**MODULE CODE**: SOFT10101

**MODULE NAME**: COMPUTER SCIENCE PROGRAMMING 201819 FULL YEAR

**NAME**: DAYEETA DAS

**NTU ID**: N0830182

**COURSE**: BSc COMPUTER SCIENCE(Hons) SW

**YEAR**: 1ST YEAR

**GROUP**: COMP 7

**TILTE OF THE PROJECT UNDERTAKEN**:

4 X 4 SUDOKU GAME TUTORIAL IN C++ CONSOLE

**SOFTWARE USED**: VISUAL STUDIO IDE

**PROGRAMMING LAGUAGE** **USED**:

C++

AIM OF THE PROJECT **4 X 4 SUDOKU GAME TUTORIAL**

The aim of the project **4 X 4 Sudoku Game Tutorial** is to help the user to solve the Sudoku puzzle which in this case comprises of a board having spaces and numbers from 1-4 have been scattered in a few places in the board from the start.

This program enables the user to solve the puzzle by taking the position on the board and the number that he/she wants to place in that position as an input. It goes through all possible checks to avoid the errors while taking the input. For instance: if the user enters a number which is beyond the range or a number which is already present in the particular row/column(hence cannot be repeated) the computer triggers the user to enter another number till he/she enters a number that will solve the puzzle. However, this program has only one predefined game-board and does not generate a random Sudoku board every time it is compiled.

The program also makes use of various C++ features such as **Class, data structure, 2 D arrays, functions, iteration and branching statements and file handling.**

REQUIREMENTS OF THE PROGRAM

Here are a list of all the different elements( constants, variables, functions and so on) that I have used within the program:

1. **Header files used**: <iostream>: , <ctype.h>, <string>, <fstream>
2. **Identifiers used(along with their data types):**

* Character type: press, num, board[4][4]
* Integer type: i, j, pos, n1, n2, c, position[4][4]
* String type: name, readcontent

1. **Data Structure** :

* Name: Person
* Elements: p1.name(string), p1.date(string), p1.rate(character)

1. **Class** :

* Name: Sudoku
* Private members: None
* Public members: void instructions()

void menu()

void input\_press();

void check\_press();

void press\_function();

void details();

void positions();

void draw\_board();

void input\_position();

void input\_number();

int game\_over();

1. **Other data types used (with identifier name):**

* Ofstream myfile
* ifstream myfileread
* bool go(default value true)

1. **Statements used:**

* Branching: if-else
* Iteration: for loop, while loop, goto statement
* Input / Output statements: cin, cout, cin.clear(), cin.ignore()
* File handling statements: myfile,open(), myfile.close()
* Break satetement

1. **Labels used:**

* Repeat
* loop

FUNCTIONALITY OF THE PROGRAM (WORKING)

* The necessary header files are included at the beginning of the program and the different variables and constants are defined which have been used throughout the program.
* The **class Sudoku** has been initialized with all the **public members**( functions).

1. The instructions for the tutorial gets displayed on the screen.
2. The name of the user is taken as input and stored in a file **name.txt**.
3. A board displaying the different positions on the game board gets displayed.
4. The game board gets displayed.
5. The position that the user wants to place the number in is taken as input. Checks are performed to make sure that the input is only a number within **1- 16**. If it is beyond this range or a special character, the input is taken again and again till it is valid. Checks are performed to make sure that the particular location is empty, if empty the row and column of the place is stored in **n1** and **n2** respectively. Once a position gets occupied, the value of that position turns to zero in the position matrix to make sure it cannot be used again.
6. Similar to the positions, the number is taken as input, if the number is not any of these characters **–‘1’,’2’,’3’ or ’4’**, or if the number is already present in that particular row/column(repeat), input is taken till it is valid.
7. If the number taken as input is valid, it gets placed on the game board at the location **board[n1][n2]**. The function **draw\_board**() is called.
8. The **game\_over()** function checks if any more **‘\_’** blanks are left in the board. If blanks are there, **c=1 else c=0**.

* All of these 8 steps are looped in a while loop till the value of c returned by the game\_over() function is zero(that is c=0).
* When the game is over, a menu is displayed and the user is asked to press any of the three numbers from the menu.
* Input is taken from the user till he/she presses a valid number.

1. If the user presses **‘1’** then the user is asked to input his/her name, date and asked to give a rating to this tutorial on a scale of 5. If the user enters a number other than **‘1’,’2’,’3’,’4’ or ‘5’** or any other special character, the input is taken till it is valid.
2. These details are stored in a text file **‘review.txt’**.
3. If the user presses **‘2’**, then the contents of the file **review.txt** is read and displayed on the screen.
4. If the user presses **‘0’**, the program ends.

All of these 4 steps are performed till the user presses ‘0’.

FLOWCHART ILLUSTRATING THE WORKING OF THE PROGRAM



TESTS UNDERTAKEN TO FOR MAKING THE PROGRAM EFFICIENT

* In the **void input\_position()** function the following checks are made:

1. The input is taken till the user enters a number in range 1-16 in a loop.
2. If the user enters a character or a string then those are ignored and the loop goes back to repeat: and asks for an input again.
3. A check is made to see if the current position is empty this is done by replacing an empty position with 0 when it is chosen by the user. If it is not empty, it asks for an input again.

