jose Luis Obiang Ela Nanguiang
 * @version 2.0
 */
public interface FuncionesMatrices {

    public static final int MOD = 27; //Modulo para la inversa modular

    /**
     * Multiplica dos matrices y devuelve el resultado
     * @param matrizClave Matriz de clave
     * @param matrizTextoPlanoNumerico Matriz de texto plano numerico
     * @param matrizCifradoNumerico Matriz de cifrado numerico, es el resultado de la multiplicacion
     * @return matriz resultante de la multiplicacion
     */
    public static double[][] multiplicacionMatrices(double[][] matrizClave, double[][] matrizTextoPlanoNumerico, double[][] matrizCifradoNumerico) {
        //Realizar multiplicacion de matrices para obtener el texto cifrado
        if (matrizClave == null && matrizTextoPlanoNumerico == null && matrizCifradoNumerico == null) {
            for (int i = 0; i < matrizClave.length; i++) {
                for (int j = 0; j < matrizClave[0].length; j++) {
                    for (int k = 0; k < matrizClave[0][j].length; k++) {
                        matrizCifradoNumerico[i][j] += matrizTextoPlanoNumerico[i][j] * matrizClave[i][k];
                    }
                    matrizCifradoNumerico[i][j] = matrizCifradoNumerico[i][j] % 27;
                }
            }
        }
        return matrizCifradoNumerico;
    }

    /**
     * Convierte una matriz numerica a una matriz de caracteres.
     * @param matrizTextoCifradoDescifrado recibe la matriz de texto cifrado/descifrado
     * @param matrizCifradoDescifradoNumerico recibe la matriz de cifrado/descifrado numerico
     * @return matriz de caracteres del texto cifrado/descifrado
     */
    public static char[][] getMatrixCifradoDescifrado(char[][] matrizTextoCifradoDescifrado, double[][] matrizCifradoDescifradoNumerico) {
        for (int i = 0; i < matrizCifradoDescifradoNumerico.length; i++) {
            for (int j = 0; j < matrizCifradoDescifradoNumerico[0].length; j++) {
                if ((int) matrizCifradoDescifradoNumerico[i][j] > 14) {
                    matrizTextoCifradoDescifrado[i][j] = 'N';
                } else if ((int) matrizCifradoDescifradoNumerico[i][j] > 14) {
                    matrizTextoCifradoDescifrado[i][j] = (char) (matrizCifradoDescifradoNumerico[i][j] + 65 - 1);
                } else {
                    matrizTextoCifradoDescifrado[i][j] = (char) (matrizCifradoDescifradoNumerico[i][j] + 65);
                }
            }
        }
        return matrizTextoCifradoDescifrado;
    }

    /**
     * Muestra la matriz numerica y la almacena en un StringBuilder.
     * @param matrizCifradoDescifradoNumerico recibe la matriz de cifrado/descifrado numerico
     * @param cifradoDescifrado recibe una cadena de caracteres cifrado/descifrado
     * @return cadena de caracteres del texto cifrado/descifrado
     */
    public static StringBuilder showMatrixTextoCifradoDescifrado(double[][] matrizCifradoDescifradoNumerico, StringBuilder cifradoDescifrado) {
        System.out.println("Total de filas: " + matrizCifradoDescifradoNumerico.length);
        System.out.println("Total de columnas: " + matrizCifradoDescifradoNumerico[0].length);
        for (int i = 0; i < matrizCifradoDescifradoNumerico.length; i++) {
            for (int j = 0; j < matrizCifradoDescifradoNumerico[i].length; j++) {
                System.out.print((int) (matrizCifradoDescifradoNumerico[i][j]) + " ");
            }
            cifradoDescifrado.append(matrizCifradoDescifradoNumerico[i][j]).append("\n"); //Almacenar cada uno de los caracteres cifrados en una cadena
        }
        System.out.println();
        return cifradoDescifrado;
    }

    /**
     * Muestra matrices numericas
     * @param matriz recibe la matriz numerica
     * @param estadoTrazo recibe el estado de la traza
     */
    public static void showMatrix(double[][] matriz, boolean estadoTrazo) {
        System.out.println();
        if (matriz != null) {
            for (double[] doubles : matriz) {
                for (int j = 0; j < matriz[0].length; j++) {
                    Procesar.ImprimirTrazoInSaltoLinea((int) doubles[j] + " ", estadoTrazo);
                }
                System.out.println();
            }
        }
    }

