Program Name: (BCS) Hons

Course Code: CSC 3201

Course Nam: Artificial Intelligence

Individual Project

Date of Submission: 26th October 2021

Submitted By:

Student Name:Pradip Dhakal

IUKL ID: 042902900047

Semester: 4th

Intake: September 2019

Submitted To:

Faculty Name: Prakash Chandra

Department: PO/LMS

Abstract

Sudoku is a logic-based puzzle. It is a type of *constraint satisfaction problem*, where the solver is given a finite number of objects (the numerals 1-9) and a set of conditions stating how the objects must be placed in relation to one another. The puzzle consists of a 9×99\times 99×9 grid further divided into nine 3×33×33×3 sub-grids (also called boxes, blocks, regions, or sub-squares). Sudoku is a popular game that is played all across the world, not just in one country. However, as popular and simple as it may appear, it is more intricate and difficult than it appears. Even Alan Turing enjoyed completing sudoku puzzles, indicating that the game has long been a favorite among math and statistics enthusiasts. And for the past few decades, everyone has been obsessed with solving the Sudoku puzzle. The puzzle's ease of construction and little requirement for mathematics skills have enticed many people to take on the task of solving it. As a result, developers have attempted to invent strategies for generating a variety of difficulties for human players that may be solved by computer programming. In this project, we used the backtracking method to develop a sudoku solver. The goal is to develop a sudoku game solver that is both efficient and effective in solving difficult sudoku puzzles.

Introduction:

Sudoku, a popular Japanese puzzle game, is centered on the placing of numbers in a logical order. Even till today, Sudoku puzzles among people around the world and still is more popular throughout the world. This game is currently popular in many countries and many developers have tried to create even more complicated and entertaining challenges. Nowadays, practically all of the newspapers, publications and websites feature the game. This project presents a sudoku game that uses simple rules to solve problems. The back tracking algorithm is therefore implemented on the basis of human perceptions. The name of the solver is hence the sudoku solver. In order to test the efficiency of the suggested method, the back tracking algorithm is then utilized to compares with this algorithm. The back tracking algorithm is a broad algorithm, which can be used for solving many problems. This algorithm provides all possible solutions until the correct response is found. The next paragraphs cover the statement of problems, the goal of the project and the abbreviations and meanings.

Algorithm

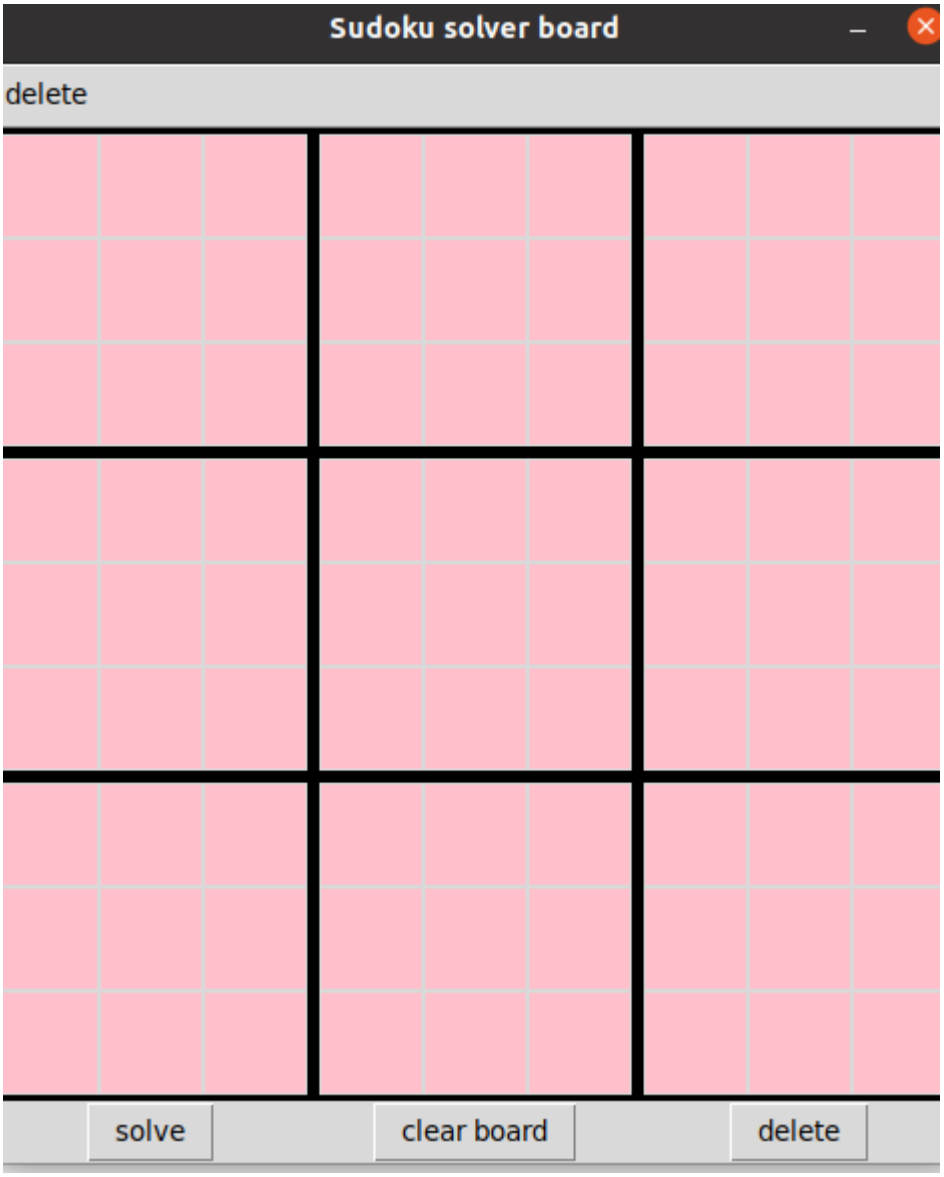
Backtracking is an algorithmic method for recursively solving problems by attempting to construct a solution one piece at a time and discarding any ideas that fail to meet the problem's requirements at any point in time. Backtracking is an algorithmic approach that allows you to solve a problem in a new way.
Algorithm steps

