TRABAJO PRACTICO FINAL DE PROGRAMACION“SUDOKU”

Jenifer Alvarado

Profesor: Luciano Diamand

Objetivo del trabajo:

El objetivo de dicho trabajo, es crear un sudoku, con el fin de poner en práctica los conocimientos adquiridos a lo largo del cursado, con la función de entender la implementación de los mismos en un trabajo más complejo.

Explicación del programa:

En la clase **SUDOKU** se importaron las siguientes librerías:

Import javax.swing.*: utilizada para la interfaz gráfica (cajas de textos, menú, JFrame,Jpanel y un cuadro de dialogo).

Import java.awt.*: utilizada para el color, tamaño y fuente del texto.

Import java.awt.event.*: utilizada para los eventos del teclado y JMenuBar.

Import java.io.IOException: utilizada para la Exception E/S de archivos.

Con extends herede de la clase JFrame e implemente ActionListener para los eventos del botón y del menu.

Declare variables para crear el menu: JMenuBar (creación del menu), JMenu (creación de la pestaña nivel), JMenuitem (opciones fácil, medio, difícil y solución), JMenuBar (crea la barra menu), panel y JFrame; también declare una matriz que va a contener los valores de cada tablero y por ultimo una variable que indicara los valores que tome la matriz según el nivel de dificultad que el usuario elija.

Constructor SUDOKU (): inicializa.

Método colorea (): recibe una matriz de tipo JTextField y se encargara de colorear las cajas de texto según el caso.

Método bloquear (): recibe una matriz de tipo JTextField y bloquea solo los valores iniciales de esa matriz.

Método panel (): diseño del panel (longitud, color, creación de cajas de texto) y llama al método eventosDelTeclado ().

Método menuBar (): inicializa JMenu, los items, lo agrega al JFrame y les agrega un evento con ActionListener.

Método actionPerformed (ActionEvent e) ->se aloja nuestro evento

- Si el usuario selecciona menuitem1 (EASY):
- Se podrá ver el panel.
- Se inicializa cada JTextfield con valores del tablero EASY.
- Se llama al método bloquear () y se bloquearan los valores iniciales.
- Se llama al método colorea () que colorea las cajas de texto según el caso.
- Y por último inicializa la variable nivelDificultad con valor a 1.

Si el usuario selecciona menuitem2 (MEDIO):

- Se podrá ver el panel
- Se inicializa cada JTextfield con valores del tablero MEDIO
- Se llama al método bloquear() y se bloquearan los valores iniciales
- Se llama al método colorea () que colorea las cajas de texto según el caso.
- Y por último se inicializa la variable nivelDificultad con valor a 2.

Si el usuario selecciona menuitem3 (DIFICIL):

- Se podrá ver el panel
- Se inicializa cada JTextfield con valores del tablero DIFICIL
- Se llama al método bloquear() y se bloquearan los valores iniciales
- Se llama al método colorea () que colorea las cajas de texto según el caso.
- Se inicializa la variable nivelDificultad con valor a 3.

Si el usuario selecciona menuitem4 (SOLUCION):

- Llamará a los método escribirArchivo () y abrirArchivo () de la clase TABLEROS para escribir y abrir las soluciones.

Método existeFila () recibe como argumento una matriz de tipo JTextField y dos enteros fila y columna, el valor que está en esa posición de fila y columna

se almacena en una variable llamada número. Luego declare una variable booleana que servirá como retorno.

Este método recorre la matriz pasada como argumento para encontrar valores de la fila que sean iguales a " numero" si esto es así entonces hay valores repetidos en esa posición, por lo tanto, la variable resultado toma como valor true, de lo contrario mantiene su valor inicial false .

Método existeColumna () recibe como argumento una matriz de tipo JTextField y dos enteros fila y columna, el valor que está en esa posición de fila y columna se almacena en una variable llamada número. Luego declare una variable booleana que servirá como retorno.

Este método recorre la matriz pasada como argumento para encontrar valores de la columna que sean iguales a " numero" si esto es así entonces hay valores repetidos en esa posición, por lo tanto, la variable resultado toma como valor true, de lo contrario mantiene su valor inicial false .

Método existe_caja (): recibe como argumento una matriz de tipo JTextField y dos enteros fila y columna, el valor que está en esa posición de fila y columna se almacena en una variable llamada número. Luego declare una variable booleana que servirá como retorno.

Luego declare cuatro enteros que van a tomar como valor el rango del cuadrado 3x3.

Con un bucle For recorre ese rango para encontrar valores del cuadrado 3x3 que sean iguales a " numero" si esto es así entonces hay valores repetidos en el cuadrado, por lo tanto, la variable resultado toma como valor true, de lo contrario mantiene su valor inicial false .

Método verif (): recibe como argumento la Matriz, la recorre con un bucle For y verifica si:

- El nivelDificultad es igual a 1 quiere decir que el usuario está jugando el nivel EASY entonces compara cada posición con la matriz que contiene la solución EASY, si las matrices son diferentes retorna false.

- El nivelDificultad es igual a 2 quiere decir que el usuario está jugando el nivel MEDIO entonces compara cada posición con la matriz que contiene la solución MEDIO, si las matrices son diferentes retorna false.
- El nivelDificultad es igual a 3 quiere decir que el usuario está jugando el nivel DIFICIL entonces compara cada posición con la matriz que contiene la solución DIFICIL, si las matrices son diferentes retorna false.

Método eventosDelTeclado (): este método recibe una matriz y dos enteros,

Contiene el **método KeyReleased (KeyEvent e)** que contiene el evento del teclado.

Declare una variable adicional llamada tf para que obtenga el valor que el usuario ingrese y lo pinte.

Si los valores que el usuario ingrese no están entre 1 y 9 tf toma como valor "" (vacío).

Este método también recorre la matriz y setea los espacios a 0.

Llama a los métodos:

Existe-fila (), existe_columna () existe-caja () si retornan true quiere decir que el valor que ingrese el usuario es erróneo por lo tanto se pintara de rojo en cada caso.

Método verif () si retorna true quiere decir que las matrices son iguales por lo tanto el jugador gano y muestra un cuadro de dialogo.

En la clase **TABLERO** se importaron las librerías:

Para archivos:

Import java.awt. Desktop;

Import java.io.BufferedWriter;

Import java.io.File;

Import java.io.FileWriter;

Import java.io.IOException;

Para recibir como argumento un JTextField:

Import javax.swing. JTextField;

Se crean las matrices EASY, MEDIO, DIFICIL, SOLEASY, SOLMEDIO, SOLDIFICIL, que van a contener los valores de los tableros del SUDOKY y sus soluciones.

Se declaran variables para la escritura del archivo.

Contiene los métodos:

IniciarValoresEASY (): recibe como argumento una matriz de tipo JTextfield que le cargara con un bucle For los valores contenidos en la matriz EASY.

IniciarValoresMEDIO (): recibe como argumento una matriz de tipo JTextfield que le cargara con un bucle For los valores contenidos en la matriz MEDIO.

IniciarValoresDIFICIL (): recibe como argumento una matriz de tipo JTextfield que le cargara con un bucle For los valores contenidos en la matriz DIFICIL.

EscribirArchivo (): que escribe un archivo txt con los valores de las matrices que contienen las soluciones.

AbrirArchivo (): abre el archivo escrito por el método **escribirArchivo ().**

Clase **JUGAR:** Es la clase que contiene el método main () que llama al constructor de la clase SUDOKU.

¿Cómo jugar?

Primero el usuario debe seleccionar un nivel EASY, MEDIO o DIFICIL.

Para jugar se debe seleccionar con el cursor cada cero del tablero para ingresar un valor.

En cualquier momento el usuario puede consultar las soluciones.