

Sudoku

CS 101 Project

Autumn, 2014

Pranav Damani, Yash Dhoble, Yash Bhagat, Vikas Sutrar

Introduction

This project aims to create a Sudoku game and auto-solver. Sudoku is a logic based, combinatorial, number-placement puzzle. Completed puzzles are always a type of Latin square with an additional constraint on the contents of individual regions. For example, the same single integer may not appear twice in the same row, column or in any of the nine 3×3 sub-regions of the 9×9 playing board.

Purpose and Scope

The purpose of this project is to create a Sudoku program and auto-solver. This enables the user to play the game as well as the program has the ability to solve the grid as input by the user. Also the program has the ability to generate Sudoku grids of varying levels of toughness.

It has a scope of verifying the given set of entries by the user and if required, solving the generated Sudoku uniquely.

Salient Features

- Auto-solver : The program finds the unique solution of the grid as input by the user.
- Difficulty Levels : The user has the opportunity to select the level of the grid. There are three levels : Easy, Moderate and Hard
- Timer : The program keeps note of the time taken by the user to solve the puzzle
- Scores : Depending upon the level and time taken the program assigns score to the user.

Methodology

We will use backtracking algorithm to code this program. The algorithm is stated below :

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, return true

c) Else, remove digit and try another If all digits have been tried and nothing worked, return false

Graphics

- * The graphics in this project have been added by using simplecpp libraries. The working can be done by using either turtle or coordinate system. We have used coordinate system to produce the grids and numbers. The numbers in the sudoku grid are input through mouse pointer and we also had to validate the mouse click.

Sample

The following pages give a basic idea of how the program will look while it is executed in autosolver or grid generator mode.

Autosolver Mode

The screenshot displays the Code::Blocks IDE interface with the 'Autosolver Mode' for a Sudoku puzzle. The main window shows the C++ source code for 'Untitled1.cpp', which includes functions for initializing the canvas, selecting the Sudoku grid, and solving it. The code uses a 9x9 grid and a 3x3 sub-grid to define the puzzle structure.

```

}
}
else if(num==2)
{
    initCanvas();
    selectSudoku(t0, matrix, grid, str);
    cout<<str;
    //generate(t0, str, matrix, grid);
    SolveSudoku(grid);

    //9*9 sized rectangle
    Rectangle r(220 , 220, 360, 360); r.setColor(GREEN);

    int i,j;

    //creating the lines of the sudoku 9*9 box
    //used coloured lines to define the 3*3 boxes
    Line lv1(80,40 , 80,400);

```

The 'Simplecpp Canvas' window shows a 9x9 grid representing the Sudoku puzzle. The grid is divided into 3x3 sub-grids. The numbers in the grid are as follows:

9		5			6	7	3	
4				5	2	1		8
				7			2	
					1	3	7	
2	6	1				8	4	9
	3	9	6					
	5			4				
1		8	7	9				3
	9	7	2			4		1

Buttons for 'RESET', 'SUBMIT', and 'DELETE' are visible next to the grid. Below the grid is a row of buttons labeled 1 through 9.

The 'Logs & others' window shows the execution output, indicating that the program is running successfully. The console window shows the prompt 'Enter 1 or 2:' and the options '(1)Autosolver' and '(2)Generator'.

Untitled1.cpp [sudoku1] - Code::Blocks svn bu

File Edit View Search Project Build Debug Tools Plugins DoxyBlocks Settings Help

Build target: Debug

Management

Projects Symbols Files

F:\

Mask:

F:\

C++

Codeblocks-Simplecpp

DC++

IIT

movies

CB-Simplecpp-setup.exe

codeblocks_13.12-1.tar.gz

gB29x5yQ4k.docx

```

    }
    }
    else if(num==2)
    {

        initCanvas();
        selectSudoku(t0, matrix,grid, str);
        cout<<str;
        //generate(t0,str,matrix,grid);
        SolveSudoku(grid);

        //9*9 sized rectangle
        Rectangle r(220 , 220, 360, 360); r.setColor(GREEN);

        int i,j;

        //creating the lines of the sudoku 9*9 box
        //used coloured lines to define the 3*3 boxes
        Line lv1(80,40 , 80,400);

```

Logs & others

Code::Blocks Search results Build log Build messages

Checking for existence: F:\IIT\CS101\sudoku1\bin\Debug\sudoku1.exe

Executing: "F:\Codeblocks-Simplecpp\cb_console_runner.exe" "F:\

Simplecpp Canvas

9	2	5	8	1	6	7	3	4
4	7	6	3	5	2	1	9	8
8	1	3	4	7	9	6	2	5
5	8	4	9	2	1	3	7	6
2	6	1	5	3	7	8	4	9
7	3	9	6	8	4	5	1	2
6	5	2	1	4	3	9	8	7
1	4	8	7	9	5	2	6	3
3	9	7	2	6	8	4	5	1

RESET

SUBMIT

1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---

DELETE

Enter 1 or 2:
(1)Autosolver
(2)Geneerator

Grid Generator Mode

Simplecpp Canvas

Untitled1.cpp [sudoku1] - Code::Blocks svn build

6		4		7		8		5
	8						7	
	2	3	5	8				
5		8		2		6	4	1
9						3	2	
4	6			3	7		5	
8								6
	5				2			4

RESET

SOLUTION

SUBMIT

DELETE

cout<<"Bad input!"<<endl;

reloaded sudoku on the

pen("1.txt", "r"); fread

pen("2.txt", "r"); fread

pen("3.txt", "r"); fread(str, 1, 81, fp); if(fp==NULL){cout<<"Cannot open file"; return

pen("4.txt", "r"); fread(str, 1, 81, fp); if(fp==NULL){cout<<"Cannot open file"; return

pen("5.txt", "r"); fread(str, 1, 81, fp); if(fp==NULL){cout<<"Cannot open file"; return

pen("6.txt", "r"); fread(str, 1, 81, fp); if(fp==NULL){cout<<"Cannot open file"; return

pen("7.txt", "r"); fread(str, 1, 81, fp); if(fp==NULL){cout<<"Cannot open file"; return

F:\IIT\CS101\sudoku1\bin\Debug\sudoku1.exe

Enter 1 or 2:

(1)Autosolver

(2)Geneerator

Enter a level : 'E' for easy and 'H' for hard : E

6040708050800000700235800005080206410000000009000003204600370508000000060500020

4_

Logs & others

Code::Blocks

Search results

Build log

Build messages

CppCheck

CppCheck messages

Debugger

DoxyBlocks

Closed files list

Checking for existence: F:\IIT\CS101\sudoku1\bin\Debug\sudoku1.exe

Executing: "F:\Codeblocks-Simplecpp\cb_console_runner.exe" "F:\IIT\CS101\sudoku1\bin\Debug\sudoku1.exe" (in F:\IIT\CS101\sudoku1\.)

WINDOWS-1252

Line 484, Column 21

Insert

Read/Write

default

03:07 PM
24-Nov-2014

Variants

- * The program can further be modified to create different variations of the classical Sudoku puzzle. Some of the popular variations include :
 - Variation in grid sizes
 - Imposing Additional Constraints
 - Mini Sudoku
 - Cross Sums Sudoku
 - Killer Sudoku

References

Books :

- Cohoon, James P. and Davidson, Jack W., An Introduction to Programming and Object- Oriented Designing
- Arora, Sumita, Computer Science with C++, Dhanpat Rai Co.

Websites :

- www.en.wikipedia.org
- <http://www.youtube.com/watch?v=p-gpaIGRCQI>
- www.stackoverflow.com
- www.sanfoundry.com
- www.quora.com
- www.cplusplus.com