Universidad ORT Uruguay

Facultad de Ingeniería

PROGRAMACIÓN II

Obligatorio I - Grupo ID M2D

Rodrigo Soca (327536)

Octubre 2023

Índice

CLASES DEL DOMINIO

Entrada.java:

```
package Dominio;


/**
 *
 * @author  Rodrigo Soca 327536
 */
public class Entrada {

    private String color;
    private char caracter;
//Constructor "cuando el color es Rojo o Azul en el constructor se
cambia la letra por su codigo para representarla"

    public Entrada(char unCaracter, String unColor) {
        if (unColor.equals("R") || unColor.equals("\033[31m")) {
            this.color = "\033[31m";
        } else if (unColor.equals("A") ||
unColor.equals("\033[34m")) {
            this.color = "\033[34m";
        }
        this.caracter = unCaracter;
    }
//Getters y Setters "cuando el color es Rojo o Azul en el
constructor se cambia la letra por su codigo para representarla"

    public String getColor() {
        return color;
    }

    public char getCaracter() {
        return caracter;
    }

    public void setColor(String unColor) {
        if (unColor.equals("R")) {
            this.color = "\033[31m";
        } else if (unColor.equals("A")) {
            this.color = "\033[34m";
        }
    }

    public void setCaracter(char unCaracter) {
        this.caracter = unCaracter;
    }
```

```
// modificacion del equals
    @Override
    public boolean equals(Object o) {
        Entrada e = (Entrada) o;
        return this.getCaracter() == e.getCaracter() &&
this.getColor().equals(e.getColor());
    }

//metodos
    //Verifica si la matriz esta resuelta
    public static Boolean estaResueltaMat(Entrada[][] mat) {
        String color = mat[0][0].getColor();
        boolean mismoColor = true;
        for (int i = 0; i < mat.length && mismoColor; i++) {
            for (int j = 0; j < mat[0].length && mismoColor; j++) {
              mismoColor = mat[i][j].getColor().equals(color);
            }
        }
        return mismoColor;
    }
    //Copia la matriz Creando una nueva
    public static Entrada[][] copiarMat(Entrada[][] mat) {
        Entrada[][] mat2 = new Entrada[mat.length][mat[0].length];
        for (int i = 0; i < mat.length; i++) {
            for (int j = 0; j < mat[0].length; j++) {
                char caracter = mat[i][j].getCaracter();
                String color = mat[i][j].getColor();
                mat2[i][j] = new Entrada(caracter, color);
            }
        }
        return mat2;
    }

 //modifica la matriz segun caracter y color de la posicion
[fila][col] en la matriz
    public static Entrada[][] modificarMat(Entrada[][] mat, int
fila, int col) {
        int posF = fila - 1;
        int posC = col - 1;
        char caracterEntrada = mat[posF][posC].getCaracter();

        switch (caracterEntrada) {
            case '-':
                // caso fila -
                for (int j = 0; j < mat[0].length; j++) {
                    cambiarColor(mat, posF, j);
                }
                break;
```

```
            case '|':
                // caso columna |
                for (int i = 0; i < mat.length; i++) {
                    cambiarColor(mat, i, posC);
                }
                break;
            case '/':
                // caso diagonal abajo izquierda o arriba derecha
                int cont2 = 0;
                for (int i = posF; i > 0; i--) {
                    cont2++;
                }
                posF = 0;
                posC += cont2;

                // Verificar límites antes de realizar cambios
                while (posC >= 0 && posF < mat.length) {
                    if (posF >= 0 && posF < mat.length && posC >= 0
&& posC < mat[0].length) {
                        cambiarColor(mat, posF, posC);
                    }
                    posF++;
                    posC--;
                }
                break;
            case '\\':
                // caso diagonal A ARRIBA izquierda o abajo derecha
                int cont = 0;
                int c = posC;
                for (int i = posF; i < mat.length && c <
mat[0].length; i++) {
                    cont++;
                    c++;
                }
                posF = mat.length - 1;
                posC += cont - 1;

                // Verificar límites antes de realizar cambios
                while (posF >= 0 && posC >= 0) {
                    if (posF >= 0 && posF < mat.length && posC >= 0
&& posC < mat[0].length) {
                        cambiarColor(mat, posF, posC);
                    }
                    posF--;
                    posC--;
                }
                break;
        }
```

```
        return mat;
    }
    //Si la entrada tiene un color lo cambio al otro

    public static void cambiarColor(Entrada[][] mat, int fila, int
col) {
        String color = "\033[31m"; //color rojo es "\033[31m"
        if (mat[fila][col].getColor().equals(color)) {
            mat[fila][col].setColor("A");
        } else {
            mat[fila][col].setColor("R");
        }

    }

}
```

