PROJECT REPORT

On

“Python Sudoku Game”

Submitted By

Ruchika Ashok Chindhalore

Guided By:-

Mr. Ratnesh K. Choudhary

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

S. B. JAIN INSTITUTE OF TECHNOLOGY

MANAGEMENT AND RESEARCH, NAGPUR.

(An Autonomous Institute, Affiliated to RTMNU, Nagpur)

2021-2022

S.B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT AND RESEARCH, NAGPUR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SESSION 2021-2022

CERTIFICATE

This is to certify that the Project titled **"Python Sudoku Game"** is a bonafide work of **Ruchika Ashok Chindhalore** carried out for the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering in **Computer Science & Engineering**

Mr. Ratnesh K. Choudhary
Assistant Professor

Mr. Animesh Tayal
Head of Department

INDEX

LIST OF FIGURE

CHAPTER 1
INTRODUCTION

Sudoku is the Japanese abbreviation of a phrase meaning the digits must remain single, also known as Number Place, where Su means number, Sudoku which translates as single or bachelor.

Sudoku is not a mathematical or arithmetical puzzle. It works just as well if the numbers are substituted with letters or some other symbols, but numbers work best. The aim of the puzzle is to enter a numerical digit from 1 through 9 in each cell of a 9x9 grid made up of 3x3 sub squares or sub grids, starting with various digits given in some cells; each row, column, and sub squares region must contain each of the numbers 1 to 9 exactly once.

Throughout this document we refer to the whole puzzle as the grid/game board, a 3x3 sub grid as block and the individual grids that contains the number as a cell.

