

Project report

Introduction:

Sudoku has its deep roots in ancient number puzzles. For many centuries people have been interested in creating and solving puzzles. The puzzle was popularized in 1986 by the Japanese puzzle company Nikoli, under the name Sudoku, meaning single number. It became an international hit in 2005.

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

The Basic Rules of Sudoku:

There is only one valid solution to each Sudoku puzzle. The only way the puzzle can be considered solved correctly is when all 81 boxes contain numbers and the other Sudoku rules have been followed.

When you start a game of Sudoku, some blocks will be pre-filled for you. You cannot change these numbers in the course of the game.

Each column must contain all of the numbers 1 through 9 and no two numbers in the same column of a Sudoku puzzle can be the same.

Each row must contain all of the numbers 1 through 9 and no two numbers in the same row of a Sudoku puzzle can be the same

Each block must contain all of the numbers 1 through 9 and no two numbers in the same block of a Sudoku puzzle can be the same.

Why Sudoku is a fun game?

Sudoku is a “Brain Game”

Just as people have to exercise our bodies in order to stay physically fit, we also need to “exercise” our brains. Many seniors are interested in keeping their minds sharp and in continuing to strengthen their cognitive abilities as they get older.

THE PROJECT AT A GLANCE

This project aims at the generation and solving of Sudoku puzzles using programs coded using C++ as the programming language. The project has been designed as a two way interface for the user working on the program. On the one hand the program accepts a Sudoku puzzle from the user and generates a solution to it.

Program module

Vaildation of the Sudoku:

Bool ValidateinputBX () -

Checks if an element is repeated in a particular 3x3 grid.

Bool ValidateinputHV () -

Checks if an element is repeated in the same horizontal row or the same vertical column.

For solving algorithm:

UsedinRow () -

Checks if an element is already present in the Row or not.

UsedinCol () -

Checks if an element is already present in the Column or not.

UsedinBox () -

Checks if an element is already present in the Box or not.

FindUnassignedLocation () -

This functions tells the Co-ordinates of Empty Grids.

NoConflict () -

This function tells if the Element in consideration violates the rules of the Sudoku or not.

SolveSudoku () -

Solves the Sudoku recursively using the above functions by employing the method of Backtracking.

Main () –

Provides the User with the option of Sudoku Validator or Sudoku Autosolver.

Features Of The Programme:

The user will see a 9x9 Sudoku grid. And then he/she will partially fill it.

User may click the [check button](#) to check if his/her inputs are in accordance with the rules of the game.

User may click the [solve button](#) to get a solution to his/her Question Sudoku.

NOTE - The Sudoku will be **uniquely** solved only when the inputs are more than 17 and follow the rules of the game.

The user will notice all the Error Statements in a [message box](#) on the Right of the screen.

In order to quit the User can simply click the [Quit Button](#) on the lower right of the screen.

Team members:-

Prashant kumar varun

Priyash singh

Tanmay bhoite

Manish Godara