Desafio.java:

```
package Dominio;

/**
 *
 * @author Rodrigo Soca 327536
 */
import static Dominio.Entrada.*;
import java.io.File;
import java.io.FileNotFoundException;

import java.util.Scanner;

public class Desafio {

    //atributos
    private Entrada[][] matAsociada;
    private int nivel;
    private String[] Solucion;

    //constructores
    public Desafio(Entrada[][] unaMat, int unNivel, String[]
unaSolucion) {
        this.matAsociada = unaMat;
        this.nivel = unNivel;
        this.Solucion = unaSolucion;
    }

    //construyo el desafio segun su tipo
    public Desafio(String tipo, int filas, int col, int nivel) {
        if (tipo.equalsIgnoreCase("predefinido")) {
            Entrada[][] mat = new Entrada[5][6];
            mat[0][0] = new Entrada('|', "A");
            mat[0][1] = new Entrada('|', "A");
            mat[0][2] = new Entrada('-', "R");
            mat[0][3] = new Entrada('/', "A");
            mat[0][4] = new Entrada('|', "R");
            mat[0][5] = new Entrada('-', "R");
            mat[1][0] = new Entrada('-', "R");
            mat[1][1] = new Entrada('/', "A");
            mat[1][2] = new Entrada('/', "A");
            mat[1][3] = new Entrada('|', "A");
            mat[1][4] = new Entrada('-', "R");
            mat[1][5] = new Entrada('-', "R");
```

```
            mat[2][0] = new Entrada('-', "R");
            mat[2][1] = new Entrada('-', "R");
            mat[2][2] = new Entrada('|', "A");
            mat[2][3] = new Entrada('-', "R");
            mat[2][4] = new Entrada('/', "R");
            mat[2][5] = new Entrada('-', "R");
            mat[3][0] = new Entrada('\\', "R");
            mat[3][1] = new Entrada('-', "R");
            mat[3][2] = new Entrada('|', "R");
            mat[3][3] = new Entrada('\\', "R");
            mat[3][4] = new Entrada('|', "A");
            mat[3][5] = new Entrada('|', "R");
            mat[4][0] = new Entrada('\\', "R");
            mat[4][1] = new Entrada('/', "R");
            mat[4][2] = new Entrada('/', "R");
            mat[4][3] = new Entrada('|', "A");
            mat[4][4] = new Entrada('/', "A");
            mat[4][5] = new Entrada('\\', "A");

            String[] solucion = new String[]{"4 4", "5 6", "5 4"};
            this.matAsociada = mat;
            this.nivel = 3;
            this.Solucion = solucion;

        } else if (tipo.equalsIgnoreCase("datos")) {

            try {
                Scanner scanner = new Scanner(new
File(".\\Test\\datos.txt"));
                Desafio des = desafioDeScanner(scanner);
                this.Solucion = des.Solucion;
                this.nivel = des.nivel;
                this.matAsociada = des.getMatAsociada();
            } catch (FileNotFoundException ex) {
                System.out.print("error");
            }

        } else if (tipo.equalsIgnoreCase("azar")) {
            Entrada[][] mat = new Entrada[filas][col];
            int numColor = (int) (Math.random() * 2 + 1);
            for (int i = 0; i < mat.length; i++) {
                for (int j = 0; j < mat[0].length; j++) {

                    int numChar = (int) (Math.random() * 4 + 1);
                    switch (numColor) {
                        case 1 -> {
                            switch (numChar) {
                                case 1 -> {
```

```
                                mat[i][j] = new Entrada('/', "R");
                            }
                        case 2 -> {
                             mat[i][j] = new Entrada('\\', "R");
                            }
                            case 3 -> {
                              mat[i][j] = new Entrada('-', "R");
                            }
                            case 4 -> {
                              mat[i][j] = new Entrada('|', "R");
                            }
                        }
                    }
                    case 2 -> {
                        switch (numChar) {
                            case 1 -> {
                             mat[i][j] = new Entrada('/', "A");
                            }
                            case 2 -> {
                             mat[i][j] = new Entrada('\\', "A");
                            }
                            case 3 -> {
                              mat[i][j] = new Entrada('-', "A");
                            }
                            case 4 -> {
                              mat[i][j] = new Entrada('|', "A");
                            }
                        }
                    }

                }
            }
        }
        String[] solucion = new String[nivel];
        for (int n = 0; n < nivel; n++) {
           int numFila = (int) (Math.random() * mat.length) + 1;
         int numCol = (int) (Math.random() * mat[0].length) + 1;
            modificarMat(mat, numFila, numCol);

            solucion[n] = Integer.toString(numFila) + " " +
Integer.toString(numCol);

        }
        this.Solucion = solucion;
        this.matAsociada = mat;
        this.nivel = nivel;

    }
```

```
}

//getters y setters
public Entrada[][] getMatAsociada() {
    return matAsociada;
}

public int getNivel() {
    return nivel;
}

public String[] getSolucion() {
    return Solucion;
}

public void setMatAsociada(Entrada[][] matAsociada) {
    this.matAsociada = matAsociada;
}

public void setNivel(int nivel) {
    this.nivel = nivel;
}

public void setSolucion(String[] Solucion) {
    this.Solucion = Solucion;
}

//metodos
//crea un desafio segun el scanner
public static Desafio desafioDeScanner(Scanner in) {

    int n = in.nextInt();
    int m = in.nextInt();
    in.nextLine();
    Entrada[][] mat = new Entrada[n][m];
    for (int i = 0; i < mat.length; i++) {
        String frase = in.nextLine();
        String[] datos = frase.split(" ");

        for (int j = 0; j < m; j++) {

            if (datos[j].charAt(1) == 'R') {
                mat[i][j] = new Entrada(datos[j].charAt(0),"R");
            } else if (datos[j].charAt(1) == 'A') {
                mat[i][j] = new Entrada(datos[j].charAt(0),"A");
            }

        }
```

```
        }
        int nivel = Integer.parseInt(in.nextLine());

        String[] Solucion = new String[nivel];
        for (int i = 0; i < nivel; i++) {
            String fraseSol = in.nextLine();

            Solucion[i] = fraseSol;
        }
        in.close();
        Desafio des = new Desafio(mat, nivel, Solucion);

        return des;
    }

}
```