CHAPTER 2

METHODOLOGY

2.1 SYSTEM ARCHITECTURE

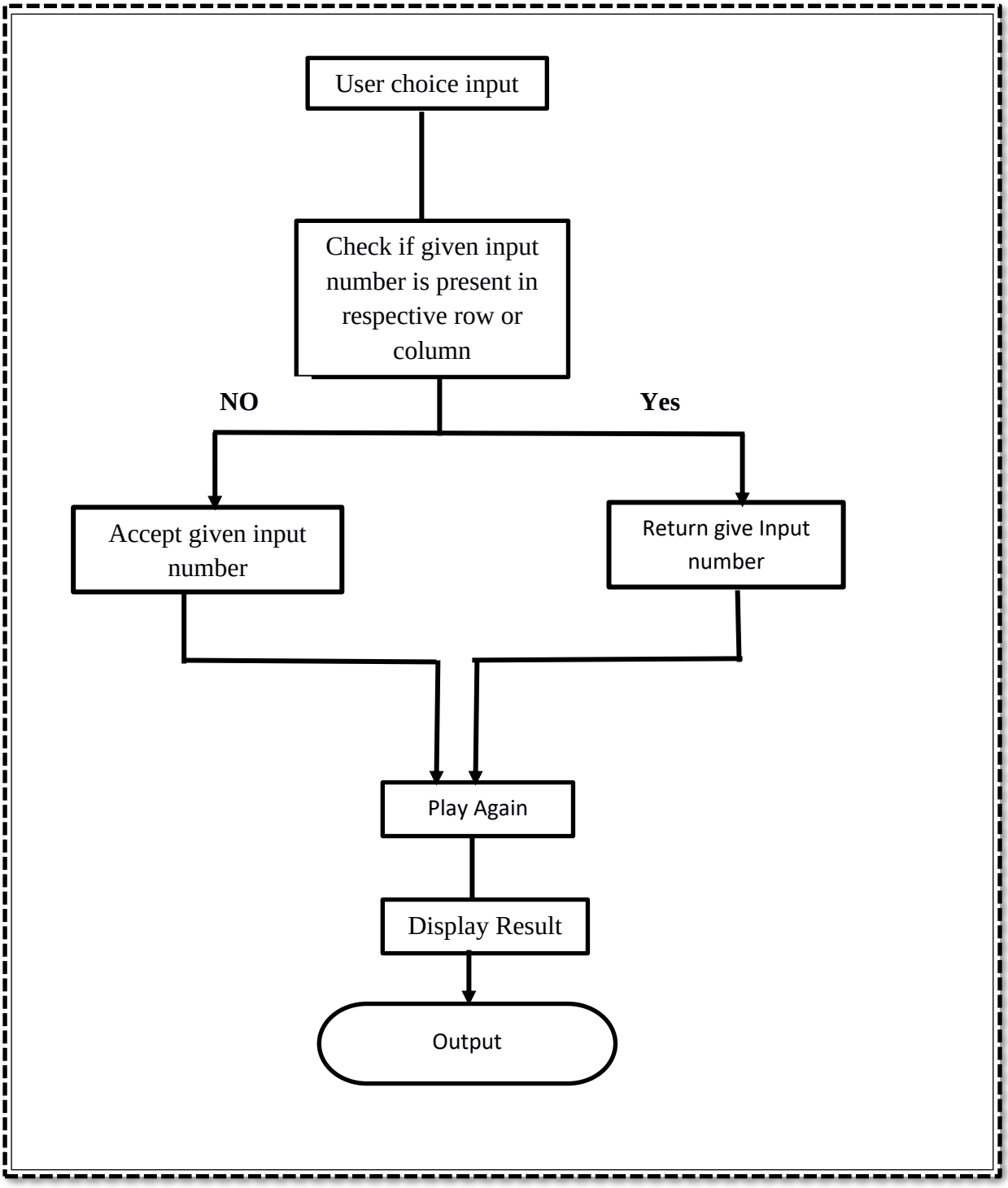

Fig 2.1. System Architecture

CHAPTER 3

TOOLS/PLATFORMS

3.1 SOFTWARE REQUIREMENT

a) **CLIENT-SIDE TECHNOLOGY:** Python

b) **LIBRARIES:** Pygame

c) **IDE / FRAMEWORK:** Visual Studio

d) **OPERATING SYSTEM:** Windows 10

1. Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning Python:

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

2. Pygame

The pygame library is an open-source module for the Python programming language specifically intended to help you make games and other multimedia applications. Built on top of the highly portable SDL (Simple DirectMedia Layer) development library, pygame can run across many platforms and operating systems

- Pygame is a cross-platform set of Python modules which is used to create video games.
- It consists of computer graphics and sound libraries designed to be used with the Python programming language.
- Pygame was officially written by Pete Shinners to replace PySDL.
- Pygame is suitable to create client-side applications that can be potentially wrapped in a standalone executable.

3.Visual Studio Code:

- Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python, C++ and Fortran. It is based on the Electron framework, which is used to develop Node.js Web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).
- Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, Typescript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace.

Fig 3.1.1 Visual Studio Code

- Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports many programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette.

CHAPTER 4

DESIGN & IMPLEMENTATION

4.1 ALGORITHM

Step 1: Start

Step 2: Installation of Pygame module

Step 3: Initializing Sudoku game window and variables

Step 4: Function for highlighting selected cell

Step 5: Function to draw lines for making sudoku grid

Step 6: Function to fill value in the cell

Step 7: Function for raising error when wrong value is entered

Step 8: Function to check if the entered value is valid

Step 9: Function to solve Sudoku game

Step 10: Function to show result

Step 11: Rest code

Step 12: End.

