**SUDOKU**

|  |  |  |
| --- | --- | --- |
| *Resumen*Este documento presenta los resultados obtenidos en el desarrollo del proyecto final de programación 4, en el cual se requería el diseño de un juego llamado sudoku, el cual debía de tener dos modos de juego, un modo de juego llamado sudoku tradicional y uno llamado sudoku serpenteante, dicho juego debe ser solucionado por los algoritmos de backtracking y propagación de restricciones, adicionalmente la interfaz gráfica del proyecto debe ser entendible para el usuario e intuitiva.*Palabras claves—*Backtracking, Restricciones, Algoritmo, Sudoku. |  | **JORGE MARIO SALAZAR MARÍN**  Estudiante de Ingeniería Sistemas  Universidad Tecnológica de Pereira  [jorgemsm12316@gmail.com](mailto:harold.cardona616@gmail.com)  **JUAN JOSE VERA ARANGO**  Estudiante de Ingeniería en Sistemas  Universidad Tecnológica de Pereira  jjvera96@gmail.com |

INTRODUCCIÓN

Sudoku es un juego matemático que se publicó por primera vez a finales de la década de 1970, dándose a conocer en el ámbito internacional en 2005 cuando numerosos periódicos empezaron a publicarlo en su sección de pasatiempos; en el ámbito educativo el sudoku forma parte de un juego muy didáctico a la hora del aprendizaje de la algoritmia, ya que en informática realizar algoritmos para la solución de sudokus, es un buen ejercicio para el buen aprendizaje de la programación por restricciones.

METODOLOGÍA

Este proyecto fue desarrollado en un lenguaje de programación orientado a objetos llamado JAVA en el IDE NetBeans , el proyecto debía de cumplir una serie de características específicas para su funcionamiento, las cuales fueron desarrolladas por los dos integrantes del grupo , cada uno de ellos era encargado de realizar determinadas tareas dentro del proyecto.

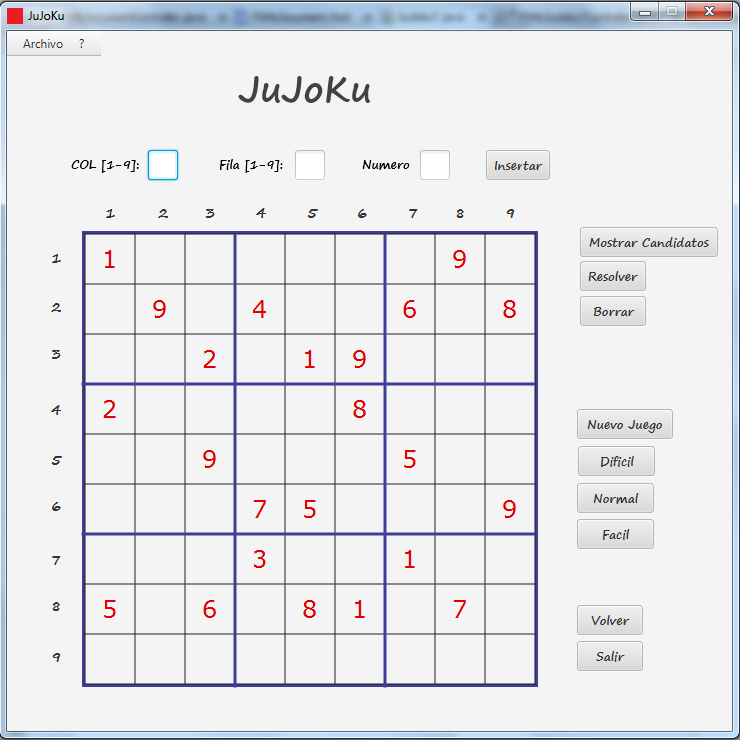
DESARROLLO

Primero se realizó la implementación del modelo para el sudoku clásico o tradicional, el cual fue desarrollado con ayuda de lo realizado en clase y complementado por los patrones de diseño observador e iterador, el patrón observador fue el encargado de calcular los candidatos de cada casilla después de insertar un numero en el sudoku , ya que como cada casilla observa a cada una de las casillas de la misma fila columna y región, y el iterador es el encargado de retornar de ser necesario cada uno de los valores candidatos de cada casilla del tablero del sudoku , de igual manera fueron implementados estos dos patrones para el sudoku serpenteante , ya que las casillas debían de observarse de igual forma con las casillas de la fila y la columna , solo que para este caso la región no iba a ser calculada de la misma manera que en el sudoku tradicional , sino que iba a varias dependiendo del patrón que tuvieran las regiones.