Pantalla.java:

```
package Interfaz;

/**
 *
 * @author  Rodrigo Soca 327536
 */

import Dominio.*;

public class Pantalla {


    // imprimir
decorados----------------------------------------------------------------
----------------------------------------------------------------
    //imprime los numeros arriba de la matriz
    public static void patron0(int cols) {
        System.out.print(" ");
        String patron = "   ";
        for (int i = 0; i < cols; i++) {
            int num = i + 1;
            System.out.print(patron + num);
        }
    }

    //imprime los separadores de la matriz
    public static void patron1(int cols) {
        System.out.print("  +");
        String patron = "---+";
        for (int i = 0; i < cols; i++) {
            System.out.print(patron);

        }
    }

    // muestra el tablero con la decoracion
----------------------------------------------------------------------
----------------------------------------
    public static void mostrarTablero(Entrada[][] mat) {
        String b = "\u001B[0m"; //borrar
        patron0(mat[0].length);
        System.out.println();

        patron1(mat[0].length);
        System.out.println();
```

```
        for (int i = 0; i < mat.length; i++) {
            String indice = String.valueOf(i + 1);
            System.out.print(indice + " " + "|");
            for (int j = 0; j < mat[0].length; j++) {
                System.out.print(" " + mat[i][j].getColor() +
mat[i][j].getCaracter() + b + " " + "|");
            }
            System.out.println();
            patron1(mat[0].length);
            System.out.println();

        }
    }

    public static void mostrarTableroDe2(Entrada[][] mat1,
Entrada[][] mat2) {
        int cols1 = mat1[0].length;
        int cols2 = mat2[0].length;
        String b = "\u001B[0m"; // borrar

        // Imprimir la primera matriz
        patron0(cols1);
        System.out.print("         ");
        patron0(cols2);
        System.out.println();

        patron1(cols1);
        System.out.print("  ==>  ");
        patron1(cols2);
        System.out.println();

        for (int i = 0; i < mat1.length; i++) {
            String indice = String.valueOf(i + 1);
            System.out.print(indice + " " + "|");
            for (int j = 0; j < cols1; j++) {
                System.out.print(" " + mat1[i][j].getColor() +
mat1[i][j].getCaracter() + b + " " + "|");
            }
            System.out.print("  ==>  "); // Separador entre las
matrices
            //imprimir segunda mat
            System.out.print(indice + " " + "|");
            for (int j = 0; j < cols2; j++) {
                System.out.print(" " + mat2[i][j].getColor() +
mat2[i][j].getCaracter() + b + " " + "|");
            }
            System.out.println();
```

```
            patron1(cols1);
            System.out.print("  ==>  ");
            patron1(cols2);
            System.out.println();
        }
    }
    public static void mostrarTiempo(long milisegundos) {

        long segundosTotales = milisegundos / 1000;
        int horas = (int) (segundosTotales / 3600);
        int minutos = (int) ((segundosTotales % 3600) / 60);
        int segundos = (int) (segundosTotales % 60);
        System.out.print(horas+"h:"+minutos+"m:"+segundos+"s");
    }
}
```

Partida.java :

```
package Interfaz;

/**
 *
 * @author Rodrigo Soca 327536
 */
import Dominio.*;
import static Dominio.Entrada.*;
import static Interfaz.Pantalla.*;
import java.util.*;

public class Partida {
//crea la partida segun el desafio

    public static void partida(Desafio des) throws Exception {
        long tInicio;
        long tFin;
        long tTotal;
        Entrada[][] mat1 = des.getMatAsociada();
        String[] solucion = des.getSolucion();
        Scanner in = new Scanner(System.in);
        ArrayList<Integer> historial = new ArrayList<Integer>();
        tInicio = new Date().getTime();
        //muestro opciones y inicializo el tiempo
        boolean volverMenu = false;
        while (!volverMenu) {
            System.out.println();
            System.out.println("Ingrese una opcion X,S o H  o
Ingrese un movimiento la fila y la columna se ingresan de a una por
vez, por separado");
            System.out.println("Si ingresas -1 -1 se retrocedera 1
movimiento");
            System.out.println();
            System.out.println("X) => Terminar partida y volver a
menu");
            System.out.println("S) => Ver solucion del tablero
original");
            System.out.println("H) => Ver el historial de
movimientos ");
            System.out.println();
            boolean correcto = false;
            while (!correcto) {

                try {

                    String frase = in.nextLine();
                    frase = frase.toLowerCase();
```

```
                    char opcion = frase.charAt(0);
                    if (opcion == 'x' || opcion == 's' || opcion ==
'h') {

                        correcto = true;
                        switch (opcion) {
                            //salir de la partida
                            case 'x': {
                                System.out.println("Si vuelves al
menu se borrara el progreso de la partida");
                                System.out.println();
                                System.out.println("En caso de
desear salir escribe SI en otro caso presiona cualquier boton");
                                frase = in.nextLine();
                                frase = frase.toLowerCase();
                                System.out.println();
                                if (frase.equals("si")) {
                                    tFin = new Date().getTime();
                                    tTotal = tFin - tInicio;
                                    System.out.println();
                                    System.out.print("Jugaste en
esta partida: ");
                                    mostrarTiempo(tTotal);
                                    System.out.println();
                                    volverMenu = true;
                                    break;
                                } else {
                                    volverMenu = false;

                                }

                            }
                            break;
                            //mostrar solucion
                            case 's': {
                                System.out.println("Solucion del
tablero original");

System.out.println(Arrays.toString(solucion));
                                System.out.println();
                            }
                            break;
                            //mostrar historial
                            case 'h': {
                    System.out.println("Historial de movimientos");

                            System.out.println();
                                if (historial.size() > 0) {
```

```
                                    System.out.print("[" +
historial.get(0) + " " + historial.get(1));
                                    for (int i = 2; i <
historial.size() - 1; i += 2) {
                                        System.out.print(", " +
historial.get(i) + " " + historial.get(i + 1));
                                    }
                                    System.out.print("]");

                                } else {
                                    System.out.print("No se han
realizado movimientos");

                                }
                                System.out.println(" ");

                            }
                            break;

                        }
                        //si no se ingreso un caracter de las
opciones uso try catch para ver si habian ingresado un integer
                    } else {
                        try {
                            int num1 = Integer.parseInt(frase);
                            String frase2 = in.nextLine();
                            int num2 = Integer.parseInt(frase2);
                            Entrada[][] mat2 = copiarMat(mat1);
                            //el entero ingresado no se puede salir
del rango de la matriz
                            if ((num1 <= mat1.length && num1 > 0 &&
num2 <= mat1[0].length && num2 > 0) || (num1 == -1 && num2 == -1)) {
                                correcto = true;
                                //si se ingresa -1 -1 retrocedo un
paso consultando el historial y actualizo el historial borrando el
ultimo movimiento
                                if (num1 == -1 && num2 == -1) {
                                    if (historial.size() > 1) {
                                        int numCol =
historial.get(historial.size() - 1);
                                        int numFil =
historial.get(historial.size() - 2);
                                        modificarMat(mat2, numFil,
numCol);
                                        mostrarTableroDe2(mat1,
mat2);
                                        System.out.println();
                                        mostrarTablero(mat2);
```

```
                                    mat1 = copiarMat(mat2);

historial.remove(historial.size() - 2);

historial.remove(historial.size() - 1);

                                } else {
                                    System.out.println("No
realizaste Movimientos anteriormente");
                                    System.out.println();
                                }
                            } else {
                                //Modifico la matriz y actualizo
la anterior
                                historial.add(num1);
                                historial.add(num2);
                                modificarMat(mat2, num1, num2);
                                mostrarTableroDe2(mat1, mat2);
                                System.out.println();
                                mostrarTablero(mat2);
                                mat1 = copiarMat(mat2);