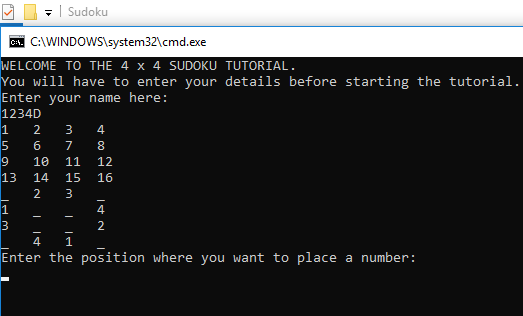
* In the **void input\_number()** function the following checks are made:

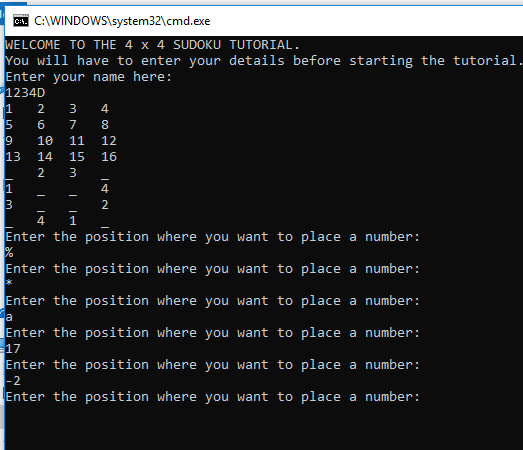
1. If the user enters a character outside the range of 1-4, the input is taken.
2. If the user enters any special character or any alphabet the input is taken till it becomes a valid input.
3. If the number chosen is present in the particular row/column, the input is taken till there are no repeats.

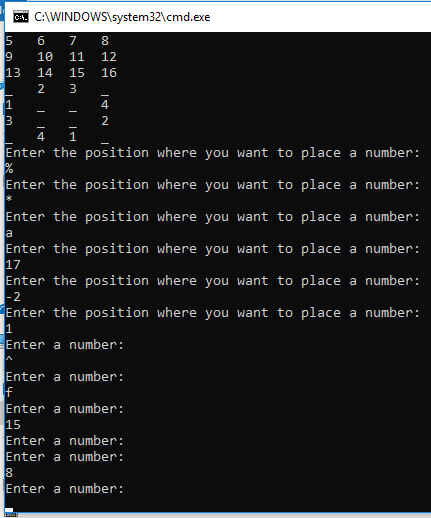
* In the **void check\_press()** function,

1. If the user enters any characters beyond ‘1’,’2’ and ‘0’ the input is taken.
2. If the user enters a special character or an alphabet then again the loop continues till the user enters a valid input.

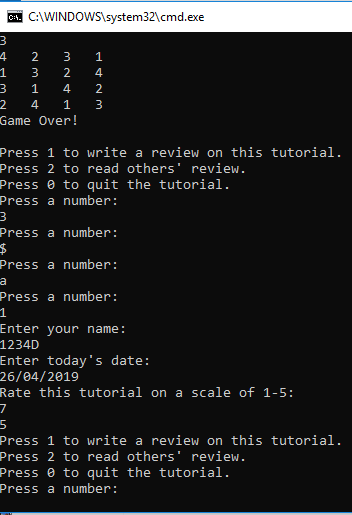
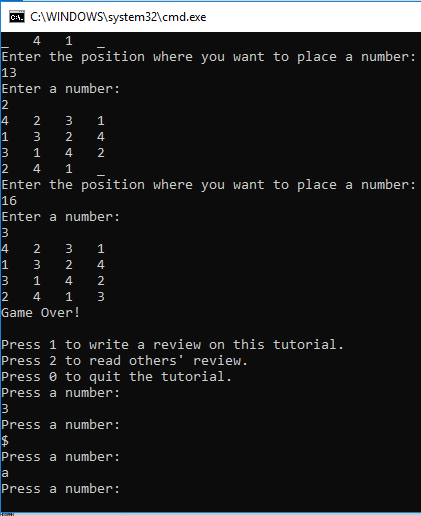
The name entered can be alphanumeric if the user wishes to not to reveal his/her identity



The position will be asked for as input till it is valid

The number will be taken as input till it is valid

The number for the menu will be asked till

it is valid  


The rating is asked as input till it is valid

CRITIQUE ON PROGRAM **4 X 4 SUDOKU TUTORIAL**

* This particular can be improved further by generating an automated Sudoku board.
* At the moment this is only a single time game but if an automated board could be generated then the user could do the tutorial.
* Another option had to be added to the menu in that case that would allow the user to go back and play and a new game.
* The program takes the name of the user in the format of a string and this has been done on purpose because some players might not want to reveal their real names and make up a name which might not just contain alphabets but might be alphanumeric.
* A check could be put up to the previous aspect to accept only valid names.

Last but not the least I would like to conclude the report by stating the fact that every project can be improved further with new ideas and by correcting any new faults that can come to notice.