Proyecto Final de Programación Orientada a Objetos (4)

Juego Sudoku

Jorge Mario Salazar Marín

Estudiante de Ingeniería de Sistemas y Computación

1.114.402.197

Juan José Vera Arango

Estudiante de Ingeniería de Sistemas y Computación

1.087.559.573

Universidad Tecnológica de Pereira

26 de Mayo de 2016

Pereira – Risaralda

TABLA DE CONTENIDO

[INTRODUCCIÓN 3](#_Toc451965868)

[OBJETIVOS 4](#_Toc451965869)

[Objetivo general. 4](#_Toc451965870)

[Objetivos específicos. 4](#_Toc451965871)

[MARCO TEORICO 5](#_Toc451965872)

[APLICATIVO SUDOKU 6](#_Toc451965873)

[¿Cómo Lo Hicimos? 6](#_Toc451965874)

[MODELO SUDOKU TRADICIONAL 7](#_Toc451965875)

[Clase SudokuParser.java: 7](#_Toc451965876)

[Clase CrearSudoku.java: 7](#_Toc451965877)

[ Clase SudokuFacil.java : 7](#_Toc451965878)

[ Clase SudokuMedio.java : 8](#_Toc451965879)

[ Clase SudokuDificil.java : 8](#_Toc451965880)

[Clase SolvedBacktracking.java : 8](#_Toc451965881)

[Clase Sudoku.java. 9](#_Toc451965882)

[Clase Cell.java : 9](#_Toc451965883)

[MODELO SUDOKU SERPENTEANTE 10](#_Toc451965884)

[Clase SolvedS.java: 10](#_Toc451965885)

[Clase SudokuS.java : 11](#_Toc451965886)

[Clase Cell.java : 11](#_Toc451965887)

[VISTA Y CONTROLADOR (SUDOKU TRADICIONAL) 12](#_Toc451965888)

[Controlador: 12](#_Toc451965889)

[Vista: 13](#_Toc451965890)

[INTERFAZ GRÁFICA (SUDOKU TRADICIONAL) 13](#_Toc451965891)

[VISTA Y CONTROLADOR (SUDOKU SERPENTEANTE) 14](#_Toc451965892)

[INTERFAZ GRÁFICA (SUDOKU SERPENTEANTE) 14](#_Toc451965893)

[VISTAS UTILIZADAS PARA LA INTERACCIÓN CON EL USUARIO …………………….15](#_Toc451965894)

DIAGRAMA DE CLASES………………………………………………………………………16

REFERENCIAS………………………………………………………………………………….17

# INTRODUCCIÓN

Sudoku es un juego matemático que se publicó por primera vez a finales de la década de 1970 y se popularizo en Japón en 1986, dándose a conocer en el ámbito internacional en 2005, cuando numerosos periódicos empezaron a publicarlo en su sección de pasatiempos. El objetivo del sudoku es rellenar una cuadricula de 9 x 9 celdas, dividida en subcuadrículas de 3 x 3, con las cifras del 1 al 9 partiendo de algunos números ya dispuestos en algunas celdas.  De alguna forma el Sudoku se basa en la búsqueda de la combinación numérica perfecta. Hay diferentes niveles de dificultad y la resolución del problema requiere paciencia y ciertas dotes lógicas. Profesores de todo el mundo lo recomiendan como método para desarrollar el razonamiento lógico. Enseñar a los niños a jugar sudoku, es una buena alternativa para estimular y potenciar sus habilidades matemáticas, de lógica y pensamiento crítico.

