

Manual de Juego Sudoku

Manuel Suárez-Veronica Pozo-Alvaro Ortiz

1 de agosto de 2013

Índice general

1. Introducción	2
2. El Juego	3
3. Historia	5
4. Aplicación	6
5. Funcionalidad	10
6. Colaboraciones	14
7. Conclusiones	16

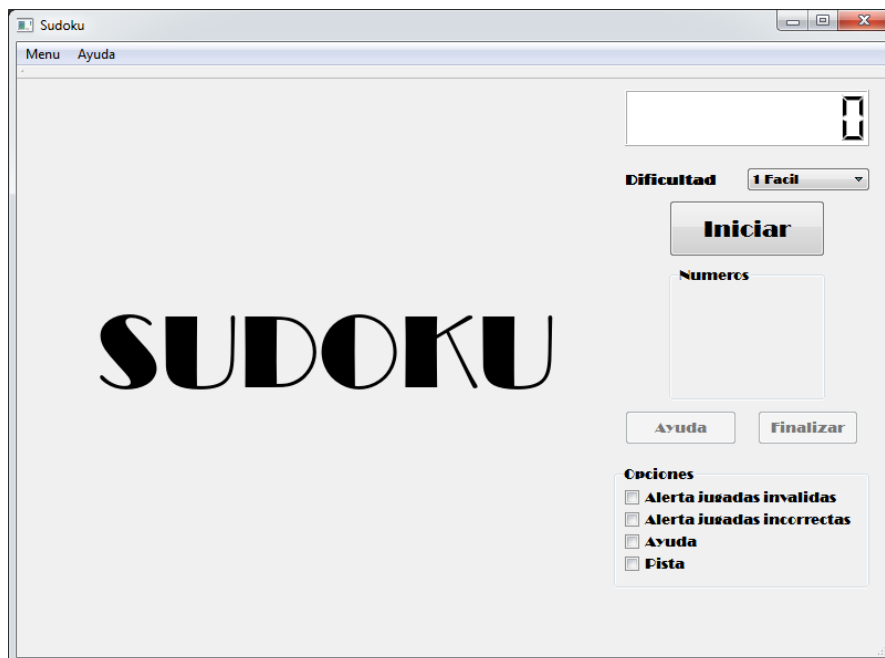
Capítulo 1

Introducción

El libro a continuación es creado para demostrar el proyecto realizado, Sudoku. En el cual se utilizara el lenguaje Python con una interfaz gráfica creada en Qt Creator, ademas utilizamos la libreria Pyside que nos sirve para trabajar Qt en Python.

Capítulo 2

El Juego



Sudoku es un pasatiempo que se publicó por primera vez a finales de la década de 1970 y se popularizó en Japón en 1986, dándose a en el ámbito internacional en 2005 cuando numerosos periódicos empezaron a publicarlo en su sección de pasatiempos. 1 El objetivo del sudoku es rellenar una cuadrícula de 9 x 9 celdas dividida en subcuadrículas de 3 x 3 (también llamadas cajas o regiones) con las cifras del 1 al 9 partiendo de algunos números ya dispuestos en algunas de las celdas. Aunque se podrían usar colores, letras, figuras, se conviene en usar números para mayor claridad, lo que importa, es que sean nueve elementos diferenciados, que no se deben repetir en una misma fila, columna o subcuadrícula. Un sudoku está bien planteado si la solución es única. La solución

de un sudoku siempre es un cuadrado latino, aunque el recíproco en general no es cierto ya que el sudoku establece la restricción añadida de que no se puede repetir un mismo número en una región.

Capítulo 3

Historia

El Sudoku puede haberse ideado a partir de los trabajos del matemático suizo Leonhard Euler (1707-1783), quién utilizó los cuadrados latinos¹ para realizar cálculos probabilísticos, aunque su aparición más obvia se dio en el año 1979, cuando el norteamericano Howard Garns publica en Nueva York, a través de la empresa Dell Magazines, un juego muy similar conocido bajo el nombre de “Number Place”. En el año 1984 una editorial japonesa lo exportó hacia ese país, divulgándolo en el periódico “Monthly Nikolist” bajo el nombre de “Los números deben estar solos” (cuyo apócope en japonés es SuDoku). La editorial introdujo con el correr de los años algunas innovaciones que hicieron que el juego ganara gran popularidad entre los lectores. Sus inicios reales como Sudoku en Occidente se remontan a los años 2004/2005, gracias a su publicación en los diarios “The Times” y “The Daily Mail”. Desde aquel entonces, el juego ha alcanzado una gran popularidad en el mundo moderno: en 2005 se publicó el primer libro acerca de Sudokus en el mundo (“Los mejores Sudokus”, con 200 tableros agrupados en 4 niveles de dificultad) y se lo incluyó entre los 9 problemas del ICPC (sigla de International Collegiate Programming Contest).

