INFORME DEL PROYECTO II PROGRAMACION DECLARATIVA

Lidier Robaina, Leandro Hernandez (C-312)

Sudoku Hidato

Juego de logica creado por el Dr. Gyora Benedek, un matemático israelí. El objetivo de Hidato es rellenar el tablero con numeros consecutivos que se conectan horizontal, vertical o diagonalmente.

En cada juego de Hidato, los números mayor y menor están marcados en el tablero. Hay algunos números más en el tablero para ayudar a dirigir al jugador sobre cómo empezar a resolverlo

Panorama

Para facilitar el trabajo a las personas que utilicen nuestro proyecto, hemos dividido el mismo en una secuencia de menus interactivos los cuales permitiran efectuar todas las operaciones que se requieren en el documento de orientacion.

Hemos destinado ademas un archivo que contendra informacion de numerosos hidatos para efectuar operaciones de consulta, seleccion u eliminacion en dependencia de lo que el usuario decida. Lo hemos llamado “hidatos.txt”

Nuestro proyecto posee anexado un directorio denominado **clibrary** que contendra la definicion de una libreria que nos permite obtener un numero aleatorio. Hemos decidido incluir dicha libreria por problemas en la instalacion de paquetes.

Cada hidato esta formado por un tipo Matrix. Ademas posee la posicion en la esta ubicado el minimo y en la que esta ubicado el maximo. En conclusion:

type Matrix = [[String]]
data Position = None | Position Int Int
data Hidato = Empty | World Matrix Position Position

Para facilitar observar de manera clara el Hidato, definimos las casillas visitables como “xx”, las no visitables “--” y las ya visitadas con el numero correspondiente.

Caracteristicas de los menus

- *Menu Inicio*

Muestra un menu con las opciones de mostrar y generar hidatos

- *Menu Consulta*

Muestra todos los hidatos y opciones para seleccionarlos

- *Menu Seleccion*

Muestra el hidato seleccionado y opciones para resolverlo u eliminarlo

- *Menu Resultado*

Muestra el hidato seleccionado y su solucion

- *Menu Generacion*

Genera un nuevo hidato en base a algunos parametros, permite guardarlo

Detalles de implementacion

1. Tipos creados:

type Matrix = [[String]]
type Direction = (Int,Int)
data Position = None | Position Int Int
data Hidato = Empty | World Matrix Position Position

2. Metodos importantes

start_execution ->
Metodo que comienza la ejecucion. Muestra el menu de inicio

solve **->**
Se encarga de darle solucion a Hidatos que son pasados como parametros. En caso de que no tenga solucion devolvera un Hidato vacio. Para hallar la solucion tratamos a la matrix del hidato como un grafo donde cada casilla es un vertice y la adyacencia entre vertices es considerada arista. Realizamos un recorrido dfs desde la posicion inicial utilizando un array de direcciones, se comprueba en todo momento si la posicion es valida, si es invalida, se continua la exploracion sobre otra arista. Una vez se llega a la posicion final se devuelve el hidato resultante, en caso de que no sea alcanzada dicha posicion se devuelve un hidato vacio.

unique_solution **->**
Recibe como parametro un Hidato y verifica si este posee solucion unica. Para resolver el Hidato utilizamos un dfs, el dfs siempre obtiene la solucion mas a la izquierda, la forma de determinar si un hidato posee solucion unica es determinar la solucion mas a la derecha y si dichas soluciones son iguales entonces la solucion es unica, si son diferentes, entonces existe al menos dos soluciones para dicho hidato.

start_generation ->
Se encarga de generar hidatos de manera aleatoria. Recibe los siguientes parametros:

- rows -> Numero de filas
- cols -> Numero de columnas
- minValue -> Valor Minimo
- maxValue -> Valor Maximo
- cantDigitos -> Cantidad de digitos que contendra cada celda de la matrix del hidato.

Para generar el hidato primero se rellena la matrix con casillas desocupadas, luego se colocan todos los obstaculos, se intenta resolver el tablero resultante, si no tiene solucion se vuelve a ejecutar el metodo, si tiene solucion se verifica si es unica, en caso de serlo se ubican algunos valores en el tablero final que ayudaran al usuario a resolverlo y se devuelve; en caso de no ser unica se ubican tambien algunos valores y se vuelve a comprobar la unicidad y se repite este paso hasta que la cumpla, luego de esto se devuelve el hidato

Modo de uso

```
ghc Main.hs clibrary/lib.h -o Main
./Main
```