El juego del sudoku se popularizo mundialmente por medio de los periódicos y revistas de entretenimiento, quienes buscaban que sus lectores tuvieran una sección específicamente para el entretenimiento, pero no contaban con el éxito que tuvo finalmente este juego de habilidad matemática, porque en realidad sudoku no es juego solo de entretenimiento, sudoku es considerado un juego matemático que para ser resuelto requiere de cierta habilidad mental. El juego del sudoku ha tenido diferentes variantes desde su nacimiento , pero no pierde la esencia de ser un juego de lógica matemática , por otro lado las ciencias de la computación han tomado al juego del sudoku como un buen ejercicio para el aprendizaje de ciertos algoritmos de programación de software, lo que nos lleva a ver lo importante que ha sido el juego del sudoku para el mundo en general, no solo para los que buscan un reto y entretenimiento , sino también para que ellos que pertenecen a la academia y quienes día a día se esfuerzan por encontrar el conocimiento.

# OBJETIVOS

## Objetivo general.

Diseñar y realizar un aplicativo para jugar el juego sudoku.

## Objetivos específicos.

* Hacer que el sudoku pueda leer su configuración inicial desde archivo.
* Generar sudokus con varios niveles de dificultad (Fácil, Normal, Difícil).
* Mostrar y ocultar los candidatos de cada una de las celdas no resueltas del sudoku.
* Solucionar el sudoku haciendo uso de 2 algoritmos de solución (Backtracking y propagación de restricciones).
* Diseñar una interfaz gráfica agradable e intuitiva para el usuario.
* Realizar un modo de juego adicional para el sudoku (Sudoku Serpenteante).
* Permitir al usuario guardar su partida actual y posteriormente cargarla si así se desea.

# MARCO TEORICO

Los rompecabezas Sudoku son, en realidad, un caso especial de los Cuadrados Latinos; cualquier solución de un Sudoku es un Cuadrado Latino. El Sudoku impone la restricción adicional de que los subgrupos de 3x3 deben contener también los dígitos 1-9 (en la versión estándar).  
  
Este tipo particular de rompecabezas, tal y como lo conocemos, se publicó por primera vez al final de la década de los 70 en la revista "Math Puzzles and Logic Problems"(Rompecabezas Matemáticos y Problemas Lógicos) de Dell Magazines. El nombre que Dell le dio a estos rompecabezas fue "Number Place", que es como dicha compañía los ha seguido nombrando hasta hoy. Dell tomó el concepto de Cuadrado Latino de Euler y lo aplicó a una cuadricula de 9x9 con la adición de nueve subcuadriculas, cada una de las cuales contiene todos los números del 1 al 9.   
  
Como vemos, el concepto de Sudoku no se inventó en Japón, como mucha gente cree, pero sí que lo fue el nombre. En 1984, Nikoli, la compañía líder en creación de puzzles de Japón, descubrió los "Number Place" de Dell y decidió presentarlos a los fans japoneses de los rompecabezas. Los rompecabezas, que inicialmente se llamaron Suuji Wa Dokushin Ni Kagiru ("los números deben ser únicos" o "los números deben existir sólo una vez") pronto se volvieron muy populares.   
  
En 1986, después de añadir muchas mejoras, sobretodo creando patrones simétricos y reduciendo el número de pistas que se daban, el sudoku llegó a ser una de los rompecabezas más vendidos en Japón. Cuando se dio cuenta que uno de los principales problemas del Sudoku era su largo nombre, Kaji Maki, el presidente de Nikoli, lo abrevió a Sudoku (Su = número, dígito; Doku = único, soltero). Hoy en día hay más de 600.000 copias de revistas de Sudoku publicadas mensualmente en Japón. En todo ese tiempo, prácticamente nadie en Europa conocía o había prestado atención a los Sudokus.

El Sudoku ha aparecido en los periódicos de todo el mundo y se le describe en los medios de todo el mundo como "el cubo Rubik del siglo XXI" y como el "rompecabezas que más rápido ha crecido del mundo".