Capítulo 4

Aplicación



Nuestra aplicación tiene una ventana principal donde podemos apreciar un menu bar con dos opciones:

- Menu Nos da las opciones de cargar y guardar partida, además de salir del programa.
- Ayuda Nos da las opciones de Ayuda para el programa y un acerca de.

Tenemos en el panel izquierdo:

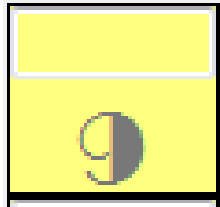


- LCD Widget LCD para mostrar el tiempo.
- ComboBox Widget para elegir el nivel de dificultad.
- Boton Boton para generar y mostrar el tablero de Sudoku.
- Matriz 3x3 de Botones Botones para poder llenar el tablero
- Boton Boton de ayuda y Finalizar juego. (Ayuda se activa con los checkbox de abajo)
- CheckBox Conjunto de Checkbox para activar las alertas de jugadas inválidas, incorrectas, Ayuda, Pista.

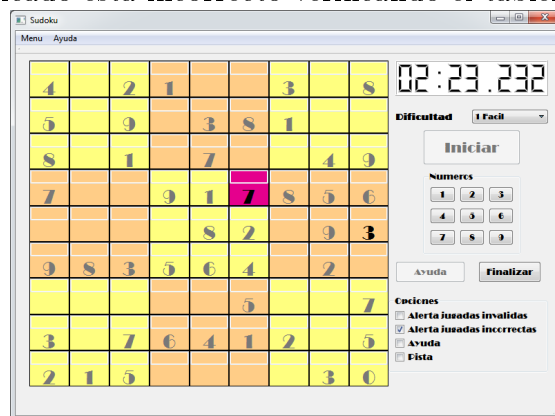
Las matrices 3x3 que componen el tablero 9x9 se aprecian así:

9	5	
	4	1
	3	8

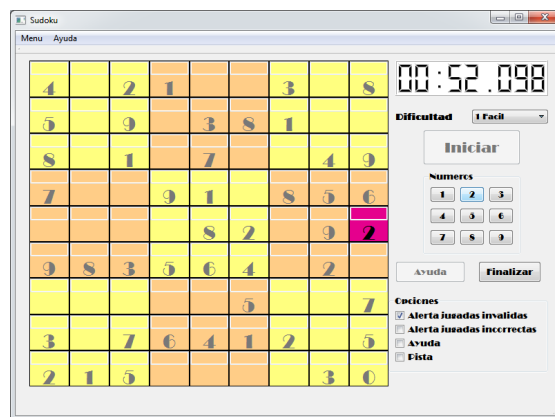
La casilla hereda de QFrame con un QLineEdit y un Boton, y con el Qframe se puso fondo.



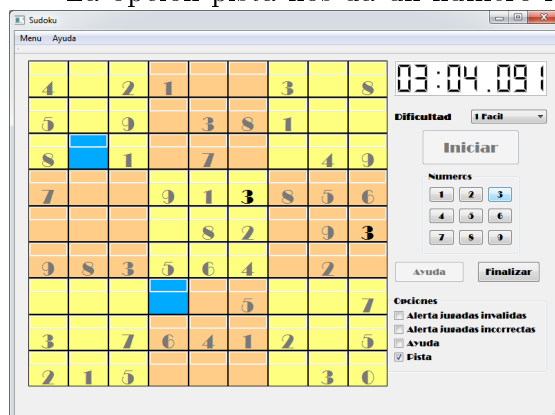
Una jugada Invalida y una jugada incorrecta se muestra asi: La opción de Jugada Incorrecta nos muestra con un color diferente si las jugadas son incorrectas o que el valor ubicado esta incorrecto verificando el tablero generado.



La opción de Jugada Incorrecta nos muestra con un color diferente si las jugadas son invalidas o que el valor ubicado ya esta en esa fila,columna o cuadrícula.



La opción pista nos da un numero random correcto, y se muestra así:



La opción ayuda nos activa el boton .Ayuda.^{el} cual por cada click nos ubica un valor correcto en una posicion random.