4.2 FLOWCHART

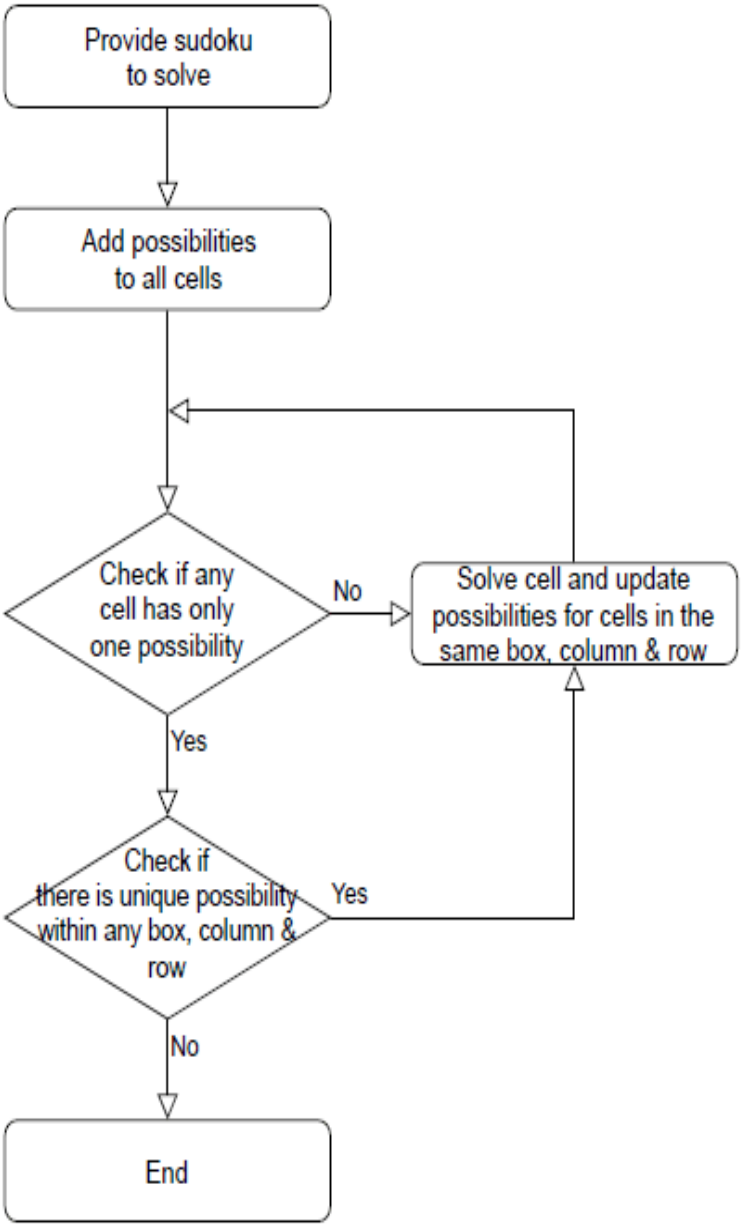

Fig. 4.2.1 Flowchart

4.3 SOURCE CODE

```
import pygame
pygame.font.init()
Window = pygame.display.set_mode((500, 500))
pygame.display.set_caption("SUDOKU GAME -BY RUCHIKA")
x = 0
z = 0
y = 0
diff = 500 / 9
```

```
value= 0
defaultgrid =[
      [7, 8, 0, 4, 0, 0, 1, 2, 0],
       [6, 0, 0, 0, 7, 5, 0, 0, 9],
       [0, 0, 0, 6, 0, 1, 0, 7, 8],
       [0, 0, 7, 0, 4, 0, 2, 6, 0],
       [0, 0, 1, 0, 5, 0, 9, 3, 0],
       [9, 0, 4, 0, 6, 0, 0, 0, 5],
       [0, 7, 0, 3, 0, 0, 0, 1, 2],
       [1, 2, 0, 0, 0, 7, 4, 0, 0],
       [0, 4, 9, 2, 0, 6, 0, 0, 7]   ]
font = pygame.font.SysFont("comicsans", 30)
font1 = pygame.font.SysFont("comicsans", 10)
 def cord(pos):
   global x
   x = pos[0]//diff
   global z
   z = pos[1]//diff
 #Function for highlighting selected cell
def highlightbox():
   for k in range(2):
      pygame.draw.line(Window, (0, 0, 0), (x * diff-3, (z + k)*diff), (x * diff + diff + 3, (z +
k)*diff), 7)
      pygame.draw.line(Window, (0, 0, 0), ( (x + k)* diff, z * diff ), ((x + k) * diff, z * diff +
diff), 7)
#Function to draw lines for making sudoku grid
def drawlines():
   for i in range (9):
      for j in range (9):
         if defaultgrid[i][j]!= 0:
            pygame.draw.rect(Window, (0,128,128), (i * diff, j * diff, diff + 1, diff + 1))
            text1 = font.render(str(defaultgrid[i][j]), 1, (0, 0, 0))
            Window.blit(text1, (i * diff + 15, j * diff + 15))
   for l in range(10):
      if l % 3 == 0 :
```

```
            thick = 7
        else:
            thick = 1
        pygame.draw.line(Window, (0, 0, 0), (0, l * diff), (500, l * diff), thick)
        pygame.draw.line(Window, (0, 0, 0), (l * diff, 0), (l * diff, 500), thick)
#Function to fill value in the cell
def fillvalue(value):
    text1 = font.render(str(value), 1, (0, 0, 0))
    Window.blit(text1, (x * diff + 15, z * diff + 15))
#Function for raising error when wrong value is entered
def raiseerror():
    text1 = font.render("wrong!", 1, (0, 0, 0))
    Window.blit(text1, (20, 570))
    #Function to check if the entered value is valid
def validvalue(m, k, l, value):
    for it in range(9):
        if m[k][it]== value:
            return False
        if m[it][l]== value:
            return False
    it = k//3
    jt = l//3
    for k in range(it * 3, it * 3 + 3):
        for l in range (jt * 3, jt * 3 + 3)
if m[k][l]== value:
     return False
    return True
      #Function to solve sudoku game
def solvegame(defaultgrid, i, j):
    while defaultgrid[i][j]!= 0:
        if i<8:
            i+= 1
        elif i == 8 and j<8:
            i = 0
            j+= 1
```

```
            elif i == 8 and j == 8:
                return True
        pygame.event.pump()
        for it in range(1, 10):
            if validvalue(defaultgrid, i, j, it)== True:
                defaultgrid[i][j]= it
                global x, z
                x = i
                z = j
                Window.fill((255,165,0))
                drawlines()
