Sudoku *Mania*

Contents

* Introduction
* Project Basics
* Requirements
* Functions used and Working
* Batch Details

Introduction

Sudoku puzzles are popular all over the world so much that one can find a Sudoku puzzle in half of the newspapers published .This program is designed both to create a Sudoku puzzle and to solve it. Sudoku is basically a puzzle game which tests the logical capabilities of the solver. Sudoku , today, has developed to much complicated levels like the diabolical Sudoku which can have more than one possible solutions. Making a Sudoku solver or a generator requires a good understanding of the basic logic involved in deciphering the puzzle and also a good understanding of the programming language involved.

Project Basics

The project is required to provide a two way interface to the concerned user. The program will work both as a simple Sudoku solver as well as a Sudoku puzzle game which will generate a puzzle and will then allow the user to solve it and will later on check the user’s solution.

Rules of the Game:

A Sudoku is a 9\*9 matrix and contains digits from 1 to 9,thus contains 81 elements in all. Out of these elements some are missing from the matrix(depending upon the difficulty level) and the user is required to fill them back in such a way that every row , column or 3\*3 sub-matrix has all the digits from 0-9 filled exactly once without any repetition.

Generating a puzzle:

To generate the Sudoku puzzle we used files. We stored a certain number of solved sudokus in different files. We wrote a function which would use a random Sudoku from these files, remove a certain number of elements randomly and then will return the puzzle so generated.

Solving the Sudoku:

For solving the Sudoku we used an algorithm which will check whether any number has any unique position in the entire Sudoku, if it does then that number is assigned to that position or else it finds the total number of places where a given number can exist and then assigns various numbers to various positions if there is only one possible solution.

Requirements

Ubuntu(>=10.04)

Decent Processor

Ezwindows

Functions used and working

For solving algorithm:

* Recursive functions
* Display()-It is a nice board to display Sudoku elements.
* CheckRow()-checks if an element is present in a row or not.
* CheckColumn()-checks if an element is present in a column or not.
* CheckBox()-checks if an element is present in a 3x3 matrix or not.
* CanBox()-checks the probability of a number in a 3x3 box.
* sudokuSolver()-Solves the Sudoku using the above functions.
* sudokuSolved()-checks if a Sudoku is completely solved or not.
* Main()- provides the user with a choice for Sudoku solver or Sudoku game. If a user chooses for Sudoku solver it solves a random unsolved Sudoku file and displays it in the terminal.
* For and while and do-while loops
* If else ladders

For generating algorithms:

* We used a unique strategy ..
* Used files to store solved sudokus and using a function written by us the program removes certain number of elements randomly and returns the generated puzzle.
* Used ‘rand’ and ‘srand’ from the C++ library.
* If the user choses the Sudoku game the main function generates a unsolved Sudoku from a random solved Sudoku file.

Ez Windows:

* Used function to generate lines,rectangles and using these we generated a grid.
* Took input in form of files and displayed the output on EzWindows using our program.
* Mouseclick1()- Allocates co-ordinates of the region of mouse click. In EzWindow starts game.
* sudoku1()-Allocates position to the user interface command.
* Write()-writes values of question Sudoku on EzWindow screen.
* Sudoku()-EzWindow for Sudoku grid which gives question Sudoku to user and enables him to solve it.
* Sudoku2()-EzWindow for user to provide an unsolved Sudoku which our program is supposed to solve.
* Sudoku3()-EzWindow for showing message.
* Mouseclick()-Function for EzWindow Sudoku.
* Mouseclick2()-Allocates co-ordinates of the region of mouse click in EzWindow Start game and switches to EzWindow solver.
* *However, the program worked well, but we faced some problems while integrating the graphic interface with the solver and generator and therefore, had to drop our work on graphics.(Code is included with other programs.)*

Lab Batch Details

Project Mentor

Shaikh Samiulla

Team Members

Ankit Agrawal

Anand Agarwalla

Anand Soni

Anurag Srivastava

Anushrut Sharma

Ankit Kumar Singh