# APLICATIVO SUDOKU

## ¿Cómo Lo Hicimos?

El aplicativo fue desarrollado en paralelo por los dos integrantes del equipo, en el lenguaje de programación orientado objetos JAVA en el IDE Netbeans; JAVA que es un lenguaje de programación que nos proporciona una mejor organización de nuestro código, ya que se utilizaron patrones de diseño para poder obtener una programación más óptima y entendible; el concepto de patrones de diseño fue muy importante en el desarrollo de este aplicativo ya que con ellos se consiguió una programación un poco más sencilla y óptima. Requeríamos realizar un aplicativo con dos modos de juego , un modo de juego tradicional y otro modo de juego llamado Sudoku Serpenteante , por lo que se decidió inicialmente crear dos modelos de sudoku uno para cada modo de juego, idea que nos ayudaría a manejar una mejor programación en paralelo de los integrantes del grupo, porque mientras uno de ellos trabajaba en el modelo de sudoku tradicional , el otro podía trabajar en el modelo del sudoku Serpenteante, dicha estrategia nos ayudó a optimizar mucho el tiempo de trabajo en el proyecto. Posteriormente se observó que los dos modelos no eran muy distintos y que se podían juntar en un solo modelo, pero no se optó por esa opción por la razón de evitar errores y no saber en cuál de los modelos estaba ocurriendo, por lo que finalmente se deicidio dejar los modelos de los dos modos de juego separados; primero debíamos de construir los modelos de los dos sudokus de manera que tal que cumplieran las reglas tradicionales de los sudokus, reglas que veremos a continuación:

Para los dos modos de juego se tienen 81 celdas, que para el sudoku tradicional están distribuidas en regiones de 3x3 y para el serpenteante están distribuidas con una forma ingresada por defecto pero que no es simétrica.

Para el sudoku en modo Tradicional, se deben insertar los números del 1 al 9 de la siguiente manera:

* Los números no se pueden repetir en la misma Fila.
* Los números no se pueden repetir en la misma Columna.
* Los números no se pueden repetir en la subcuadricula en la que se van a insertar, para el sudoku tradicional dicha subcuadricula es de 3x3, nosotros la llamaremos región.

Para el sudoku en modo Serpenteante, se deben insertar los números del 1 al 9 de la siguiente manera:

* Los números no se pueden repetir en la misma Fila.
* Los números no se pueden repetir en la misma Columna.
* Los números no se pueden repetir en la región en la que se van a insertar, para el sudoku serpenteante esta región no es simétrica.

# MODELO SUDOKU TRADICIONAL

El modelo del sudoku tradicional cuenta con 8 clases, cada cumple con una función específica, función que lleva a tener un buen funcionamiento del aplicativo, bajo los requerimientos de funcionamiento que se tienen inicialmente, a continuación veremos una breve explicación del funcionamiento de cada una de las clases de este modelo:

Clase SudokuParser.java: esta clase es la encargada de leer por archivo una configuración inicial si desea, el archivo debe contar con unas características especiales, para poder ser leído correctamente por esta clase, primero debe llevar un número que es el que determina cual es el tamaño del sudoku (para nosotros seria 9), y acto seguido van los números a insertar en la cuadricula, teniendo en cuenta que van en una sola fila y que los números (0) se reemplazan por un simple ( . ), todo esto debe de ir escrito en un archivo con extensión .txt, si el archivo cumple con las características anteriores , este podrá ser leído correctamente.

Clase CrearSudoku.java: esta clase va de tres clases más, que serían las clases que determinan la dificultad del sudoku a crear:

* Clase SudokuFacil.java : esta clase es la encargada de generar sudokus de dificultad fácil, que al ser llamada por la clase CrearSudoku.java, deberá generar aleatoriamente una plantilla de números que conforman un sudoku , que tiene una única solución, el algoritmo que nos permite generar un sudoku de este tipo , consiste en partir de un sudoku previamente solucionado y aleatoriamente ir quitando números de dicha solución , que para el caso del sudoku fácil , se deben quitar números en un rango de 35 a 30 numero , y adicionalmente para tener una mayor aleatoriedad, se aplica un algoritmo que consiste en correr números de filas y columnas , sin afectar el sudoku , para que siga teniendo una única solución.
* Clase SudokuMedio.java : esta clase es la encargada de generar sudokus de dificultad media, que de igual manera que la clase anterior también es llamada por la clase CrearSudoku.java, generara aleatoriamente una plantilla de números para un sudoku con dicha dificultad , con una solución única , el algoritmo para generar un sudoku de dificultad media , es exactamente igual al algoritmo de generación de sudokus fáciles , solo que el rango de números que se deben quitar va de 50 a 36, de igual manera que el algoritmo para correr números de filas y columnas , para lograr una mayor aleatoriedad, sin afectar la única solución.
* Clase SudokuDificil.java : esta clase es la encargada de generar sudokus de dificultad difícil, que de igual forma que las dos clases anteriores es llamada por la clase CrearSudoku.java, y generara aleatoriamente una plantilla de números que formaran un sudoku , con una única solución, el algoritmo utilizado para obtener estos números es igual al de las dos clases anteriores , a excepción de que el rango de a quitar para esta dificultad va de 63 a 50, de la misma forma también se aplican el algoritmo de correr números en filas y columnas sin afectar , la solución única.

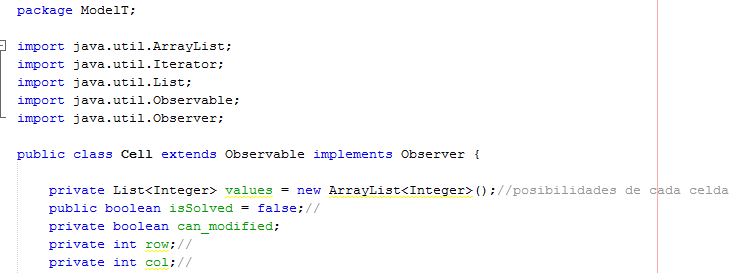
Adicionalmente cada una de las clases para generar sudokus de diferentes dificultades, hacen uso del patrón de diseño estrategia, que se plantea como una interfaz que después va a ser implementada por cada una de las clases de dificultad, la interfaz estrategia tiene como clase abstracta a CrearSudoku.java, y de esta manera se conectan cada una de las clases para generar un sudoku con determinada dificultad, con el patrón estrategia.

Clase SolvedBacktracking.java :Esta clase es la encargada de resolver el sudoku en un momento determinado por el usuario, dicha clase resolverá el sudoku von el algoritmo de BackTracking (vuelta atrás), el cual consiste en ir ingresando números candidatos en las casillas no resueltas, inicialmente el algoritmo empezara por la primer casilla vacía e ingresara el primer candidato de esa casilla , y buscara la siguiente casilla vacía y hará lo mismo , con cada una de las casillas, en caso de llegar a una casilla resuelta y no tener candidatos para insertar, el algoritmo deberá volver a la casilla en donde empezó a ejecutarse e insertar no el primer candidato sino el segundo, y hará eso sucesivamente hasta encontrar una solución , y en caso de que el algoritmo explore todas las posibilidades y no encuentre una solución podemos concluir que el sudoku no tiene solución.

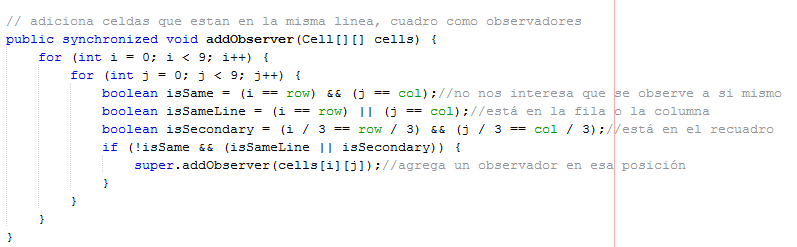
## Clase Sudoku.java : Esta clase es la que nos ayudara a conectar el modelo del sudoku tradicional , con su vista y controlador , ya que esta clase es la que me ayuda a inicializar las celdas del sudoku con sus candidatos, además de decirme si una celda esta resulta o no, también me insertara valores al modelo del sudoku y con ayuda de la clase Cell.java , actualizara candidatos y valores de las casillas, sin desestimar las restricciones de inserción en el modelo del sudoku, por lo dicho entonces es válido pensar que la clase Sudoku.java y la clase Cell.java trabajan conjuntamente , ya que estas dos clases son las encargadas de que el modelo del sudoku tradicional funcione de manera correcta y eficiente, se utilizó el patrón de diseño singleton (instancia única), para crear un nuevo sudoku con el tamaño requerido (para nuestro caso el tamaño será de 9), por lo que el sudoku solo se instanciara una única vez , y en caso de requerir un nuevo sudoku , podremos acceder a la única instancia que se crea por medio de un método get, así pues Sudoku.java nos servirá como puerta para interactuar con la clase Cell.java y poder controlar dicho modelo.