                highlightbox()
                pygame.display.update()
                pygame.time.delay(20)
                if solvegame(defaultgrid, i, j)== 1:
                    return True
                else:
                    defaultgrid[i][j]= 0
                Window.fill((0,0,0))
                drawlines()
                highlightbox()
                pygame.display.update()
                pygame.time.delay(50)
        return False
    #Function to show result
    def gameresult():
        text1 = font.render("Game Over", 1, (0, 0, 0))
        Window.blit(text1, (20, 570))
    flag=True
    flag1 = 0
    flag2 = 0
    rs = 0
    error = 0
#Rest Code
    while flag:
```

```
Window.fill((0,0,128))
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        flag = False
    if event.type == pygame.MOUSEBUTTONDOWN:
        flag1 = 1
        pos = pygame.mouse.get_pos()
        cord(pos)
    if event.type == pygame.KEYDOWN:
        if event.key == pygame.K_LEFT:
            x-= 1
            flag1 = 1
        if event.key == pygame.K_RIGHT:
            x+= 1
            flag1 = 1
        if event.key == pygame.K_UP:
            y-= 1
            flag1 = 1
        if event.key == pygame.K_DOWN:
            y+= 1
            flag1 = 1
        if event.key == pygame.K_1:
            value = 1
        if event.key == pygame.K_2
        value = 2
        if event.key == pygame.K_3:
            value = 3
        if event.key == pygame.K_4:
            value = 4
        if event.key == pygame.K_5:
            value = 5
        if event.key == pygame.K_6:
            value = 6
        if event.key == pygame.K_7:
            value = 7
```

```
if event.key == pygame.K_8:
    value = 8
if event.key == pygame.K_9:
    value = 9
if event.key == pygame.K_RETURN:
    flag2 = 1
if event.key == pygame.K_r:
    rs = 0
    error = 0
    flag2 = 0
    defaultgrid=[
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0]
    ]
if event.key == pygame.K_d:
    rs = 0
    error = 0
    flag2 = 0
    defaultgrid  =[
        [0, 0, 4, 9, 1, 0, 0, 0, 2],
        [1, 9, 0, 4, 0, 0, 0, 7, 0],
        [0, 0, 7, 6, 0, 3, 0, 4, 8],
        [5, 3, 0, 0, 8, 2, 0, 0, 4],
        [0, 0, 2, 4, 6, 0, 0, 8, 3],
        [0, 0, 1, 3, 9, 0, 6, 2, 0],
        [0, 1, 0, 2, 0, 0, 0, 6, 0],
        [0, 4, 0, 5, 0, 8, 0, 0, 1],
        [0, 5, 0, 0, 0, 6, 7, 0, 0],
```

```
                ]
    if flag2 == 1:
        if solvegame(defaultgrid , 0, 0)== False:
            error = 1
        else:
            rs = 1
        flag2 = 0
    if value != 0:
        fillvalue(value)
        if validvalue(defaultgrid , int(x), int(z), value)== True:
            defaultgrid[int(x)][int(z)]= value
            flag1 = 0
        else:
            defaultgrid[int(x)][int(z)]= 0
            raiseerror()
        value = 0
     if error == 1:
        raiseerror()
    if rs == 1:
        gameresult()
    drawlines()
    if flag1 == 1:
        highlightbox()
    pygame.display.update()
pygame.quit()
```

