**PROGRAM DESIGN 2**

Name : 皮宜恩

Student ID : F74027036

Department : CSIE 106 甲

Assignment : Project 1

Environment : Ubuntu 14.02

Execution :

make

./Sudoku

**DESCRIPTION**

This Program is completely written in c++ language. It’s a classic reproduction of Sudoku game. To respect the requirement of this project we have to main program, one that you can execute to give question that’s mean to generate a random solvable Sudoku board and the other one is to solve any Sudoku board if it’s solvable.

**PREAPARATION**

To be able to write this program, I first play a lot of online Sudoku game to be more familiar with the game, then I read a lot of article on Wikipedia about different algorithm that someone can use to create a randomly Sudoku board. After, some research I was easily be able to write a c++ to read and solve any solvable Sudoku board but the most difficult task for me, was to randomly create my own Sudoku board. Well, unfortunately the method that I used for this requirement might not be the most robust one, but at least I my program can randomly create a board even it’s take quite long time.

**DESIGN**

I create some private method like; bool AbsentSurLigne, bool AbsentSurColonne and bool AbsentSurBlock. Those Methods move on each row/column/block of the board and return FALSE if the find the value (1-9), otherwise they return TRUE. Now, to solve the quiz I create a Method calls EstValide (int grille [][], int position) , as you can see in the prototype, this method receive a board and solve it. And still, this method return a Boolean to constantly verify if there’s no blockage. However, to make this Method totally functional we constantly actualize the board during the recursive call by initialize the case to 0 if there’s any blockage by using the BACKTRACKING algorithm.

About the give question part, well my method is not a really accurate one, but what I do is to create a solvable Sudoku board and randomly remove some case and print out the new incomplete board.

**MY PERSONAL POINT OF VIEW**

I think my algorithm is quite good to quickly solve any Sudoku board, still I was thinking about using link list to optimize the result and verify if there’s only one solution. To generate a random board, I think my code is really not robust, it can be so much better. So, I am looking forward to have free time to work on that project and maybe improve it somehow that it can be play by computer or by player.

|  |
| --- |
| **SUDOKU** |
| - no\_number : integer  - dont\_care : integer  - tmp : integer  - check : Boolean  - grille[sudoku\_size ][sudoku\_size ] : Integer |
| <<constructor >> +Sudoku ()  <<constructor>> + Sudoku ( init\_grille [sudoku\_size][sudoku\_size] : Integer)  + ReadIn() : void  + GiveQuestion() : void  + Solve() : void  + AddExtra() : void  + PrintBoard ( grille [sudoku\_size][sudoku\_size] : integer) : void  - AbsentSurLigne ( s : integer , o: integer , n : integer) : Boolean  - AbsentSurColonne ( s : integer , o: integer , n : integer) : Boolean  - AbsentSurBlock ( s : integer , o: integer, n : integer) : Boolean |

**UML DIAGRAM**