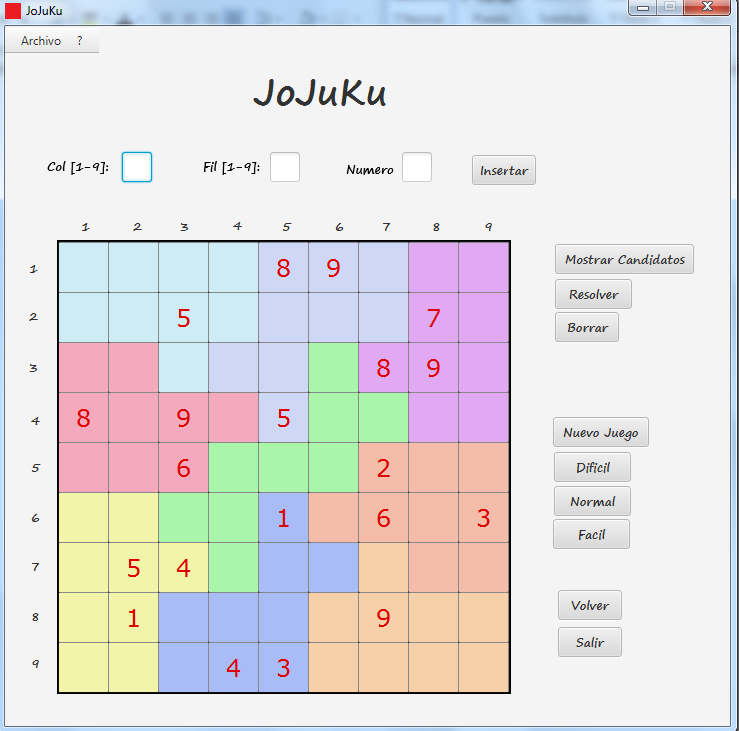
De igual manera fue implementado el patrón estrategia en el proyecto , dicho patrón es el encargado de generar sudokus con su respectiva dificultad , la cual puede ser Fácil , Medio o Difícil , con el fin de que el usuario escoja la dificultad en la cual quiere realizar el sudoku, además de esto el usuario puede guardar su progreso en el juego , para después si es requerido ser cargada dicha grabación, esto fue implementado en dos clases, la primera de ellas es la que se encarga de leer por archivo un sudoku y la otra clase es la que se encarga de guardar en un archivo el sudoku que ha generado el usuario , a medida que ha ido jugando.

El sudoku clásico requería ser solucionado por medio de dos algoritmos , el primero de ellos es el algoritmo de backtracking (vuelta atrás), el cual consistía en ir insertando candidatos únicos en cada una de las casillas no resueltas del tablero del sudoku , y en caso de no contar con casillas de únicos candidatos , el algoritmo debe escoger la casilla con menos candidatos del tablero e insertar uno de ellos y en base a esa inserción se calculan de nuevo los candidatos de todas las casillas , y así sucesivamente ir insertando candidatos en cada celda y en caso de no llegar a una solución , el algoritmo debe estar en la capacidad de volver atrás . Hasta el lugar en donde se encontró la casilla con menos candidatos del tablero e insertar el siguiente candidato, e insertar candidatos nuevamente, todo esto hasta encontrar una solución para dicho sudoku y el segundo algoritmo es igual al backtracking dolo que los candidatos van a tener una serie de restricciones para ser insertados.

Otro de los patrones de diseño utilizado en el proyecto fue el MVC (Modelo Vista Controlador), con la ayuda de este se conectaron los modelos de los dos sudokus , con su respectivo controlador , para después ser enviados los datos del controlador a la vista de cada sudoku , la vista fue diseñada con ayuda del Scene Builder , el cual es una herramienta que podemos utilizar para construir la interfaz gráfica de cada uno de los modos de juego de los sudokus , además de cada una de las ventanas para hacer más didáctico y entendible el programa de sudoku, a continuación se muestra la interfaz gráfica de cada uno de los modos de juego del sudoku.

Interfaz del Sudoku Tradicional:

Interfaz Para el Sudoku Serpenteante:



DIFICULTADES

Las dificultades que se tuvo frente a desarrollar este proyecto fueron:

* La poca documentación de JavaFX, debido a esto se perdió mucho tiempo explorando el mismo.
* La decisión de que patrones de diseño usar en el sudoku.
* La implementación de los algoritmos para la solución de los sudokus.
* Ingeniarse o buscar un algoritmo para la creación de sudokus con única solución.
* La integración del MVC.
* La implementación del patrón de diseño Factory Method.

CONCLUSION

La conclusión del trabajo anterior fue la gran utilidad que tiene los patrones de diseño para ayudar a la solución de diferentes problemas, que unos son más específicos para ciertos problemas. Además de que el lenguaje de programación Java en el cual muchas cosas necesarias para el diseño de programas ya están creadas y son muy fáciles de usar.

También nos deja motivados para continuar en el proyecto y sacarle una versión en las distintas plataformas con el fin de seguir explorando JavaFx.