Clase Cell.java :Esta clase es considerada por nosotros como la clase principal del modelo del sudoku tradicional, ya que es la encarga de hacer cumplir las restricciones de juego del sudoku en cada celda , y lo hace por medio del patrón Observador(observer), el cual tiene las tres restricciones del juego del sudoku, y luego de que nosotros con ayuda de las restricciones agreguemos a cada una de las celdas los observadores , nos permitirá actualizar las listas de candidatos de cada una de las celdas, de allí la clasificación de esta clase como la principal, ya que sin esta clase el aplicativo no sería un juego de sudoku, a continuación una muestra de la clase Cell.java , de la parte del observador y de la parte en donde se agregan los observadores:

Definición de la clase Cell.java , con sus atributos , y se define extendiendo a observable e implemetando la interfaz Observer :



En esta parte de la clase se están agregando las restricciones para poder agregar los observadores a cada una de las celdas:



# MODELO SUDOKU SERPENTEANTE

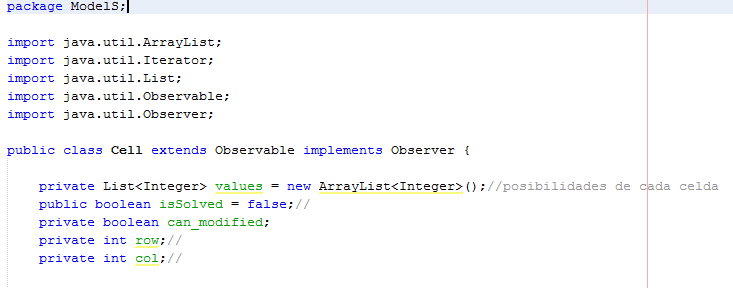
El modelo del sudoku serpenteante cuenta con 3 clases, ya que en este tipo de sudoku solo requeríamos que el usuario interactuara con este modo de juego insertando números a base de candidatos y teniendo en cuenta las restricciones para el sudoku serpenteante, adicional a eso agregamos una clase para resolver el sudoku:

Clase SolvedS.java: Esta clase tiene como objetivo resolver un sudoku serpenteante, haciendo uso del algoritmo backtracking previamente explicado, no varía mucho el backtracking del sudoku tradicional al del sudoku serpenteante, ya que los dos insertan números a base de los candidatos de cada una de las celdas, esta clase fue implementada opcionalmente ya que no se requería que este tipo de sudoku tuviera un botón para solucionarse.