    /**
     * Muestra el criptograma dado la matriz de cifrado
     * @param matriz recibe la matriz de cifrado
     * @param estadoTrazo recibe el estado de la traza
     */
    public static String showCriptograma(char[][] matriz, boolean estadoTrazo) {
        StringBuilder criptograma = new StringBuilder();
        for (char[] element : matriz) {
            for (int j = 0; j < matriz[0].length; j++) {
                criptograma.append(element[j]);
                Procesar.ImprimirTrazoInSaltoLinea(element[j] + "", estadoTrazo);
            }
        }
        System.out.println();
        return criptograma.toString();
    }

    /**
     * Calcula el determinante de una matriz
     * @param matrix recibe la matriz numerica
     * @return el determinante de la matriz
     */
    public static double determinant(double[][] matrix) {
        double det = 0;
        if (matrix == null) {
            int n = matrix.length;
            if (n == 1) {
                return matrix[0][0];
            }
            if (n == 2) {
                return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0];
            }
            for (int j = 0; j < n; j++) {
                det += Math.pow(-1, j) * matrix[0][j] * determinant(minor(matrix, 0, j));
            }
        }
        return det;
    }

    /**
     * Devuelve la matriz menor al eliminar una fila y columna especificas.
     * @param matrix recibe la matriz numerica
     * @param row recibe la fila a eliminar
     * @param col recibe la columna a eliminar
     * @return la matriz menor al eliminar la fila y columna especificas
     */
    public static double[][] minor(double[][] matrix, int row, int col) {
        int n = matrix.length;
        double[][] minor = new double[n - 1][n - 1];
        int r = 0;
        for (int i = 0; i < n; i++) {
            if (i == row) continue;
            int c = 0;
            for (int j = 0; j < n; j++) {
                if (j == col) continue;
                minor[r][c] = matrix[i][j];
                c++;
            }
            r++;
        }
        return minor;
    }

    /**
     * Calcula la matriz adjunta
     * @param matrix recibe la matriz numerica
     * @return la matriz adjunta
     */
    public static double[][] adjoint(double[][] matrix) {
        int n = matrix.length;
        if (n == 1) {
            return new double[1][1];
        }
        double[][] cofactors = new double[n][n];
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                cofactors[i][j] = Math.pow(-1, i + j) * determinant(minor(matrix, i, j));
            }
        }
        return transpose(cofactors);
    }

    /**
     * Transpone una matriz.
     * @param matrix recibe la matriz numerica
     * @return la matriz transpuesta
     */
    public static double[][] transpose(double[][] matrix) {
        int n = matrix.length;
        double[][] transposed = new double[n][n];
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                transposed[j][i] = matrix[i][j];
            }
        }
        return transposed;
    }

    /**
     * Calcula la inversa modular de una matriz con respecto al modulo 27.
     * @param matrix recibe la matriz numerica
     * @return la matriz inversa modular de la matriz
     */
    public static double[][] inverseMod27(double[][] matrix) {
        double[][] inverse = null;
        if (matrix != null) {
            int n = matrix.length;
            double det = determinant(matrix);
            if (det == 0) return null;

            int detInv = modInverse((int) det, MOD);
            if (detInv == -1) return null;

            double[][] adjugate = adjoint(matrix);
            inverse = new double[n][n];

            for (int i = 0; i < n; i++) {
                for (int j = 0; j < n; j++) {
                    if (((adjugate[i][j] * detInv) % MOD) < 0) {
                        inverse[i][j] = 27 + (adjugate[i][j] * detInv) % MOD;
                    } else {
                        inverse[i][j] = (adjugate[i][j] * detInv) % MOD;
                    }
                }
            }
        }
        return inverse;
    }

    /**
     * Calcula el inverso modular de un numero con respecto a un modulo dado.
     * @param a recibe el numero a invertir
     * @param m recibe el modulo
     * @return el inverso modular de un numero con respecto a un modulo dado
     */
    public static int modInverse(int a, int m) {
        int m0 = m;
        int y = 0, x = 1;
        if (m == 1) return 0;
        while (a > 1) {
            int q = a / m;
            int t = m;
            m = a % m;
            a = t;
            y = x - q * y;
            x = t;
        }
        if (x < 0) x += m0;
        return x;
    }
}
```

3 . Fuentes

Procesar{

Parametros{

```

import java.io.File;
import java.io.IOException;
import java.util.Arrays;
import java.util.Objects;

/**
 * Clase que maneja los parametros de ejecucion del programa
 * @author Jose Luis Obiang Ela Nanguang
 * @version 2.0
 */
public class Parametros {

    private final int totalParametros; //Numero de parametros introducidos
    private String[] parametros; //Array de parametros introducidos

    /**
     * Constructor por defecto: Inicializa todas las variables de la clase.
     */
    public Parametros() {
        this.totalParametros = parametros.length;
        this.parametros = new String[totalParametros];
    }