4.4 SYSTEM DESIGN

➢ 4.4.1 USE-CASE DIAGRAM

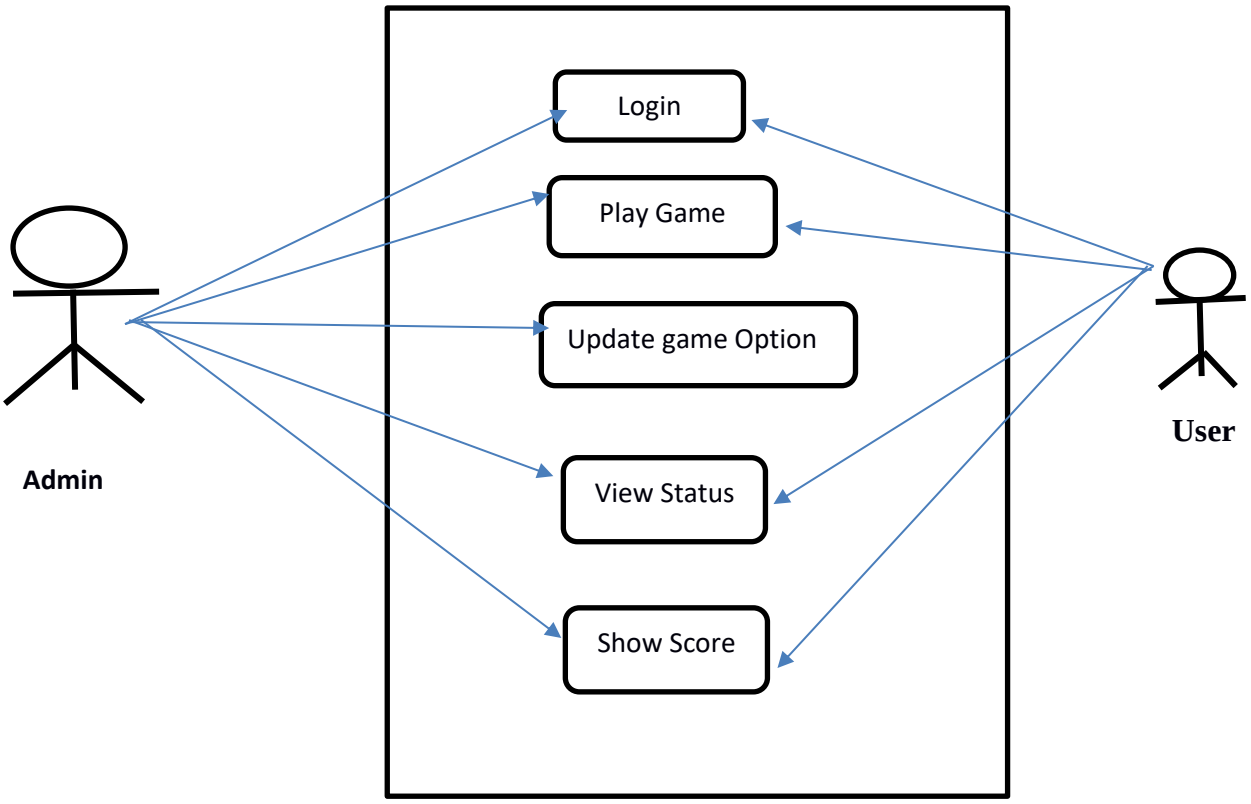

Fig.4.4.1 Use-Case

CHAPTER 5

RESULT & DISCUSSION

5.1 OUTPUT

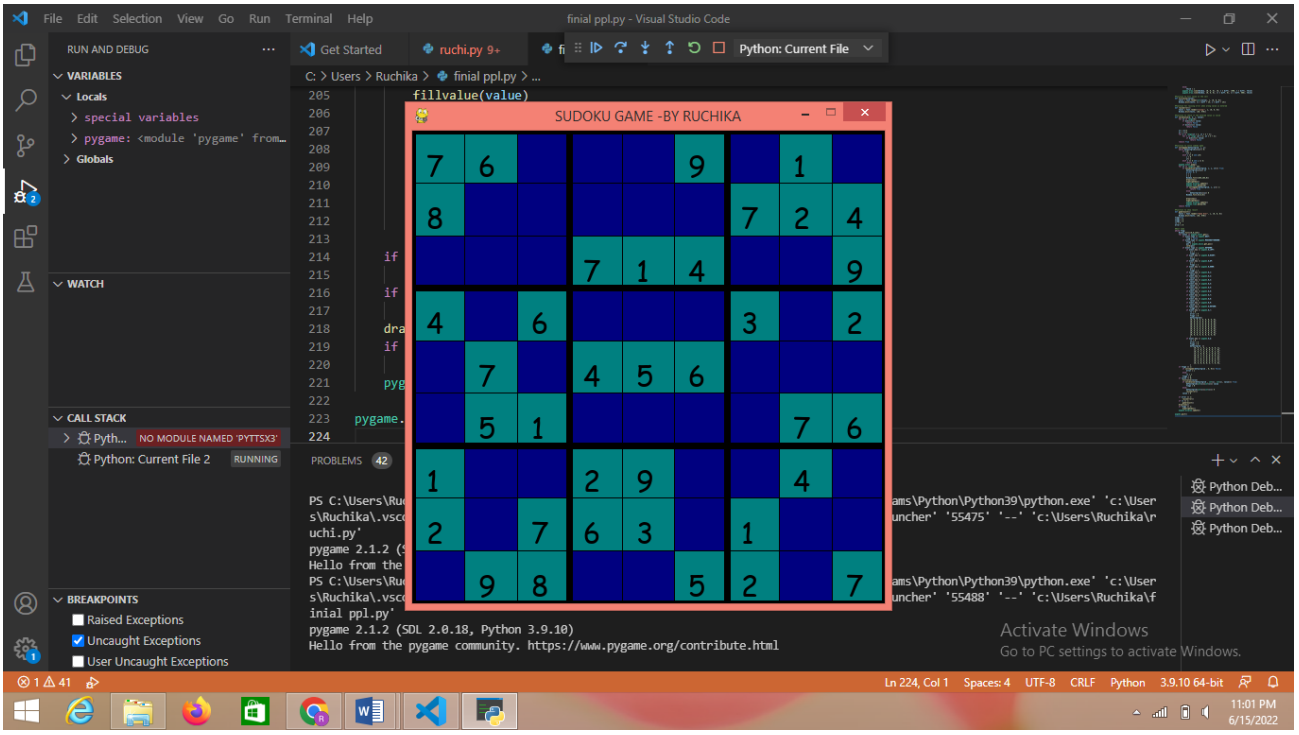

Fig 5.1.1 Sudoku Game

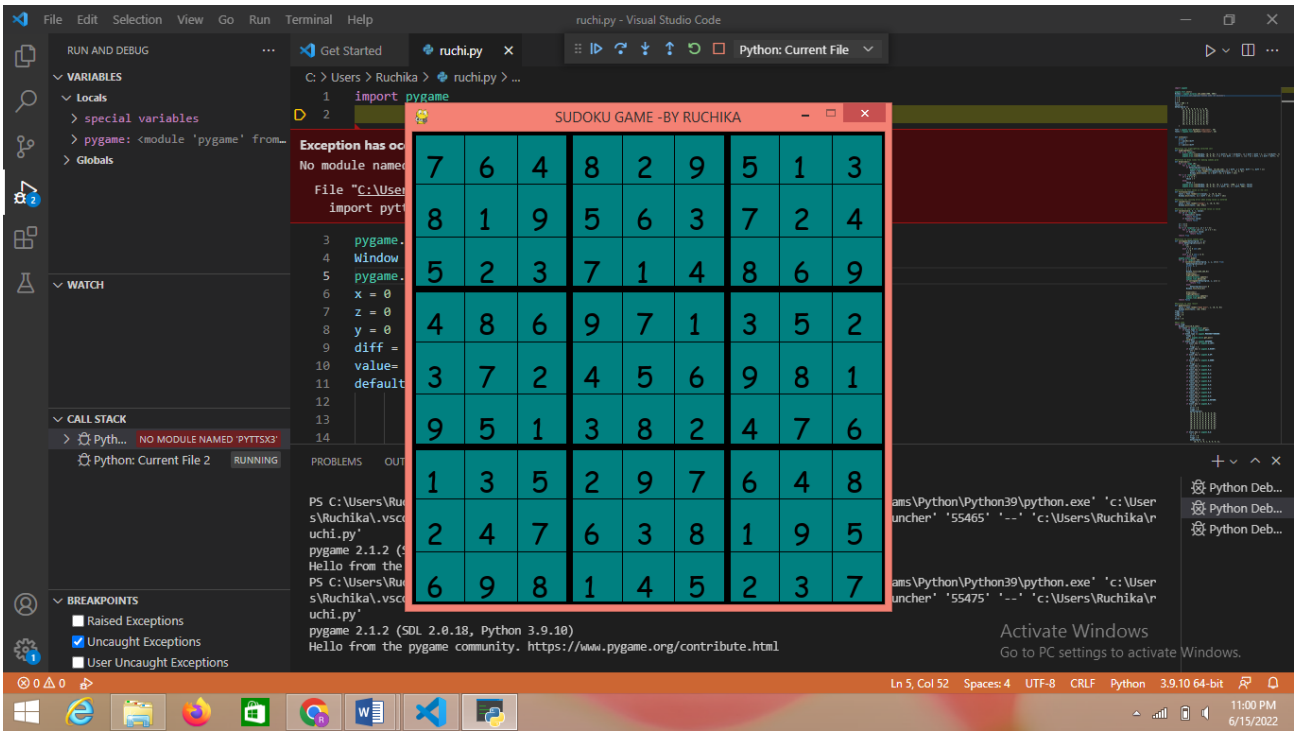

Fig 5.1.2 Sudoku Solved Puzzle

5.2 DISCUSSION

- **Sudoku Rule No 1: Use Numbers 1-9**

Sudoku is played on a grid of 9 x 9 spaces. Within the rows and columns are 9 "squares" (made up of 3 x 3 spaces). Each row, column and square (9 spaces each) needs to be filled out with the numbers 1-9, without repeating any numbers within the row, column or square. Does it sound complicated? As you can see from the image below of an actual Sudoku grid, each Sudoku grid comes with a few spaces already filled in; the more spaces filled in, the easier the game – the more difficult Sudoku puzzles have very few spaces that are already filled in.