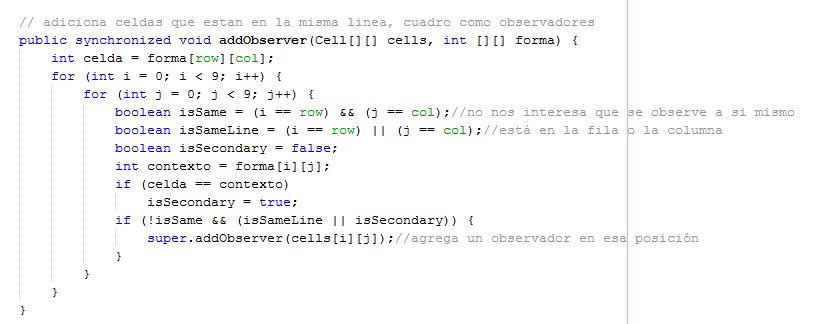
Clase SudokuS.java :Esta clase es la que nos ayuda a inicializar el sudoku, y trabaja de la mano con la clase Cell.java , y además es la que conecta el modelo del sudoku serpenteante con su respectivo controlador y vista, de allí la importancia de esta clase, esta clase cuenta con la implementación del patrón de diseño Singleton (Instancia Única), por lo cual solo instanciamos una única el sudoku, y en caso de necesitar un nuevo sudoku , hacemos uso de un método get que nos atrapa la única instancia que se crea en el modelo; Se tiene un método llamado Setup, que es el encargado de inicializar el sudoku con todas sus características , con los candidatos en cada una de sus celdas y además de todos los otros atributos que ayudan a normal funcionamiento del modelo, además el método Setup recibe como parámetro no solo el tamaño del cual vamos a iniciar el sudoku, sino también la forma que debe tener este sudoku para que sea un sudoku serpenteante.

Clase Cell.java : Esta clase al igual que la clase Cell.java del sudoku tradicional, es considerada por nosotros como la clase principal del Sudoku Serpenteante, ya que en ella es que se aplican todas las restricciones del sudoku , para que el aplicativo sea un juego de sudoku , por lo que esta clase contiene el patrón de diseño Observador , que es el que le va ayudar a cada celda a saber que números se están ingresando en su misma fila , columna o este caso forma de región, aparte de contener varios métodos que me ayudan a saber varias características acerca de cada una de las celdas del tablero, a continuación veremos cómo se define esta clase y como se están agregando los observadores a cada una de las celdas:

Definición de la clase celda para el modelo del sudoku serpenteante con sus respectivos atributos:



Código en donde se están agregando los observadores de cada una de las celdas del tablero , teniendo en cuenta las restricciones del juego del sudoku serpenteante, se puede observar como se le ingresa como parámetro no solo la matriz de celdas , sino también la matriz que contiene la forma del sudoku:



# VISTA Y CONTROLADOR (SUDOKU TRADICIONAL)

Controlador:el controlador para el sudoku tradicional, es el encargado de conectar el modelo con la vista del sudoku tradicional, por lo que contiene una serie de métodos y botones que le facilitan la interacción al usuario, con la parte gráfica. Los usuarios por medio de la parte grafica deberán ingresar los números de fila y columna y adicionalmente el número que desean insertar al sudoku, y acto seguido presionar el botón de insertar, los números serán capturados por campos de texto (Text Field), que luego por medio del controlador serán procesados, enviados al modelo y posteriormente el usuario debe recibir una respuesta en la interfaz. Se cuenta también con un botón para resolver el sudoku, pero que primero requiere que este presionado uno de los botones que contiene el nombre del algoritmo con que quiere que se resuelva el sudoku (Toggle Button), y dependiendo de que botón se presiona el controlador sabrá a cuál de las dos clases de resolver llamara, clases que se encuentran en el modelo del sudoku tradicional. De igual manera existe un botón para generar sudokus con su respectiva dificultad, que al igual que el botón para resolver también debe estar presionado alguno de los botones de dificultad para que el controlador pueda detectar, a cuál de las clases de generar puede llamar haciendo uso el modelo del patrón estrategia. Se cuenta con los botones para guardar y cargar archivos, el controlador detecta cuál de los dos es presionado y llama a su respectiva clase en el modelo. Así como los botones nombrados anteriormente existen otros que nos ayudan a tener una interfaz gráfica entendible para el usuario, pero en general el controlador del sudoku tradicional nos sirve para conectar nuestra parte grafica con el modelo del respectivo modo de juego de sudoku.