                                //consulto si resolvio la matriz
en ese caso muestro un mensaje de felicitaciones y el tiempo que
toma
                                if ((estaResueltaMat(mat2)) {
                                    System.out.println();

System.out.println("Felicitaciones Ganaste");
                                    tFin = new Date().getTime();
                                    tTotal = tFin - tInicio;
                                    System.out.println();
                                    System.out.print("Lo
resolviste en: ");
                                    mostrarTiempo(tTotal);
                                    System.out.println();
                                    volverMenu = true;

                                } else {
                                    volverMenu = false;
                                }

                            }
                        } else {
                            throw new Exception("El numero Esta
fuera de rango");
                        }

                    } catch (Exception e) {
```

```
                            throw new Exception("Entrada no
valida");

                        }
                    }

                } catch (Exception e) {
                    System.out.println("Error =>" + e.getMessage());
                    System.out.println("Ingrese una opcion X,S o H
o Ingrese numero valido");
                    System.out.println();
                    System.out.println("X) => Terminar partida y
volver a menu");
                    System.out.println("S) => Ver solucion del
tablero original");
                    System.out.println("H) => Ver el historial de
movimientos ");
                    System.out.println();
                }
            }
        }
    }

    public static void jugar() {
        Scanner in = new Scanner(System.in);
        boolean terminar = false;
        while (!terminar) {
            System.out.println("Menu ");
            System.out.println();
            System.out.println();
            System.out.println("Ingrese una opcion A,B,C o X");
            System.out.println();
            System.out.println("A) => Crear partida tomando datos
del archivo 'datos.txt'");
            System.out.println("B) => Crear partida usando el
tablero predefinido");
            System.out.println("C) => Crear partida usando un
tablero al azar");
            System.out.println("X) => Salir del juego");
            System.out.println();
            boolean correcto = false;
            while (!correcto) {
                try {
                    String frase = in.nextLine();

                    frase = frase.toLowerCase();
                    char opcion = frase.charAt(0);
```

```
                    if (opcion == 'a' || opcion == 'b' || opcion ==
'c' || opcion == 'x') {
                        correcto = true;
                    } else {
                        throw new Exception("caracter no valido");
                    }

                    switch (opcion) {
                        //creo partida con datos.txt
                        case 'a': {
                            System.out.println("Elegiste crear la
partida con el tablero del archivo 'datos.txt'");
                            Desafio des = new Desafio("datos", 0, 0,
0);
                            Entrada[][] mat = des.getMatAsociada();
                            mostrarTablero(mat);
                            partida(des);
                        }
                        break;
                        //creo partida con tablero predefinido
                        case 'b': {
                            System.out.println("Elegiste crear la
partida con el tablero predefinido");
                            System.out.println();
                            Desafio des = new Desafio("predefinido",
5, 6, 3);
                            Entrada[][] mat = des.getMatAsociada();
                            mostrarTablero(mat);
                            partida(des);

                        }
                        break;
                        //creo partida con un tablero al azar
                        case 'c': {
                            boolean correcto2 = false;

                            System.out.println("Elegiste crear la
partida con el tablero al azar");
                            while (!correcto2)
                            try {
                                System.out.println("ingrese cantidad
de filas de 3 a 9");
                                int filas = in.nextInt();
                                System.out.println("ingrese cantidad
de Columnas de 3 a 9");
                                int columnas = in.nextInt();
                                System.out.println("ingrese  el
nivel de 1 a 8");
```

```
                                int nivel = in.nextInt();

                                if (filas < 3 || filas > 9) {
                                    throw new Exception("cantidad de
filas incorrecto");
                                } else if (columnas < 3 || columnas
> 9) {
                                    throw new Exception("cantidad de
columnas incorrecto");
                                } else if (nivel < 1 || nivel > 8) {

                                    throw new Exception("nivel
incorrecto");
                                } else if (filas >= 3 && filas <= 9
&& columnas >= 3 && columnas <= 9 && nivel >= 1 && nivel <= 8) {
                                    correcto2 = true;
                                    Desafio des = new
Desafio("azar", filas, columnas, nivel);
                                    Entrada[][] mat =
des.getMatAsociada();
                                    mostrarTablero(mat);
                                    partida(des);
                                    in.nextLine();
                                }
                            } catch (Exception e) {
                                System.out.println("Error=>" +
e.getMessage());
                                System.out.println("ingrese valores
dentro del Rango");
                                System.out.println();
                            }
                        }
                        break;

                        case 'x': {
                            System.out.println("Gracias por jugar
Soliflips");
                            terminar = true;
                        }
                        break;

                        default: {
                            System.out.println("Error => No se
ingreso correctamente la opcion");
                            System.out.println("Ingrese A,B,C o X");

                        }
```

