SOFTWARE REPORTING SPECIFICATION
CS 101 Projects 2014

Sudoku

Group CUSE

Members-

Rajkumar Ramavath	– 140110088
Om prakash	- 140110083
Akhil Sai	- 14D170028
Kartik Sahu	- 14D110018

Index

Reference

1.lecture slides

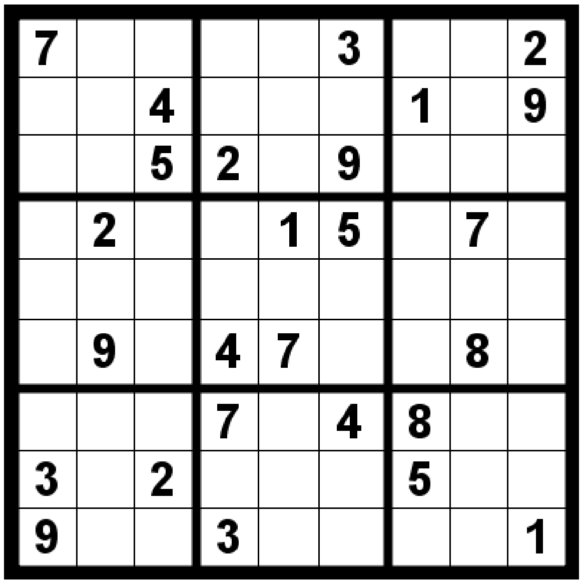
2. Similar projects of previous semester

3.Internet

4.An Introduction to Programming through C++ by Abhiram Ranade

Overall Description

Sudoku is a popular numerical puzzle solved and enjoyed all over the world. This game has a 9x9 grid filled with a few numbers which is given to the user. The grid is then divided into 9,3x3 boxes as shown in the figure below. The aim of the game is to fill the blank spaces with numbers from 1 to 9 in such a way that all the rows, columns and boxes have all the numbers from 1 to 9. Thus because of the dimensions of the grid, we see that no number should be repeated along a row, column or in the block.



7					3			2
		4				1		9
		5	2		9			
	2			1	5		7	
	9		4	7			8	
			7		4	8		
3		2				5		
9			3					1

The user is given a Sudoku puzzle and the program can also auto solves the Sudoku given by the user.

Functionality

a) Sudoku Generator

Main aim of this part is to check the Sudoku solving skills and capability of the user to solve Sudoku .

The method used to generate sudoku is quite obvious . The unsolved Sudoku is saved in particular directories or files and named on the basis of difficulties i.e. the user is asked the level of toughness he wishes in Sudoku(namely easy, medium and hard) .Based on the users choice of difficulty level the unsolved Sudoku is generated and is solved internally by the program using the “sudokusolver()” function defined in the program and the users input after completing the sudoku is matched with the solved sudoku to validate him whether he has correctly solved sudoku or not.

a) Sudoku Autosolver

Main aim of the program is to generate all possible configurations of numbers from 1 to 9 to fill the empty cells one by one until the correct configurations is found and finally displaying it.

The method used for solving Sudoku is BACKTRACKING. In backtracking, Sudoku is solved one by one assigning numbers to empty cells .Before assigning a number, the number is checked whether the number is possible at that cell or not , according to the rules of Sudoku i.e. We basically check that the same number is not present in current row, current column and current 3X3 subgrid. After checking for validity of a number, we assign the number, and recursively check whether this assignment leads to a solution or not. If the assignment doesn’t lead to a solution, then we try next number for current empty cell. And if none of number (1 to 9) lead to solution, we return false.

Algorithm for Sudoku AutoSolver

```
Find row, col of an unassigned cell , If there is none, return true
  For digits from 1 to 9
    a) If there is no conflict for digit at row, col assign digit to row,
       col and recursively try fill in rest of grid.
    b) If recursion successful i.e. Sudoku is solvable, return true
    c) Else, remove digit and try another.
  If all digits have been tried and nothing worked, return false.
```

Function Prototypes

1.bool FindEmptyLocation(int grid[9][9], int &row, int &col);

This function basically finds the unassigned cells or empty cells in the Sudoku which is input by the user i.e. this function accepts the Sudoku 2-D array and just changes the values of rows and columns passed to it by reference to the value of unassigned cell or empty cell.

If empty location found, it changes the values of row and col to the coordinates of that location and returns true else if not found it returns false i.e. the Sudoku is solved.

2. bool number validity(int grid[9][9], int row, int col, int num);

This function basically checks whether the particular number num accepted as parameter is valid in the row, column which it is accepting as a parameter by a series of functions.

2.1) bool PresentInRow(int grid[N][N], int row, int num)

Returns a boolean which indicates whether any assigned entry in the specified row matches the given number i.e. whether the number is possible in a row or not.

2.2) bool PresentInCol(int grid[N][N], int col, int num)

Returns a boolean which indicates whether any assigned entry in the specified column matches the given number i.e. whether the number is possible in column or not.

2.3)bool UsedInBox(int grid[N][N], int boxStartRow, int boxStartCol, int num)

Returns a boolean which indicates whether any assigned entry within the specified 3x3 box matches the given number i.e. whether the number is possible in particular box or not.

3. bool SolveSudoku(int grid[9][9])

Takes a partially filled-in grid and attempts to assign values to all unassigned locations in such a way to meet the requirements for Sudoku solution (non-duplication across rows, columns, and boxes) i.e . this particular function is practically contains the code for solving the Sudoku recursively using backtracking method.

4.int SudokuGenerator(int grid[9][9],int temp)

This function basically reads the unsolved Sudoku already present in the file based on the level of difficulty passed as temp and the Sudoku generated is randomly generated using random function.

5.)printGrid(grid[9][9])

This function basically prints the solved grid and generates the grid using the basic functions of simple cpp to create lines, rectangle and coloring it using set color function defined in header file #include<simplecpp> and even

6.)other functions used to implement graphics as well

Supportability

Windows Operating System

Graphics

1.)Used line and Rectangle function defined in the header file #incluyde<simplecpp> to generate lines, rectangles and using these we generated a grid.
2.)Took input in form of mouseclicks and retrieve the position of mouse click using getClick() function defined in simplecpp and on the basis of the position of the mouse click in a particular range displayed the output grid and other buttons which have their own functionality.
3.) using mouse click to play the game and making the game more user friendly by making the program more interactive and making the program more attractive using simple cpp