Vista:la vista es la encargada de capturar todos los datos del usuario, que posteriormente serán procesados entre el controlador y el modelo. Para la interfaz gráfica utilizamos una herramienta llamada Scene Builder (Constructor De Escenas), que nos permitió crear una interfaz gráfica agradable para el usuario, además de ser intuitiva, dicha interfaz contiene botones para facilitar la interacción usuario y aplicación, además de ser visualmente fácil de entender.

La vista es un componente demasiado importante en el aplicativo ya que por medio de ella es que el programa está capturando todos los datos que serán procesados en el modelo y ser viran para que nuestro aplicativo sea un juego de sudoku tradicional, a continuación veremos una muestra de cómo es la interfaz gráfica para el sudoku Tradicional:

# INTERFAZ GRÁFICA (SUDOKU TRADICIONAL)

Esta Interfaz cuenta con lo necesario para que el usuario juegue Sudoku Tradicional , además de tener botones para generar nuevas dificultades , botones para resolver el sudoku y guardar y cagar datos:

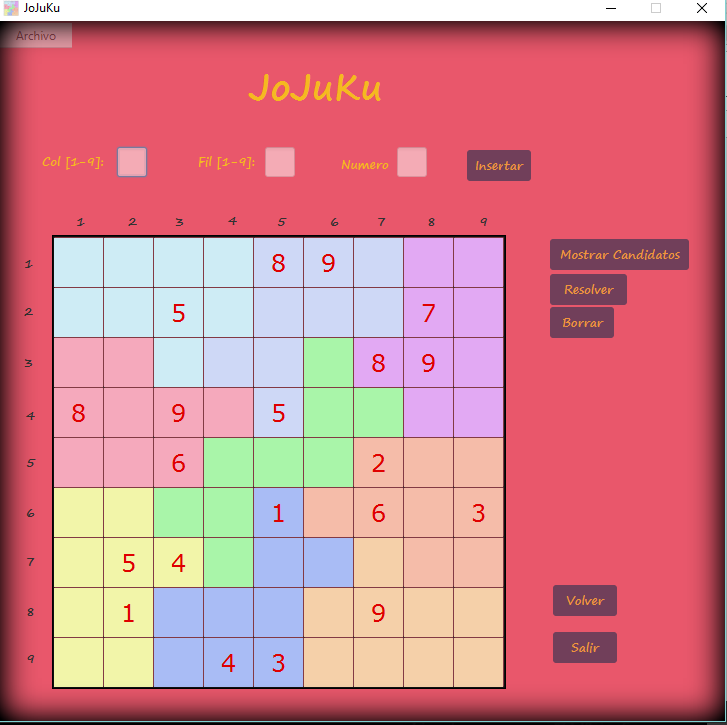


# VISTA Y CONTROLADOR (SUDOKU SERPENTEANTE)

La vista y el controlador para el sudoku serpenteante funcionan de igual manera que el sudoku tradicional, a continuación veremos la interfaz gráfica para el sudoku Serpenteante:

# INTERFAZ GRÁFICA (SUDOKU SERPENTEANTE)

Este caso la interfaz cuenta solo con los elementos necesarios para que el usuario juegue sudoku serpenteante, además de tener un botón de resolver:



# VISTAS UTILIZADAS PARA LA INTERACCIÓN CON EL USUARIO

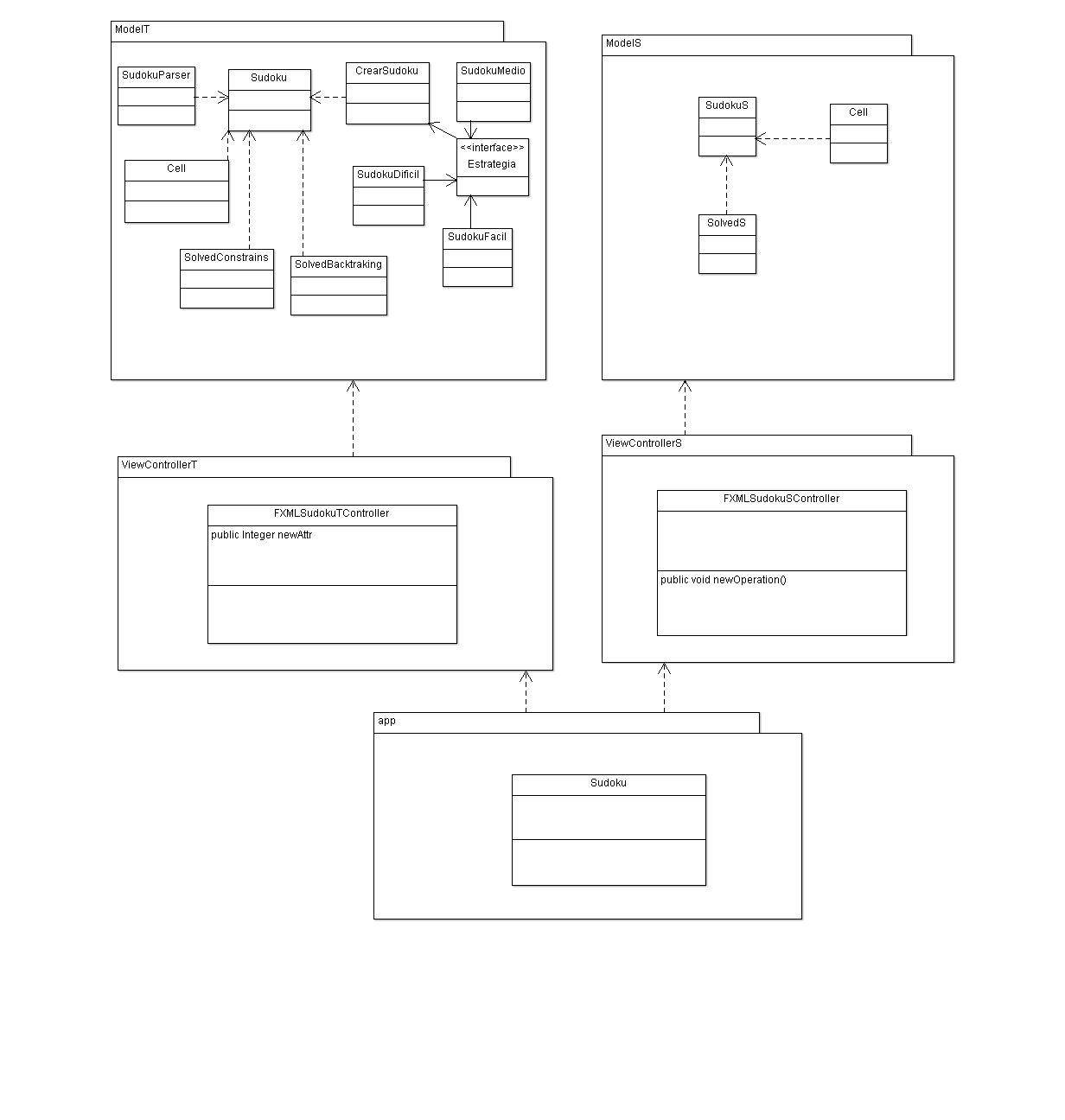
Vista de inicio del juego sudoku, en ella inicamos un nuevo juego y nos lanzara la siguiente vista:



Vista para elegir el modo de juego, que puede ser sudoku tradicional o serpenteante:



# DIAGRAMA DE CLASES



# REFERENCIAS BIBLIOGRAFICAS

<http://www.uaq.mx/ingenieria/publicaciones/eureka/n24/camepa01.pdf>

<http://norvig.com/sudoku.html>

<https://es.wikipedia.org/wiki/Sudoku>

<http://platea.pntic.mec.es/~ablanco/gi/tecnicasudoku.htm>

<https://sourceforge.net/projects/jacop-solver/>