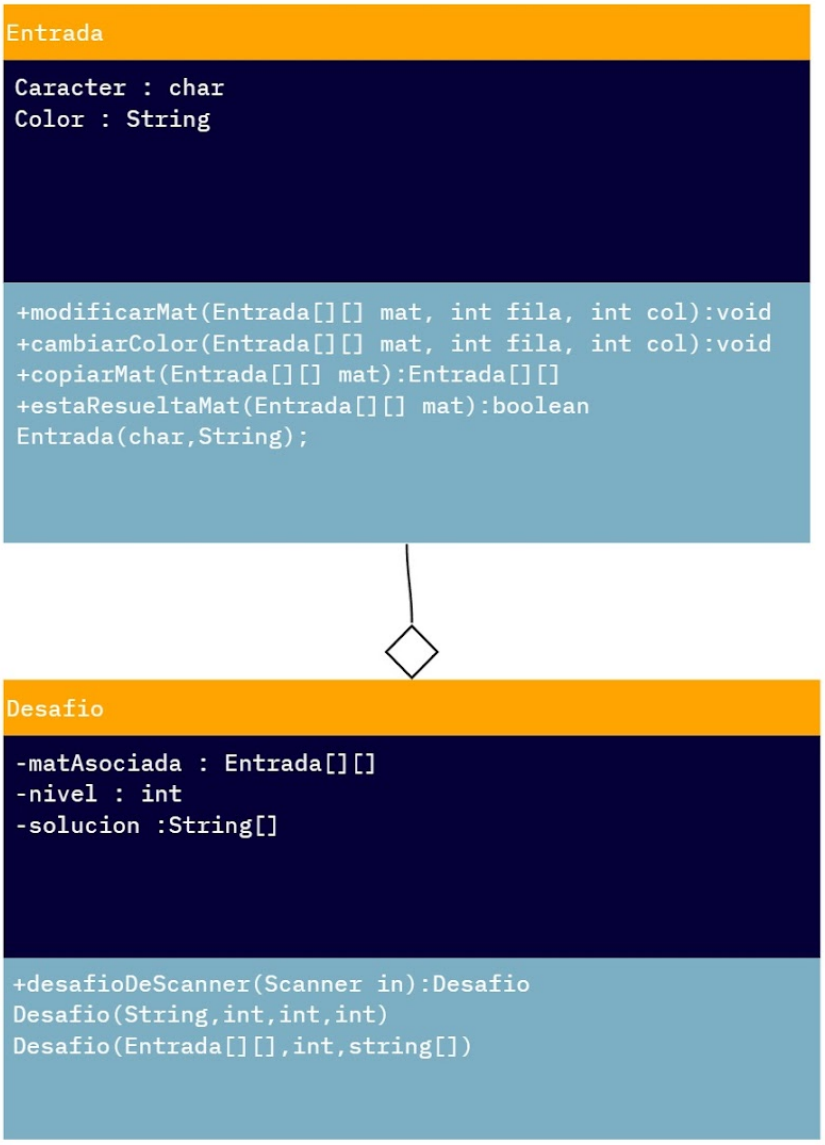
```
                    }

                } catch (Exception e) {
                    System.out.println("Error => No se ingreso
correctamente la opcion");
                    System.out.println("Ingrese nuevamente una
opcion (A,B,C o X)");
                    System.out.println();
                    System.out.println("A) => Crear partida tomando
datos del archivo 'datos.txt'");
                    System.out.println("B) => Crear partida usando
el tablero predefinido");
                    System.out.println("C) => Crear partida usando
un tablero al azar");
                    System.out.println("X) => Salir del juego");
                    System.out.println();
                    correcto = false;
                }
            }

        }
    }

}
```

