------------------------------------------------------------------------------------------------------------

***Declaration of understanding***

*I understand that plagiarised answers will receive a grade of zero and an academic misconduct.*

*I understand that an academic misconduct will result in an automatic zero in the assignment in the event that this is my first offense and an automatic grade of zero in the course if this is my second offense.*

*I understand that all answer material, must be original material.*

*I understand that plagiarism means to present someone else’s material as my own.*

*I understand that to copy material from the internet, text books, course slides, or any other source, is a form of plagiarism.*

*I understand that to copy material from the internet, text books, course slides or any other source, and then to change a few variable names etc. in that material is a form of plagiarism.*

*I understand that to copy material from the internet, text books, course slides or any other source, and then to slightly modify that material is a form of plagiarism.*

*I understand that the appropriate way to complete this assignment is to write and test all of the code myself while referencing other sources for general guidance.*

*I have read and clearly understand each one of these statements, and accept the responsibility and penalty for any actions that I take which may contravene any one of these statements.*

*Name Priyansh Thakar*

*Student Number: 200472583*

*Signature: *

------------------------------------------------------------------------------------------------------------

**COMP 1030**

**Final Project - Sudoku**

**For Sudoku Class:**

Input:

/\*\* Application Purpose: To create a class called Sudoku which provides an entry point for

compiler and allows the user to select option as per his/her need.

Author: Priyansh Thakar

Date: 03 April 2021

Time: 02:32 PM

\*/

// Importing to use Scanner for getting user data

import java.util.Scanner;

public class Sudoku

{

public static void main(String []args)

{

// Declaring and Initializing flag variable to keep track of loop

int flag = 1;

Scanner in = new Scanner(System.in);

// Looping until flag value is not 0

while (flag == 1)

{

// Printing introductory statements

System.out.println("---------------------------------------------------");

System.out.println("Please provide your choice: ");

System.out.println("1. Solve Pre-defined Sudoku puzzle?");

System.out.println("2. Get a new question puzzle?");

System.out.println("3. Solve your Sudoku puzzle?");

System.out.println("4. Check your Sudoku puzzle?");

System.out.println("5. Exit");

// Storing user choice in variable

int ch = in.nextInt();

// For readability

System.out.println("---------------------------------------------------");

switch (ch)

{

// For solving pre-defined puzzle

case 1:

// Declaring and Initializing a 2d array of matrix

int[][] matrix1 = {{5, 0, 8, 0, 1, 0, 4, 0, 7},

{3, 4, 0, 0, 0, 0, 9, 0, 0},

{1, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 7, 3, 0, 9, 1, 2, 5, 0},

{8, 0, 1, 7, 0, 6, 0, 4, 0},

{9, 2, 0, 0, 0, 5, 0, 7, 1},

{0, 8, 5, 4, 0, 0, 0, 0, 0},

{0, 1, 9, 5, 0, 0, 7, 0, 0},

{7, 3, 4, 0, 0, 0, 5, 9, 8}};

// Creating object of SudokuSolver

SudokuSolver ss1 = new SudokuSolver(matrix1);

// Solving Sudoku

if (ss1.attemptValues(0, 0) == true)

{

// Printing Sudoku matrix with static method

SudokuPrinter.printMatrix(matrix1);

}

else

{

System.out.println("No Solution exists");

}

break;

// For providing the user with new question

case 2:

// Declaring a new Sudoku matrix

int[][] matrix2 = new int[9][9];

System.out.println("Solve the following question:");

// Creating object of SudokuQuestion

SudokuQuestion sq1 = new SudokuQuestion();

// Storing the puzzle question in the matrix

matrix2 = sq1.generatePuzzle();

// Printing Sudoku matrix with static method

SudokuPrinter.printMatrix(matrix2);

break;

// For providing the solution of user's question

case 3:

// Declaring a new Sudoku matrix

int[][] matrix3 = new int[9][9];

System.out.println("Please provide your 9 x 9 matrix.");

System.out.println("Give out space between each column element and enter after

each row.");

System.out.println("You can enter now:");

// Storing each user integer into our matrix array

for (int row = 0; row < 9; row++)

{

for (int column = 0; column < 9; column++)

{

matrix3[row][column] = in.nextInt();

}

}

// Creating object of SudokuSolver

SudokuSolver ss2 = new SudokuSolver();

// Using the setter to add matrix to ss2

ss2.setMatrix(matrix3);

// Solving Sudoku

if (ss2.attemptValues(0, 0) == true)

{

// Printing Sudoku matrix with static method

SudokuPrinter.printMatrix(ss2.getMatrix());

}

else

{

System.out.println("No Solution exists");

}

break;

// For checking the user's provided Sudoku

case 4:

// Declaring a new Sudoku matrix

int[][] matrix4 = new int[9][9];

System.out.println("Please provide your 9 x 9 matrix.");

System.out.println("Give out space between each column element and enter after

each row.");

System.out.println("You can enter now:");

// Storing each user integer into our matrix array

for (int row = 0; row < 9; row++)

{

for (int column = 0; column < 9; column++)

{

matrix4[row][column] = in.nextInt();

}

}

// Checking Sudoku with static method

if(SudokuChecker.checkMatrix(matrix4) == true)

{

System.out.println("Congratulations! You solved it.");

}

else

{

System.out.println("Opps! You missed this time.");

System.out.println("You can try our Sudoku Solver to get your answer.");

}

break;

// Manipulating the value of flag to exit

case 5:

flag = 0;

break;

// Giving the default message

default:

System.out.println("Please enter a value from 1-5.");

}

}

}

}

**For SudokuSolver Class:**

Input:

/\*\* Application Purpose: To create a class called SudokuSolver which contains the logic to

solve a user provided Sudoku question.