- **Sudoku Rule No 2: Don't Repeat Any Numbers**

As you can see, in the upper left square (circled in blue), this square already has 7 out of the 9 spaces filled in. The only numbers missing from the square are 5 and 6. By seeing which numbers are missing from each square, row, or column, we can use process of elimination and deductive reasoning to decide which numbers need to go in each blank space.

For example, in the upper left square, we know we need to add a 5 and a 6 to be able to complete the square, but based on the neighboring rows and squares we cannot clearly deduce which number to add in which space. This means that we should ignore the upper left square for now, and try to fill in spaces in some other areas of the grid instead.

- **Sudoku Rule No 3: Don't Guess**

Sudoku is a game of logic and reasoning, so you shouldn't have to guess. If you don't know what number to put in a certain space, keep scanning the other areas of the grid until you seen an opportunity to place a number. But don't try to "force" anything – Sudoku rewards patience, insights, and recognition of patterns, not blind luck or guessing

- **Sudoku Rule No 4: Use Process of Elimination**

What do we mean by using "process of elimination" to play Sudoku? Here is an example. In this Sudoku grid (shown below), the far left-hand vertical column (circled in Blue) is missing only a few numbers: 1, 5 and 6.

One way to figure out which numbers can go in each space is to use "process of elimination" by checking to see which other numbers are already included within each square – since there can be no duplication of numbers 1-9 within each square (or row or column).

In this case, we can quickly notice that there are already number 1s in the top left and center left squares of the grid (with number 1s circled in red). This means that there is only one space remaining in the far left column where a 1 could possibly go – circled in green. This is how the process of elimination works in Sudoku – you find out which spaces are available, which numbers are missing – and then deduce, based on the position of those numbers within the grid, which numbers fit into each space.

Sudoku rules are relatively uncomplicated – but the game is infinitely varied, with millions of possible number combinations and a wide range of levels of difficulty. But it's all based on the simple principles of using numbers 1-9, filling in the blank spaces based on deductive reasoning, and never repeating any numbers within each square, row or column.

5.3 APPLICATION

- Playing mental games and exercises may not stop Dementia in its tracks when you already have it, but studies have shown that brain exercise throughout life can help to build a healthy reserve of brain cells and connections. This can minimize the risk of Dementia.
- Sudoku makes you think – and think critically, and it can also help improve your concentration. Sudoku is a game that requires careful thought. You have to think about where you are going to be placing the numbers and if they are going to interfere with the numbers in other boxes and lines.
- . That being said, brain exercise is excellent for building reserves, which may compensate for damage that is caused by diseases such as Dementia. Start playing Sudoku early on in life to ensure that your brain is getting the workout it needs

CHAPTER 6
CONCLUSION

In this way, I have successfully developed **Sudoku game using python** in which I have used **pygame** library. Playing Sudoku improves memory & recall. Logical thinking and memory go hand-in-hand. As your logical thinking improves, you will start to remember specific strategies and recall what worked in previous Sudoku puzzles. This can be used to remember and recall things in other areas of life.

In conclusion Sudoku is a **game of logic and reasoning**, so you shouldn't have to guess. If you don't know what number to put in a certain space, keep scanning the other areas of the grid until you seen an opportunity to place a number. But don't try to "force" anything – **Sudoku rewards patience, insights, and recognition of patterns, not blind luck or guessing**.

I have also executed this game with different input scenario and observed the output. This games performs all tasks as per expectation.

REFERENCE

1. www.Google.com
2. www.w3schools.com
3. https://youtube.com
4. www.geeksforgeeks.org/