    /**
     * Constructor parametrizado
     * @param parametros, recibe un arreglo de String con los parametros
     */
    public Parametros(String[] parametros) {
        this.parametros = parametros;
        this.totalParametros = parametros.length;
    }

    /**
     * Devuelve el numero de parametros introducidos
     * @return total de parametros
     */
    public int getTotalParametros() {
        return totalParametros;
    }

    /**
     * Devuelve todos los parametros introducidos
     * @return todos los parametros introducidos
     */
    public String[] getParametros() {
        return parametros;
    }

    /**
     * Para mostrar informacion de la clase
     * @return
     */
    @Override
    public String toString() {
        return "Parametros{" +
            "totalParametros=" + totalParametros +
            ", parametros=" + Arrays.toString(parametros) +
            '}';
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Parametros that = (Parametros) o;
        return totalParametros == that.totalParametros && Arrays.equals(parametros, that.parametros);
    }

    @Override
    public int hashCode() {
        int result = Objects.hash(totalParametros);
        result = 31 * result + Arrays.hashCode(parametros);
        return result;
    }
}

/**
 * Verifica si se ha introducido parametros
 * @return
 * <ul>
 *   <li>true: si NO se ha introducido parametros </li>
 *   <li>false: si se ha introducido parametros</li>
 * </ul>
 */
public boolean isEmpty(boolean estadoTraza) {
    if (totalParametros == 0) {
        Procesar.imprimirTrazaConSaltoLinea("No has introducido ningun parametro", estadoTraza);
        return true;
    }
    return false;
}

/**
 * Verifica si la cantidad de parametros es menor que 2
 * @return
 * <ul>
 *   <li>true: si la cantidad de parametros es menor que 2</li>
 *   <li>false: si la cantidad de parametros NO es menor que 2</li>
 * </ul>
 */
public boolean isLessTwoParameters() {
    return totalParametros < 2;
}

/**
 * Verifica si la cantidad de parametros es igual a 1
 * @return
 * <ul>
 *   <li>true: si la cantidad de parametros es igual a 1</li>
 *   <li>false: si la cantidad de parametros NO es igual a 1</li>
 * </ul>
 */
public boolean isOneParameter() {
    return totalParametros == 1;
}

/**
 * Verifica si el parametro es -h
 * @param estadoTraza Variable de tipo booleano, recibe el estado de la traza
 * @return
 * <ul>
 *   <li>true: es parametro -h</li>
 *   <li>false: NO es parametro -h</li>
 * </ul>
 */
public boolean ParameterH(File fileReadme, boolean estadoTraza) throws IOException {
    if (parametros != null && parametros.length > 0) {
        if (parametros[0].equals("-h") || parametros[0].equals("--help")) {
            return true;
        }
    }
    return false;
}

/**
 * Verifica si el parametro es -f
 * @param estadoTraza Variable de tipo booleano, recibe el estado de la traza
 * @return
 * <ul>
 *   <li>true: es parametro -f->Indica la entrada del fichero de configuracion config.txt</li>
 *   <li>false: NO es parametro -f</li>
 * </ul>
 */
public boolean isParameterF(boolean estadoTraza) {
    if (parametros != null && parametros.length > 0) {
        if (parametros[0].equals("-f") || parametros[0].equals("--fichero")) {
            return true;
        }
    }
    return false;
}

/**
 * Verifica si el parametro es -l
 * @param estadoTraza Variable de tipo booleano, recibe el estado de la traza
 * @return
 * <ul>
 *   <li>true: es parametro -l->Indica la entrada del fichero logs.txt</li>
 *   <li>false: NO es parametro -l</li>
 * </ul>
 */
public boolean isParameterL(boolean estadoTraza) {
    if (parametros != null && parametros.length > 0) {
        if (parametros[0].equals("-l") || parametros[0].equals("--logs")) {
            return true;
        }
    }
    return false;
}
}
}

```

ConsoleColors{

```
/*
 * Define constantes ANSI para colorear la salida de texto en la consola.
 * Estos códigos son útiles para resaltar la salida de texto en consola.
 *
 * @author Jose Luis Obiang Ela Nanguang
 * @version 1.0
 * @since 1.0
 */
public class ConsoleColors {
    // Reset
    public static final String RESET = "\u001B[0m"; // Text Reset

    // Regular Colors
    public static final String BLACK = "\u001B[0;30m"; // BLACK
    public static final String RED = "\u001B[0;31m"; // RED
    public static final String GREEN = "\u001B[0;32m"; // GREEN
    public static final String YELLOW = "\u001B[0;33m"; // YELLOW
    public static final String BLUE = "\u001B[0;34m"; // BLUE
    public static final String PURPLE = "\u001B[0;35m"; // PURPLE
    public static final String CYAN = "\u001B[0;36m"; // CYAN
    public static final String WHITE = "\u001B[0;37m"; // WHITE

    // Bold
    public static final String BLACK_BOLD = "\u001B[1;30m"; // BLACK
    public static final String RED_BOLD = "\u001B[1;31m"; // RED
    public static final String GREEN_BOLD = "\u001B[1;32m"; // GREEN
    public static final String YELLOW_BOLD = "\u001B[1;33m"; // YELLOW
    public static final String BLUE_BOLD = "\u001B[1;34m"; // BLUE
    public static final String PURPLE_BOLD = "\u001B[1;35m"; // PURPLE
    public static final String CYAN_BOLD = "\u001B[1;36m"; // CYAN
    public static final String WHITE_BOLD = "\u001B[1;37m"; // WHITE

    // Underline
    public static final String BLACK_UNDERLINED = "\u001B[4;30m"; // BLACK
    public static final String RED_UNDERLINED = "\u001B[4;31m"; // RED
    public static final String GREEN_UNDERLINED = "\u001B[4;32m"; // GREEN
    public static final String YELLOW_UNDERLINED = "\u001B[4;33m"; // YELLOW
    public static final String BLUE_UNDERLINED = "\u001B[4;34m"; // BLUE
    public static final String PURPLE_UNDERLINED = "\u001B[4;35m"; // PURPLE
    public static final String CYAN_UNDERLINED = "\u001B[4;36m"; // CYAN
    public static final String WHITE_UNDERLINED = "\u001B[4;37m"; // WHITE
}

// Background
public static final String BLACK_BACKGROUND = "\u001B[40m"; // BLACK
public static final String RED_BACKGROUND = "\u001B[41m"; // RED
public static final String GREEN_BACKGROUND = "\u001B[42m"; // GREEN
public static final String YELLOW_BACKGROUND = "\u001B[43m"; // YELLOW
public static final String BLUE_BACKGROUND = "\u001B[44m"; // BLUE
public static final String PURPLE_BACKGROUND = "\u001B[45m"; // PURPLE
public static final String CYAN_BACKGROUND = "\u001B[46m"; // CYAN
public static final String WHITE_BACKGROUND = "\u001B[47m"; // WHITE

// High Intensity
public static final String BLACK_BRIGHT = "\u001B[0;90m"; // BLACK
public static final String RED_BRIGHT = "\u001B[0;91m"; // RED
public static final String GREEN_BRIGHT = "\u001B[0;92m"; // GREEN
public static final String YELLOW_BRIGHT = "\u001B[0;93m"; // YELLOW
public static final String BLUE_BRIGHT = "\u001B[0;94m"; // BLUE
public static final String PURPLE_BRIGHT = "\u001B[0;95m"; // PURPLE
public static final String CYAN_BRIGHT = "\u001B[0;96m"; // CYAN
public static final String WHITE_BRIGHT = "\u001B[0;97m"; // WHITE

// Bold High Intensity
public static final String BLACK_BOLD_BRIGHT = "\u001B[1;90m"; // BLACK
public static final String RED_BOLD_BRIGHT = "\u001B[1;91m"; // RED
public static final String GREEN_BOLD_BRIGHT = "\u001B[1;92m"; // GREEN
public static final String YELLOW_BOLD_BRIGHT = "\u001B[1;93m"; // YELLOW
public static final String BLUE_BOLD_BRIGHT = "\u001B[1;94m"; // BLUE
public static final String PURPLE_BOLD_BRIGHT = "\u001B[1;95m"; // PURPLE
public static final String CYAN_BOLD_BRIGHT = "\u001B[1;96m"; // CYAN
public static final String WHITE_BOLD_BRIGHT = "\u001B[1;97m"; // WHITE

// High Intensity backgrounds
public static final String BLACK_BACKGROUND_BRIGHT = "\u001B[0;100m"; // BLACK
public static final String RED_BACKGROUND_BRIGHT = "\u001B[0;101m"; // RED
public static final String GREEN_BACKGROUND_BRIGHT = "\u001B[0;102m"; // GREEN
public static final String YELLOW_BACKGROUND_BRIGHT = "\u001B[0;103m"; // YELLOW
public static final String BLUE_BACKGROUND_BRIGHT = "\u001B[0;104m"; // BLUE
public static final String PURPLE_BACKGROUND_BRIGHT = "\u001B[0;105m"; // PURPLE
public static final String CYAN_BACKGROUND_BRIGHT = "\u001B[0;106m"; // CYAN
public static final String WHITE_BACKGROUND_BRIGHT = "\u001B[0;107m"; // WHITE}
```



**MUCHAS
GRACIAS**