Author: Priyansh Thakar

Date: 03 April 2021

Time: 03:44 PM

\*/

public class SudokuSolver

{

// Declaring a private 2d matrix

private int[][] matrix = new int[9][9];

// Declaring a constant for highest value

private final int HIGH = 9;

// Creating Constructor that takes no arguments

public SudokuSolver()

{

for (int row = 0; row < HIGH; row++)

{

for (int column = 0; column < HIGH; column++)

{

this.matrix[row][column] = 0;

}

}

}

// Creating Constructor that takes one matrix argument

public SudokuSolver(int[][] matrix)

{

this.matrix = matrix;

}

// Setter method for matrix

public void setMatrix(int[][] matrix)

{

this.matrix = matrix;

}

// Getter method for matrix

public int[][] getMatrix()

{

return matrix;

}

// Method to solve matrix with trials

public boolean attemptValues(int row, int column)

{

// Returning true if we reach the end to stop

if (row == HIGH - 1 && column == HIGH)

{

return true;

}

// Increasing row and initializing column if we reach end of row

if (column == HIGH)

{

row++;

column = 0;

}

// Going ahead until 0 is not discovered

if (matrix[row][column] != 0)

{

return attemptValues(row, column + 1);

}

// Adding proper values at 0 places

for (int i = 1; i < 10; i++)

{

// If value is correct

if (isProper(row, column, i) == true)

{

// Storing the new proper value

matrix[row][column] = i;

// Returning true if conditions satisfy

if (attemptValues(row, column + 1) == true)

{

return true;

}

}

// Keeping 0 if value not found

matrix[row][column] = 0;

}

// Returning false if not any condition matched

return false;

}

// A private method to return true if ans is correct

private boolean isProper(int row, int column, int num)

{

// Check the row for duplicate

for (int i = 0; i <= 8; i++)

{

// Returning false if any duplicate found

if (matrix[row][i] == num)

{

return false;

}

}

// Check the column for duplicate

for (int i = 0; i <= 8; i++)

{

// Returning false if any duplicate found

if (matrix[i][column] == num)

{

return false;

}

}

// Check 3 x 3 grid for duplicate

int startRowPoint = (row - (row % 3));

int startColumnPoint = (column - (column % 3));

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

// Returning false if any duplicate found

if (matrix[i + startRowPoint][j + startColumnPoint] == num)

{

return false;

}

}

}

// Returning true if all clear

return true;

}

}

**For SudokuQuestion Class:**

Input:

/\*\* Application Purpose: To create a class called SudokuQuestion which gives/generates a

Sudoku matrix to be solved by the user.

Author: Priyansh Thakar

Date: 03 April 2021

Time: 02:50 PM

\*/

// Importing to use random generator

import java.util.Random;

public class SudokuQuestion

{

// Declaring a private 2d matrix

private int[][] matrix = new int[9][9];

// Creating a private method to store our questions

private int[][] matrixStorage(int value)

{

// Declaring a matrix to store the questions

int[][] question = new int[9][9];

// For random value 0

if(value == 0)

{

int[][] question0 = {{0, 0, 0, 2, 6, 0, 7, 0, 1},

{6, 8, 0, 0, 7, 0, 0, 9, 0},

{1, 9, 0, 0, 0, 4, 5, 0, 0},

{8, 2, 0, 1, 0, 0, 0, 4, 0},

{0, 0, 4, 6, 0, 2, 9, 0, 0},

{0, 5, 0, 0, 0, 3, 0, 2, 8},

{0, 0, 9, 3, 0, 0, 0, 7, 4},

{0, 4, 0, 0, 5, 0, 0, 3, 6},

{7, 0, 3, 0, 1, 8, 0, 0, 0}};

question = question0;

}

// For random value 1

else if(value == 1)

{

int[][] question1 = {{1, 0, 0, 4, 8, 9, 0, 0, 6},

{7, 3, 0, 0, 0, 0, 0, 4, 0},

{0, 0, 0, 0, 0, 1, 2, 9, 5},

{0, 0, 7, 1, 2, 0, 6, 0, 0},

{5, 0, 0, 7, 0, 3, 0, 0, 8},

{0, 0, 6, 0, 9, 5, 7, 0, 0},

{9, 1, 4, 6, 0, 0, 0, 0, 0},

{0, 2, 0, 0, 0, 0, 0, 3, 7},

{8, 0, 0, 5, 1, 2, 0, 0, 4}};

question = question1;

}

// For random value 2

else if(value == 2)