Soliflips.java:

```
package Interfaz;

/**
 *
 * @author  Rodrigo Soca 327536
 */

import static Interfaz.Pantalla.mostrarTiempo;
import static Interfaz.Partida.jugar;
import java.util.Date;

/**
 *
 * @author socar
 */
public class Soliflips {

    public static void main(String[] args) {

        System.out.println("Rodrigo Soca Num: 327536");
        System.out.println();
        String asciiArt
                = "\033[31m"
                + "  _____  ____  _      _____ ______ _      _____
_____   _____ \n" + "\033[31m"
                + " / ____|/ __ \\| |    |_   _|  ____| |    |_   _|
__ \\ / ____|\n" + "\033[31m"
                + "| (___ | |  | | |      | | | |__  | |      | | |
|__) | (___  \n" + "\033[34m"
                + " \\___ \\| |  | | |      | | |  __| | |      | |
|  ___/ \\___ \\ \n" + "\033[34m"
                + " ____) | |__| | |____ _| |_| |    | |____ _| |_|
|     ____) |\n" + "\033[34m"
                + "|_____/ \\____/|______|_____|_|
|______|_____|_|    |_____/ "+"\u001B[0m";

        System.out.println(asciiArt);
```

```
        long tInicio= new Date().getTime();
        long tFin;
        long tTotal;
        jugar();
         tFin = new Date().getTime();
                                    tTotal = tFin - tInicio;
                                    System.out.println();
                                    System.out.print("Tu tiempo de
juego fue: ");
                                    mostrarTiempo(tTotal);
                                    System.out.println();
    }

}
```

Uml


Entrada
Caracter : char
Color : String
+modificarMat(Entrada[][] mat, int fila, int col):void
+cambiarColor(Entrada[][] mat, int fila, int col):void
+copiarMat(Entrada[][] mat):Entrada[][]
+estaResueltaMat(Entrada[][] mat):boolean
Entrada(char,String);
Desafio
-matAsociada : Entrada[][]
-nivel : int
-solucion :String[]
+desafioDeScanner(Scanner in):Desafio
Desafio(String,int,int,int)
Desafio(Entrada[][],int,string[])