Capítulo 5

Funcionalidad

Generacion de tableros:

- Algoritmo implementado.
 - Se genera un tablero con numeros random hasta que este sea resoluble.
 - Se verifica si es resoluble.
 - Se trata de resolver por arreglos de posibilidades de numeros en casillas.
 - Si no se puede resolver así, se utiliza la técnica del Backtracking.

```
bool Generador::BackTrackSolucion(int startx, int starty){  
  
    bool solved = esCompleto();  
    bool foundzero = false;  
    int x = startx, y;  
  
    for (y = starty; !solved && !foundzero && y < 9; ++y) {  
        for (x = startx; !foundzero && x < 9; ++x){  
            foundzero = (tablero[(x*9)+y] == 0);  
            startx = 0;  
        }  
        x--;  
        y--;  
  
        if (!solved && foundzero) {  
            for (int e = 1; e < 10 && !solved; ++e){  
                if (MovimientoValido(x, y, e)) {  
                    Set(x, y, e);  
                    if (BackTrackSolucion( x , y+1))  
                        solved = true;  
                    else  
                        tablero[(x*9)+ y] = 0;  
                }  
            }  
        }  
    }  
  
    return solved;  
}
```

- Utilizando Backtracking se busca solucionar cambiando los numeros de-
tras del vacio, y utilizando recursion de la misma.

```

void Generador::generarSolucionRandom(){
    srand((unsigned int) time(NULL));
    int x, y, num_de_Random,v;

    num_de_Random = 25;
    do {
        for(int i=0;i<81;i++) tablero[i]=0;
        for (int i = 0; i < num_de_Random; ++i) {
            x = rand() % 9;
            y = rand() % 9;
            v = rand() % 9 + 1;
            if (MovimientoValido(x, y, v)){
                Set(x, y, v);
            }
            else{
                i--;
            }
            x=y=v=0;
        }
    } while (!resolverTablero());
}

```

■ Dificultad.

- Se procede a quitar numeros del arreglo, y comprobar si se puede resolver y tienen única solución.
- Se agregan en un pila los numeros y posiciones que fueron quitadas y se podia obtener única solución.
- Segun el numero de dificultad se obtienen posiciones random repetidamente tantas casillas hay, y se ve las veces que puede estar ese numero.

```

bool stillFoundOne = true;
pila= QStack<Posicion>();
while (Insertar > 0 && stillFoundOne) {
    stillFoundOne = false;

    int yoffset = rand() % 9;
    int xoffset = rand() % 9;

    for (int y = 0; y < 9 && Insertar > 0 && !lista_op.empty(); ++y)
        for (int x = 0; x < 9 && Insertar > 0 && !lista_op.empty(); ++x) {
            int realx = (x + xoffset) % 9; // The real x and y positions
            int realy = (y + yoffset) % 9;
            Posicion pe((realx*9)+realy, tablero[(realx*9)+realy]);
            pila.push(pe);
            tablero[(realx*9)+realy]=0; // Remove the element from the board and try to solve it again

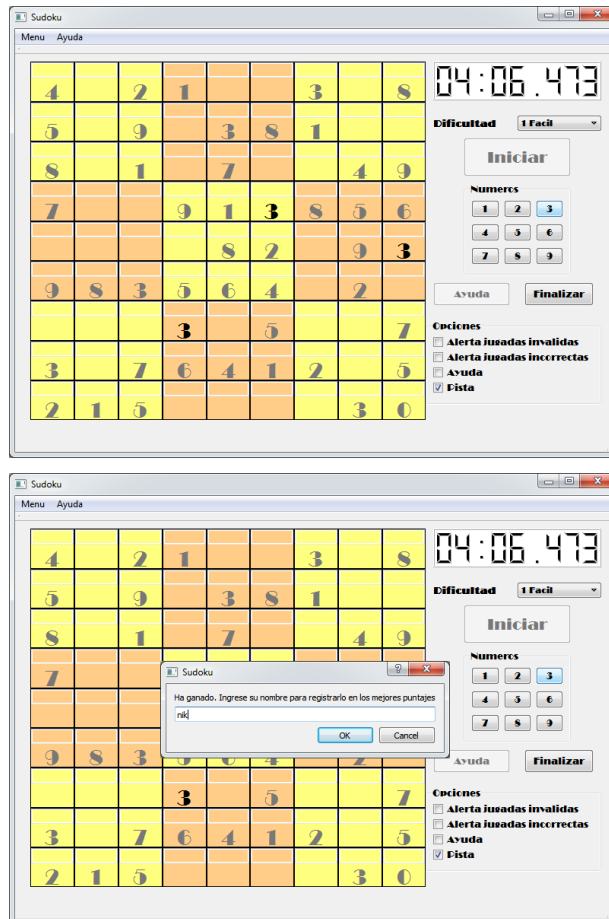
            bool foundOne = false;
            for (QList<Posicion>::iterator it = lista_op.begin(); it != lista_op.end() && !foundOne;) {
                tablero[it->GetPos()]=it->GetValor();
                if (VerificarResolucion(true)) { // Board is solvable by ScanSolve()
                    Posicion addpos(it->GetPos(), it->GetValor());
                    pila.push(addpos);

                    it = lista_op.erase(it); // Remove from the undoList and add to the pila

                    --Insertar;
                    foundOne = true;
                }
            }
        }
}

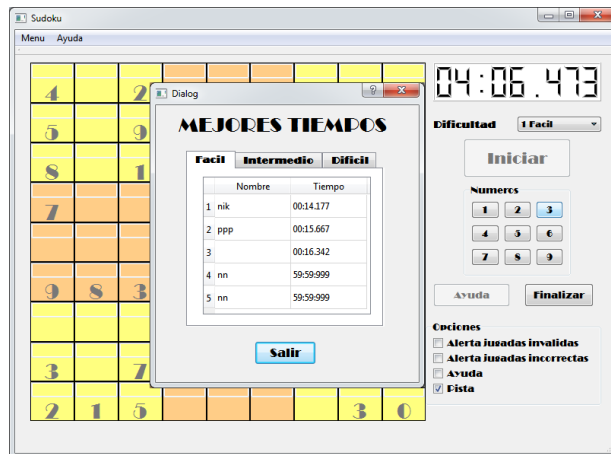
```

- Guardar y cargar tableros cifrados .
- Jugar y finalizar juego (2).



Ademas el juego finaliza una vez que se llene todo.

- Puntaje (3).



Capítulo 6

Colaboraciones

- Commits por integrante.

- Veronica Pozo: 12
- Manuel Suárez: 8
- Alvaro Ortiz: 2

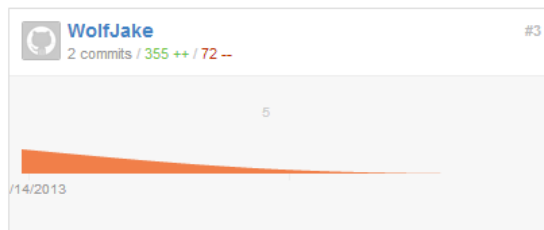
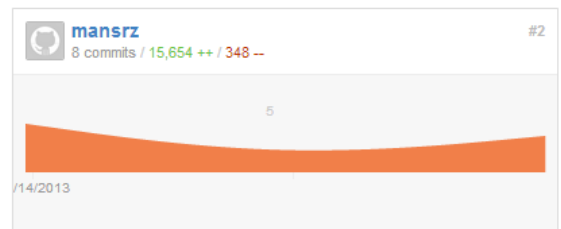
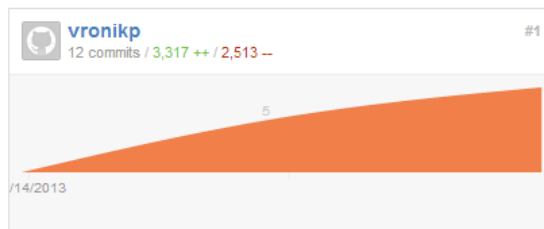
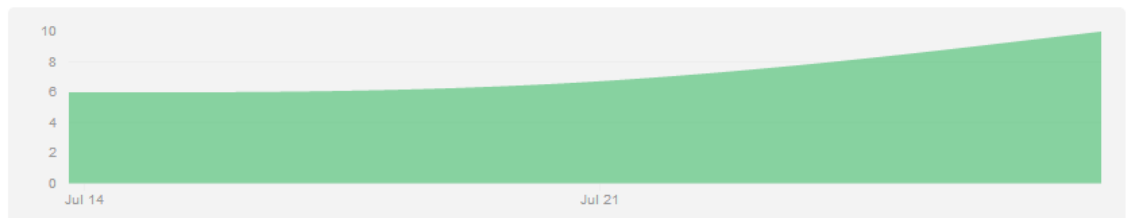


- ¿En qué fechas hay picos?

- 30 de Julio
- 26 de Julio
- 10 de Julio

July 13th 2013 - July 27th 2013
Commits to master, excluding merge commits

Contribution Type: Commits



Capítulo 7

Conclusiones

El proyecto se realizo con exito logrando implementar la mayoria de caracteristicas en nuestra aplicacion que se pedian. Se conocio y practico el lenguaje pyhton con la libreria Qt. Realizamos un trabajo grupal utilizando la herramienta de versionamiento "Git", siendo esta fundamental en el proceso de creacion del proyecto.