{

int[][] question2 = {{0, 2, 0, 6, 0, 8, 0, 0, 0},

{5, 8, 0, 0, 0, 9, 7, 0, 0},

{0, 0, 0, 0, 4, 0, 0, 0, 0},

{3, 7, 0, 0, 0, 0, 5, 0, 0},

{6, 0, 0, 0, 0, 0, 0, 0, 4},

{0, 0, 8, 0, 0, 0, 0, 1, 3},

{0, 0, 0, 0, 2, 0, 0, 0, 0},

{0, 0, 9, 8, 0, 0, 0, 3, 6},

{0, 0, 0, 3, 0, 6, 0, 9, 0}};

question = question2;

}

// For random value 3

else if(value == 3)

{

int[][] question3 = {{0, 0, 0, 6, 0, 0, 4, 0, 0},

{7, 0, 0, 0, 0, 3, 6, 0, 0},

{0, 0, 0, 0, 9, 1, 0, 8, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 5, 0, 1, 8, 0, 0, 0, 3},

{0, 0, 0, 3, 0, 6, 0, 4, 5},

{0, 4, 0, 2, 0, 0, 0, 6, 0},

{9, 0, 3, 0, 0, 0, 0, 0, 0},

{0, 2, 0, 0, 0, 0, 1, 0, 0}};

question = question3;

}

// For random value 4

else if(value == 4)

{

int[][] question4 = {{2, 0, 0, 3, 0, 0, 0, 0, 0},

{8, 0, 4, 0, 6, 2, 0, 0, 3},

{0, 1, 3, 8, 0, 0, 2, 0, 0},

{0, 0, 0, 0, 2, 0, 3, 9, 0},

{5, 0, 7, 0, 0, 0, 6, 2, 1},

{0, 3, 2, 0, 0, 6, 0, 0, 0},

{0, 2, 0, 0, 0, 9, 1, 4, 0},

{6, 0, 1, 2, 5, 0, 8, 0, 9},

{0, 0, 0, 0, 0, 1, 0, 0, 2}};

question = question4;

}

// Returning the question matrix

return question;

}

// Creating a public method to generate Sudoku matrix

public int[][] generatePuzzle()

{

// Creating an object of Random class

Random rand = new Random();

// Generating one random number with length 0-5

int randomNumber1 = rand.nextInt(5);

// Getting a question matrix with a random number

matrix = matrixStorage(randomNumber1);

// Returning the Question matrix

return matrix;

}

}

**For SudokuChecker Class:**

Input:

/\*\* Application Purpose: To create a class called SudokuChecker which checks the Sudoku

matrix for problems.

Author: Priyansh Thakar

Date: 05 April 2021

Time: 03:59 PM

\*/

public class SudokuChecker

{

// A static method to check the matrix of 9 x 9

public static boolean checkMatrix(int[][] matrix)

{

// To check the row for duplicates

for (int row = 0; row < 9; row++)

{

for (int column = 0; column < 8; column++)

{

for (int nextColumn = column + 1; nextColumn < 9; nextColumn++)

{

// Returning false if any duplicate found

if (matrix[row][column] == matrix[row][nextColumn])

{

return false;

}

}

}

}

// To check the column for duplicates

for (int column = 0; column < 9; column++)

{

for (int row = 0; row < 8; row++)

{

for (int nextRow = row + 1; nextRow < 9; nextRow++)

{

// Returning false if any duplicate found

if (matrix[row][column] == matrix[nextRow][column])

{

return false;

}

}

}

}

// To check each 3 x 3 grid for duplicates

for(int row = 0; row < 9; row += 3)

{

for (int col = 0; col < 9; col += 3)

{

// Pointing the top left-most point of each 3 x 3 grid

for (int point = 0; point < 8; point++)

{

for (int nextPoint = point + 1; nextPoint < 9; nextPoint++)

{

// Returning false if any duplicate found

if (matrix[row + point%3][col + point/3] == matrix[row + nextPoint%3][col +

nextPoint/3])

{

return false;

}

}

}

}

}

// Returning true if all clear

return true;

}

}

**For SudokuPrinter Class:**

Input:

/\*\* Application Purpose: To create a class called SudokuPrinter which prints the pre-defined

data or solution matrix on the console.

Author: Priyansh Thakar

Date: 03 April 2021

Time: 06:48 PM

\*/

public class SudokuPrinter

{

// A static method to print the matrix in 9 x 9 way

public static void printMatrix(int[][] matrix)

{

// Keeping in try block to check for exceptions

try

{

// Looping for the account of rows

for (int i = 0; i < 9; i++)

{

// Looping for the account of columns

for (int j = 0; j < 9; j++)

{

// Printing the value with spaces

System.out.print(matrix[i][j] + " ");

}

// Printing a new line for next row

System.out.println();

}

}

// Catching the NullPointerException

catch (NullPointerException e)

{

System.out.println("Null Pointer Exception occurred.");

System.out.println("Please Enter not-null values.");

}

// Catching the IndexOutOfBoundsException

catch (IndexOutOfBoundsException e)

{

System.out.println("Index out of Bounds Exception occurred.");

System.out.println("Please Enter integer values from 0-9.");

}

}

}

Output:













