Paquete: modelo

Clase: MatrizSudoku

-Declaro e inicializo una matriz tipo char llamada tableroInicial, la inicializo con 9x9

-Declaro una variable de tipo booleana llamada privado

-Declaro e inicializo un ArrayList de tipo Char llamada matricesInternas

-Declaro e inicializo un ArrayList de tipo char llamada filas

-Declaro e inicializo un ArrayList de tipo char llamada columnas.

**Nombre:** +MatrizSudoku():MatrizSudoku

**Entradas**: no tiene

**Salidas**: No tiene

**Parámetros:** char[][] tableroInicial

**Retorno:** No tiene

**Descripción:** Es el constructor de la clase.

**Nombre**: +evaluarPosibilidadesSubMatriz(char[][] matrizInterna, int índiceMatrizInterna):char[]

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** char[][] matrizInterna, int índiceMatrizInterna

**Retorno:** Un arreglo tipo char con las posibilidades de la matriz interna

**Descripción:** Este método evalúa la matriz interna, recorrido las casillas de esta para así retornarnos un arreglo con los posibles numero que faltan dentro de la matriz interna para así ver como acomodarlos dentro de las casillas que se encuentran vacías.

**Nombre:** +valoresActuales(char[]o):char[]

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** char[] 0

**Retorno:** un arreglo de tipo char donde se almacenas los valores constantes del juego ósea los que ya están predeterminados según el juego y los que ya se encuentran seguros según observación

**Descripción:** Este método recorre el arreglo que recibe ´por parámetros para saber que números tiene en este momento con la posibilidad de saber cuáles faltan, este arreglo que llega por parámetros puede ser una fila, columna o matiz interna. Por lo tanto cuando nos retorna un arreglo nos retorna todas las opciones posibles para llenar las casillas restantes.

**Nombre:**+ Validar\_casilla (char[] fila, char[]col, char[] internacionales):char[]

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** char[] fila, char[]col, char[] internacionales

**Retorno:** un arreglo con todos los posibles números que pueden ir en una casilla especifica

**Descripción:** Este método lo que hace es validar las casillas que se encuentran libres y las posibilidades que tienen esa casilla con un número determinado.

**Nombre:**+ estaCompleto():boolean

**Entradas**: no tiene

**Salidas:** No tiene

**Parámetros**: no tiene

**Retorno:** Una variable booleana que nos dice si el sudoku esta completo

**Descripción:** Recorre toda la matriz buscando que en ella no exista el 0 o el espacio en blanco de esta manera podemos decir que la matriz esta completa y nos retona un true. En caso contrario nos retorna un false

**Nombre:**+estaCorrecto():boolean

Entradas: no tiene

**Salidas:** No tiene

**Parámetros**: no tiene

**Retorno:** una variable boolean que nos indica si la forma en la que se lleno esta completo

**Descripción:** Este método nos retorna un true si ningún valor de la fila, columna, o matriz interna se repite, en caso contrario pues nos retorna un false. De esta manera estamos seguro si la forma de llenarla estuvo bien.

**Nombre:**+tieneRepetidosCoF(in top): boolean

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** un entero que dice que si es 0 es para filas, si es 1 es para columnas

**Retorno**: una variable booleana que nos dice si la fila o la columna tienen números repetidos

**Descripción:** este método recorre o la fila o la columna en busca de numero repetidos, en caso tal de que encuentre uno pues nos devuelve un false, sino nos retorna un true.

**Nombre:**+tieneRepetidoEnSub\_Matrices ():boolean

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** no tiene

**Retorno:** una variable boolean que nos indica si las matrices internas tienen algún numero repetido

**Descripción:** Este método lo que hace es recorrer las matrices internas en busca de un numero repetido dentro de ella, tenemos 9 matrices internas. Entonces recorremos cada una de ellas, si encontramos algún numero repetido entonces retorna false, encaso contrario retorna true.

Nombre:+sacarEnArreglo(int opción):void

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros**: tiene un entero, por medio de este entero le diremos si es 1 entonces que saque en un arreglo las columnas y si es 0 que saque en un arreglo las filas

**Retorno:** no tiene

**Descripción**: Este método sirve para sacar en arreglos cada una de las filas o columnas (depende de la opcion dada) y se almacenan en las listas que le corresponde.

**Nombre:**+tieneRepetidos(char arreglo):boolean

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** Un arreglo que contiene valores de la matriz

**Retorno:** una variable booleana que nos indica si tiene repetido

**Descripción:** Este método sirve para revisar si un arreglo tiene o no valores repetidos dentro de este.

**Nombre:**+tieneRepetidosMatrizIterna(char[][] matrizInterna):boolean

**Entradas**: no tiene

**Salidas:** No tiene

**Parámetros**: LA matriz interna (3x3)

**Retorno:** una variable booleana indicándonos si la matriz interna tiene elementos repetidos

**Descripción:** Este método valida que la matriz interna no tenga elementos repetidos, en caso de tener nos retorna un false

**Nombre:**+obtenerMatricesInternas():void

**Entradas:** no tiene

**Salidas**: No tiene

**Parámetros**: No tiene

**Retorno:** no tiene

**Descripción:** Este método nos ayuda a obtener las matrices internas del sudoku, estas matrices a a su vez son almacenas en un ArrayList llamado matricesInternas

**Nombre:**+sacarEnArreglo(int opción):void

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** tiene un entero, por medio de este entero le diremos si es 1 entonces que saque en un arreglo las columnas y si es 0 que saque en un arreglo las filas

**Retorno**: no tiene

**Descripción:** Este método sirve para sacar en arreglos cada una de las filas o columnas (depende de la opcion dada) y se almacenan en las listas que le corresponde.

**Nombre**:+imprimirTablero():void

**Entradas**: no tiene

**Salidas:** No tiene

**Parámetros:** no tiene

**Retorno:** no tiene

**Descripción:** Este método nos imprime el tablero del sudoku

**Nombre:**+imprimirFilasOCOlumnas (int op):void

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** tiene un entero, por medio de este entero le diremos si es 1 entonces que imprima las columnas o y si es 0 que imprima las filas

**Retorno:** no tiene

**Descripción:** Este método nos imprime el contenido de las columnas o las filas según lo que le entre por parámetros

**Clase: MainClass**

**-**Declaro una variable estática de tipo MatrizSudoku llamada tablero

-Declaro e inicializo una matriz estática de tipo caracteres llamada sudoku con una dimensión de 9x9

-Declaro e incializo el lector de tipo Scanner

**Nombre:+main** (String [] args):void

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** String [] args

**Retorno:** no tiene

**Descripción:** Este método es el main, dentro del main se llama a l método que lee archivo donde se encuentra la matriz. También se inicializa la variable tablero.

**Nombre:+leerSudoku (**String tableroSudoku, int indiceFila):void

**Entradas:** no tiene

**Salidas:** No tiene

**Parámetros:** String tableroSudoku, int indiceFila

**Retorno:** no tiene

**Descripción:** Este método lo que hace es leer el archivo aparte que contiene la matriz, los datos ingresados en este archivo son separados por comas, por ende con un Split se separan los valores para generar el sudoku