- Step 1: build a function that creates list
- Step 2: make a function to solve list

```
for i in range(0,9):
        for j in range(0,9):
```

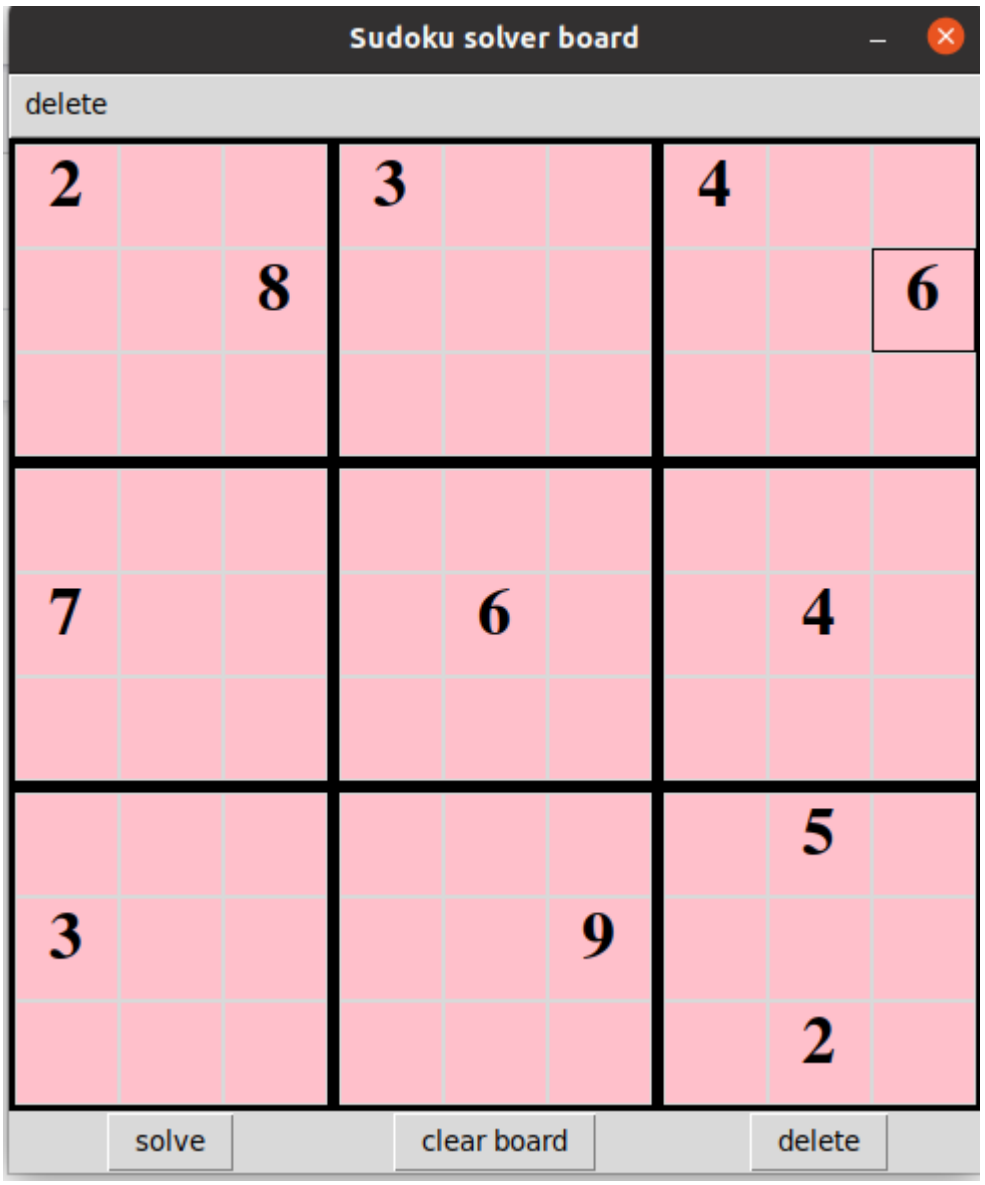
- Create function to check possible case
- Crate a function to correct list data or final result

Implementation
Run app.py file

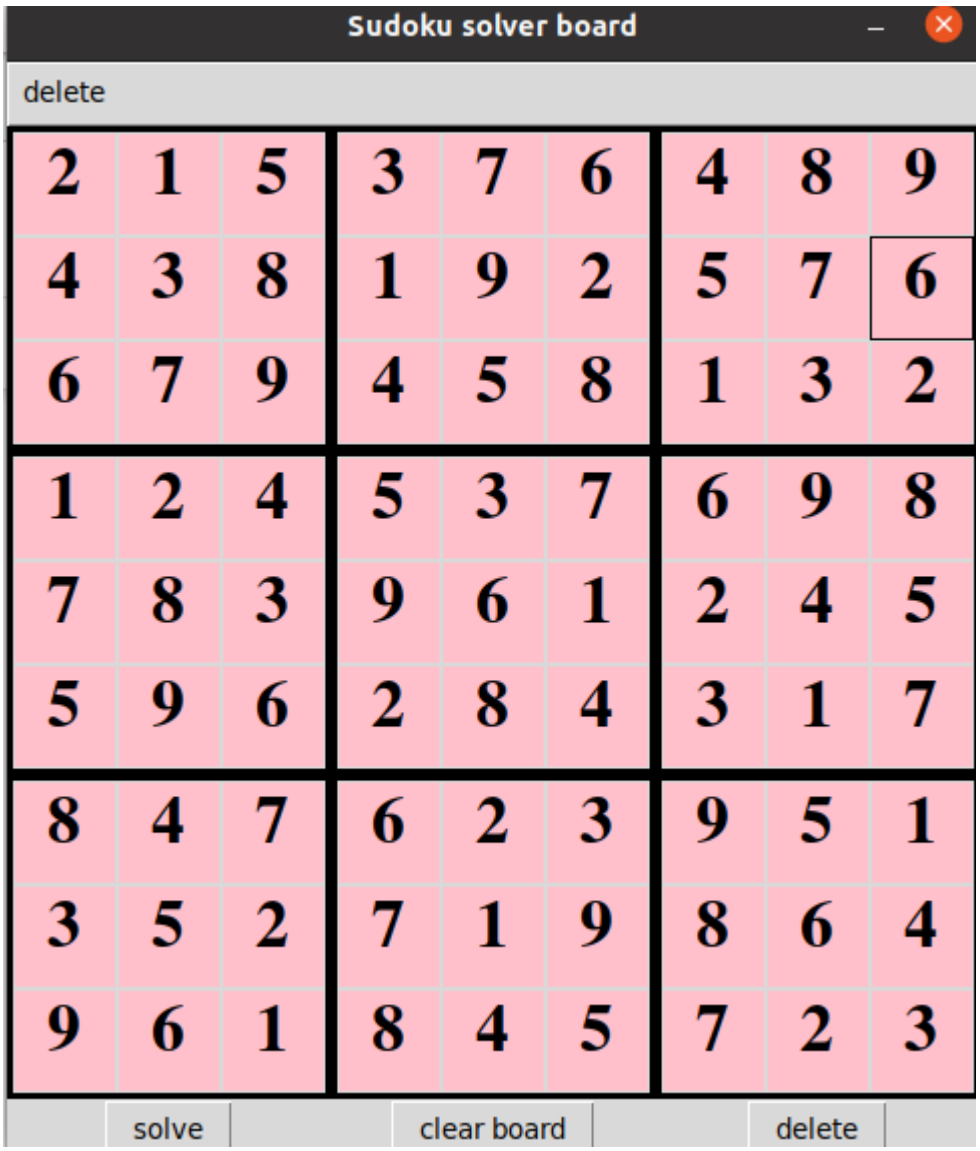


It will automatically solve game if we click solve button: here is the result

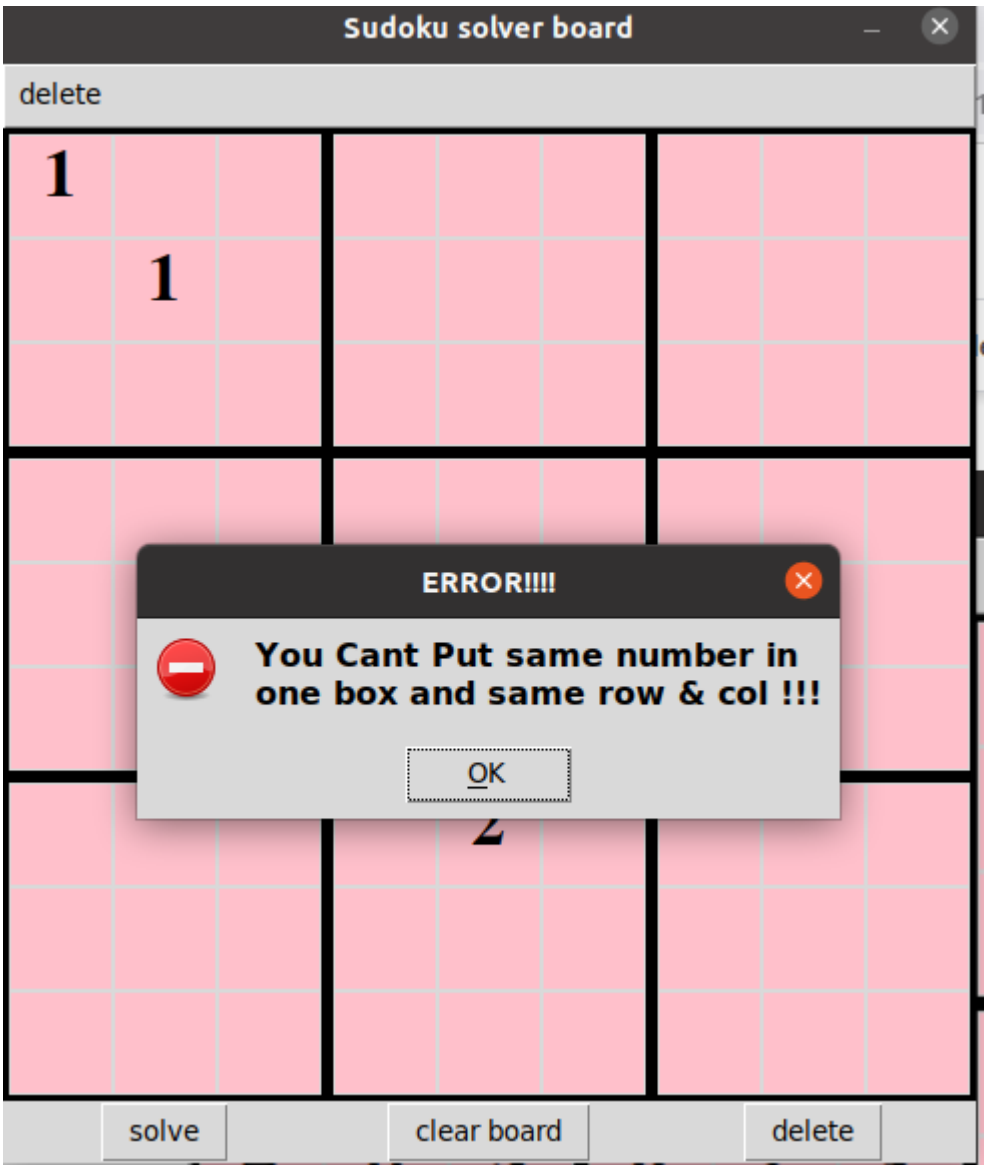
We can manually fill the number and solve them: here is the example



And press on solve button



If we put same number on same box or same row and same col it will show error like this



Conclusion:

As a result, a fast sudoku solver was created using a backtracking method, Python programming, and the Tkinter GUI package. The project meets the requirements, and the program runs smoothly. In this study, the backtracking algorithm, which is also a recursive one, was shown to be a viable strategy for solving Sudoku problems. In comparison to other methods, the algorithm is a good way to locate a solution more quickly and efficiently. In a short length of time, the provided technique may solve such riddles of any difficulty level (less than one second).The results of the tests demonstrated that the backtracking algorithm can be used in terms of computation time to answer sudoku puzzles.

Appendix

App.py (GUI)

```
import tkinter as tk
import tkinter.messagebox
import SudokoSolver
win = tk.Tk()
win.resizable(False, False)
win.title('Sudoku solver board')
number_list = [
        [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]
        ]
def display_numbers():
    global number_list
    global c0_0, c0_1, c0_2, c1_0, c1_1, c1_2, c2_0, c2_1, c2_2
    global c0_3, c0_4, c0_5, c1_3, c1_4, c1_5, c2_3, c2_4, c2_5
    global c0_6, c0_7, c0_8, c1_6, c1_7, c1_8, c2_6, c2_7, c2_8
    global c3_0, c3_1, c3_2, c4_0, c4_1, c4_2, c5_0, c5_1, c5_2
    global c3_3, c3_4, c3_5, c4_3, c4_4, c4_5, c5_3, c5_4, c5_5
    global c3_6, c3_7, c3_8, c4_6, c4_7, c4_8, c5_6, c5_7, c5_8
    global c6_0, c6_1, c6_2, c7_0, c7_1, c7_2, c8_0, c8_1, c8_2
    global c6_3, c6_4, c6_5, c7_3, c7_4, c7_5, c8_3, c8_4, c8_5
    global c6_6, c6_7, c6_8, c7_6, c7_7, c7_8, c8_6, c8_7, c8_8
    for i in range(9):
        for j in range(9):
            cx_x = f'c{i}_{j}'
            number = number_list[i][j]
            if number > 0 and number < 10:
                exec(f'{cx_x}.delete("all")')
                  exec(f'{cx_x}.create_text(25,25, text = str({number}),
font = "Times 25 bold")')
```

```
            else:
                exec(f'{cx_x}.delete("all")')
def create_frames():
    #           Frames
        global  frame_1,  frame_2,  frame_3,  frame_4,  frame_5,  frame_6,
frame_7, frame_8, frame_9
    frame_1 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_2 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_3 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_4 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_5 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_6 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_7 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_8 = tk.Frame(win, bd = 3, relief = 'solid')
    frame_9 = tk.Frame(win, bd = 3, relief = 'solid')
create_frames()

def display_frames():
    #           Displaying Frames
        global  frame_1,  frame_2,  frame_3,  frame_4,  frame_5,  frame_6,
frame_7, frame_8, frame_9
    frame_1.grid(row = 0, column = 0)
    frame_2.grid(row = 0, column = 1)
    frame_3.grid(row = 0, column = 2)
    frame_4.grid(row = 1, column = 0)
    frame_5.grid(row = 1, column = 1)
    frame_6.grid(row = 1, column = 2)
    frame_7.grid(row = 2, column = 0)
    frame_8.grid(row = 2, column = 1)
    frame_9.grid(row = 2, column = 2)
display_frames()

def create_canvass():
    #       Frame 1
    global c0_0, c0_1, c0_2, c1_0, c1_1, c1_2, c2_0, c2_1, c2_2
    c0_0 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
    c0_1 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
    c0_2 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
    c1_0 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
    c1_1 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
    c1_2 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
    c2_0 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
    c2_1 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)
```

```
c2_2 = tk.Canvas(frame_1, bg = 'pink', width = 50, height = 50)

    #       Frame 2
global c0_3, c0_4, c0_5, c1_3, c1_4, c1_5, c2_3, c2_4, c2_5
c0_3 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c0_4 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c0_5 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c1_3 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c1_4 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c1_5 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c2_3 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c2_4 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)
c2_5 = tk.Canvas(frame_2, bg = 'pink', width = 50, height = 50)

    #       Frame 3
global c0_6, c0_7, c0_8, c1_6, c1_7, c1_8, c2_6, c2_7, c2_8
c0_6 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c0_7 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c0_8 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c1_6 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c1_7 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c1_8 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c2_6 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c2_7 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)
c2_8 = tk.Canvas(frame_3, bg = 'pink', width = 50, height = 50)

#       Frame 4
global c3_0, c3_1, c3_2, c4_0, c4_1, c4_2, c5_0, c5_1, c5_2
c3_0 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c3_1 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c3_2 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c4_0 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c4_1 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c4_2 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c5_0 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c5_1 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)
c5_2 = tk.Canvas(frame_4, bg = 'pink', width = 50, height = 50)

#       Frame 5
global c3_3, c3_4, c3_5, c4_3, c4_4, c4_5, c5_3, c5_4, c5_5
c3_3 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
c3_4 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
c3_5 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
```

```
c4_3 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
c4_4 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
c4_5 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
c5_3 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
c5_4 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)
c5_5 = tk.Canvas(frame_5, bg = 'pink', width = 50, height = 50)

#       Frame 6
global c3_6, c3_7, c3_8, c4_6, c4_7, c4_8, c5_6, c5_7, c5_8
c3_6 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c3_7 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c3_8 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c4_6 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c4_7 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c4_8 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c5_6 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c5_7 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)
c5_8 = tk.Canvas(frame_6, bg = 'pink', width = 50, height = 50)

#       Frame 7
global c6_0, c6_1, c6_2, c7_0, c7_1, c7_2, c8_0, c8_1, c8_2
c6_0 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c6_1 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c6_2 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c7_0 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c7_1 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c7_2 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c8_0 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c8_1 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)
c8_2 = tk.Canvas(frame_7, bg = 'pink', width = 50, height = 50)

#       Frame 8
global c6_3, c6_4, c6_5, c7_3, c7_4, c7_5, c8_3, c8_4, c8_5
c6_3 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c6_4 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c6_5 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c7_3 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c7_4 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c7_5 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c8_3 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c8_4 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
c8_5 = tk.Canvas(frame_8, bg = 'pink', width = 50, height = 50)
```

```
    #       Frame 9 canvas
    global c6_6, c6_7, c6_8, c7_6, c7_7, c7_8, c8_6, c8_7, c8_8
    c6_6 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c6_7 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c6_8 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c7_6 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c7_7 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c7_8 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c8_6 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c8_7 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
    c8_8 = tk.Canvas(frame_9, bg = 'pink', width = 50, height = 50)
create_canvass()

def display_canvass():
    #       Frame 1 Display
    global c0_0, c0_1, c0_2, c1_0, c1_1, c1_2, c2_0, c2_1, c2_2
    global c0_3, c0_4, c0_5, c1_3, c1_4, c1_5, c2_3, c2_4, c2_5
    global c0_6, c0_7, c0_8, c1_6, c1_7, c1_8, c2_6, c2_7, c2_8
    global c3_0, c3_1, c3_2, c4_0, c4_1, c4_2, c5_0, c5_1, c5_2
    global c3_3, c3_4, c3_5, c4_3, c4_4, c4_5, c5_3, c5_4, c5_5
    global c3_6, c3_7, c3_8, c4_6, c4_7, c4_8, c5_6, c5_7, c5_8
    global c6_0, c6_1, c6_2, c7_0, c7_1, c7_2, c8_0, c8_1, c8_2
    global c6_3, c6_4, c6_5, c7_3, c7_4, c7_5, c8_3, c8_4, c8_5
    global c6_6, c6_7, c6_8, c7_6, c7_7, c7_8, c8_6, c8_7, c8_8

    #       Frame 1 Display
    c0_0.grid(row = 0, column = 0)
    c0_1.grid(row = 0, column = 1)
    c0_2.grid(row = 0, column = 2)
    c1_0.grid(row = 1, column = 0)
    c1_1.grid(row = 1, column = 1)
    c1_2.grid(row = 1, column = 2)
    c2_0.grid(row = 2, column = 0)
    c2_1.grid(row = 2, column = 1)
    c2_2.grid(row = 2, column = 2)

    #       Frame 2 Display
    c0_3.grid(row = 0, column = 0)
    c0_4.grid(row = 0, column = 1)
    c0_5.grid(row = 0, column = 2)
    c1_3.grid(row = 1, column = 0)
    c1_4.grid(row = 1, column = 1)
    c1_5.grid(row = 1, column = 2)
```

```
c2_3.grid(row = 2, column = 0)
c2_4.grid(row = 2, column = 1)
c2_5.grid(row = 2, column = 2)

#       Frame 3 Display
c0_6.grid(row = 0, column = 0)
c0_7.grid(row = 0, column = 1)
c0_8.grid(row = 0, column = 2)
c1_6.grid(row = 1, column = 0)
c1_7.grid(row = 1, column = 1)
c1_8.grid(row = 1, column = 2)
c2_6.grid(row = 2, column = 0)
c2_7.grid(row = 2, column = 1)
c2_8.grid(row = 2, column = 2)

#       Frame 4 Display
c3_0.grid(row = 0, column = 0)
c3_1.grid(row = 0, column = 1)
c3_2.grid(row = 0, column = 2)
c4_0.grid(row = 1, column = 0)
c4_1.grid(row = 1, column = 1)
c4_2.grid(row = 1, column = 2)
c5_0.grid(row = 2, column = 0)
c5_1.grid(row = 2, column = 1)
c5_2.grid(row = 2, column = 2)

#       Frame 5 Display
c3_3.grid(row = 0, column = 0)
c3_4.grid(row = 0, column = 1)
c3_5.grid(row = 0, column = 2)
c4_3.grid(row = 1, column = 0)
c4_4.grid(row = 1, column = 1)
c4_5.grid(row = 1, column = 2)
c5_3.grid(row = 2, column = 0)
c5_4.grid(row = 2, column = 1)
c5_5.grid(row = 2, column = 2)

#       Frame 6 Display
c3_6.grid(row = 0, column = 0)
c3_7.grid(row = 0, column = 1)
c3_8.grid(row = 0, column = 2)
c4_6.grid(row = 1, column = 0)
c4_7.grid(row = 1, column = 1)
```

```
    c4_8.grid(row = 1, column = 2)
    c5_6.grid(row = 2, column = 0)
    c5_7.grid(row = 2, column = 1)
    c5_8.grid(row = 2, column = 2)

    #       Frame 7 Display
    c6_0.grid(row = 0, column = 0)
    c6_1.grid(row = 0, column = 1)
    c6_2.grid(row = 0, column = 2)
    c7_0.grid(row = 1, column = 0)
    c7_1.grid(row = 1, column = 1)
    c7_2.grid(row = 1, column = 2)
    c8_0.grid(row = 2, column = 0)
    c8_1.grid(row = 2, column = 1)
    c8_2.grid(row = 2, column = 2)

    #       Frame 8 Display
    c6_3.grid(row = 0, column = 0)
    c6_4.grid(row = 0, column = 1)
    c6_5.grid(row = 0, column = 2)
    c7_3.grid(row = 1, column = 0)
    c7_4.grid(row = 1, column = 1)
    c7_5.grid(row = 1, column = 2)
    c8_3.grid(row = 2, column = 0)
    c8_4.grid(row = 2, column = 1)
    c8_5.grid(row = 2, column = 2)

    #       Frame 9 Display
    c6_6.grid(row = 0, column = 0)
    c6_7.grid(row = 0, column = 1)
    c6_8.grid(row = 0, column = 2)
    c7_6.grid(row = 1, column = 0)
    c7_7.grid(row = 1, column = 1)
    c7_8.grid(row = 1, column = 2)
    c8_6.grid(row = 2, column = 0)
    c8_7.grid(row = 2, column = 1)
    c8_8.grid(row = 2, column = 2)
display_canvass()

f_number = [0, 0]

def change_focus(arg1):
    global f_number
```

```
    exec(f'{arg1}.focus_set()')
    f_number = [int(arg1[1]), int(arg1[3])]
def change_text(event):
    global f_number
    global number_list
    exec(f'c{f_number[0]}_{f_number[1]}.delete("all")')
    cx_x = f'c{f_number[0]}_{f_number[1]}'
    number_list[f_number[0]][f_number[1]] = int(event.char)
    exec(f'{cx_x}.create_text(25,25, text = event.char, font = "Times 25
bold")')

def focus_mouse():
    global c0_0, c0_1, c0_2, c1_0, c1_1, c1_2, c2_0, c2_1, c2_2
    global c0_3, c0_4, c0_5, c1_3, c1_4, c1_5, c2_3, c2_4, c2_5
    global c0_6, c0_7, c0_8, c1_6, c1_7, c1_8, c2_6, c2_7, c2_8
    global c3_0, c3_1, c3_2, c4_0, c4_1, c4_2, c5_0, c5_1, c5_2
    global c3_3, c3_4, c3_5, c4_3, c4_4, c4_5, c5_3, c5_4, c5_5
    global c3_6, c3_7, c3_8, c4_6, c4_7, c4_8, c5_6, c5_7, c5_8
    global c6_0, c6_1, c6_2, c7_0, c7_1, c7_2, c8_0, c8_1, c8_2
    global c6_3, c6_4, c6_5, c7_3, c7_4, c7_5, c8_3, c8_4, c8_5
    global c6_6, c6_7, c6_8, c7_6, c7_7, c7_8, c8_6, c8_7, c8_8

    #            FOCUSING THE MOUSE
    #       Row 1
    c0_0.bind('<Button-1>', lambda x: change_focus('c0_0'))
    c0_1.bind('<Button-1>', lambda x: change_focus('c0_1'))
    c0_2.bind('<Button-1>', lambda x: change_focus('c0_2'))
    c0_3.bind('<Button-1>', lambda x: change_focus('c0_3'))
    c0_4.bind('<Button-1>', lambda x: change_focus('c0_4'))
    c0_5.bind('<Button-1>', lambda x: change_focus('c0_5'))
    c0_6.bind('<Button-1>', lambda x: change_focus('c0_6'))
    c0_7.bind('<Button-1>', lambda x: change_focus('c0_7'))
    c0_8.bind('<Button-1>', lambda x: change_focus('c0_8'))

    #       Row 2
    c1_0.bind('<Button-1>', lambda x: change_focus('c1_0'))
    c1_1.bind('<Button-1>', lambda x: change_focus('c1_1'))
    c1_2.bind('<Button-1>', lambda x: change_focus('c1_2'))
    c1_3.bind('<Button-1>', lambda x: change_focus('c1_3'))
    c1_4.bind('<Button-1>', lambda x: change_focus('c1_4'))
    c1_5.bind('<Button-1>', lambda x: change_focus('c1_5'))
    c1_6.bind('<Button-1>', lambda x: change_focus('c1_6'))
    c1_7.bind('<Button-1>', lambda x: change_focus('c1_7'))
```

```
c1_8.bind('<Button-1>', lambda x: change_focus('c1_8'))

#       Row 3
c2_0.bind('<Button-1>', lambda x: change_focus('c2_0'))
c2_1.bind('<Button-1>', lambda x: change_focus('c2_1'))
c2_2.bind('<Button-1>', lambda x: change_focus('c2_2'))
c2_3.bind('<Button-1>', lambda x: change_focus('c2_3'))
c2_4.bind('<Button-1>', lambda x: change_focus('c2_4'))
c2_5.bind('<Button-1>', lambda x: change_focus('c2_5'))
c2_6.bind('<Button-1>', lambda x: change_focus('c2_6'))
c2_7.bind('<Button-1>', lambda x: change_focus('c2_7'))
c2_8.bind('<Button-1>', lambda x: change_focus('c2_8'))

#       Row 4
c3_0.bind('<Button-1>', lambda x: change_focus('c3_0'))
c3_1.bind('<Button-1>', lambda x: change_focus('c3_1'))
c3_2.bind('<Button-1>', lambda x: change_focus('c3_2'))
c3_3.bind('<Button-1>', lambda x: change_focus('c3_3'))
c3_4.bind('<Button-1>', lambda x: change_focus('c3_4'))
c3_5.bind('<Button-1>', lambda x: change_focus('c3_5'))
c3_6.bind('<Button-1>', lambda x: change_focus('c3_6'))
c3_7.bind('<Button-1>', lambda x: change_focus('c3_7'))
c3_8.bind('<Button-1>', lambda x: change_focus('c3_8'))

#       Row 5
c4_0.bind('<Button-1>', lambda x: change_focus('c4_0'))
c4_1.bind('<Button-1>', lambda x: change_focus('c4_1'))
c4_2.bind('<Button-1>', lambda x: change_focus('c4_2'))
c4_3.bind('<Button-1>', lambda x: change_focus('c4_3'))
c4_4.bind('<Button-1>', lambda x: change_focus('c4_4'))
c4_5.bind('<Button-1>', lambda x: change_focus('c4_5'))
c4_6.bind('<Button-1>', lambda x: change_focus('c4_6'))
c4_7.bind('<Button-1>', lambda x: change_focus('c4_7'))
c4_8.bind('<Button-1>', lambda x: change_focus('c4_8'))

#       Row 6
c5_0.bind('<Button-1>', lambda x: change_focus('c5_0'))
c5_1.bind('<Button-1>', lambda x: change_focus('c5_1'))
c5_2.bind('<Button-1>', lambda x: change_focus('c5_2'))
c5_3.bind('<Button-1>', lambda x: change_focus('c5_3'))
c5_4.bind('<Button-1>', lambda x: change_focus('c5_4'))
c5_5.bind('<Button-1>', lambda x: change_focus('c5_5'))
c5_6.bind('<Button-1>', lambda x: change_focus('c5_6'))
```

```
    c5_7.bind('<Button-1>', lambda x: change_focus('c5_7'))
    c5_8.bind('<Button-1>', lambda x: change_focus('c5_8'))

    #       Row 7
    c6_0.bind('<Button-1>', lambda x: change_focus('c6_0'))
    c6_1.bind('<Button-1>', lambda x: change_focus('c6_1'))
    c6_2.bind('<Button-1>', lambda x: change_focus('c6_2'))
    c6_3.bind('<Button-1>', lambda x: change_focus('c6_3'))
    c6_4.bind('<Button-1>', lambda x: change_focus('c6_4'))
    c6_5.bind('<Button-1>', lambda x: change_focus('c6_5'))
    c6_6.bind('<Button-1>', lambda x: change_focus('c6_6'))
    c6_7.bind('<Button-1>', lambda x: change_focus('c6_7'))
    c6_8.bind('<Button-1>', lambda x: change_focus('c6_8'))

    #       Row 8
    c7_0.bind('<Button-1>', lambda x: change_focus('c7_0'))
    c7_1.bind('<Button-1>', lambda x: change_focus('c7_1'))
    c7_2.bind('<Button-1>', lambda x: change_focus('c7_2'))
    c7_3.bind('<Button-1>', lambda x: change_focus('c7_3'))
    c7_4.bind('<Button-1>', lambda x: change_focus('c7_4'))
    c7_5.bind('<Button-1>', lambda x: change_focus('c7_5'))
    c7_6.bind('<Button-1>', lambda x: change_focus('c7_6'))
    c7_7.bind('<Button-1>', lambda x: change_focus('c7_7'))
    c7_8.bind('<Button-1>', lambda x: change_focus('c7_8'))

    #       Row 9
    c8_0.bind('<Button-1>', lambda x: change_focus('c8_0'))
    c8_1.bind('<Button-1>', lambda x: change_focus('c8_1'))
    c8_2.bind('<Button-1>', lambda x: change_focus('c8_2'))
    c8_3.bind('<Button-1>', lambda x: change_focus('c8_3'))
    c8_4.bind('<Button-1>', lambda x: change_focus('c8_4'))
    c8_5.bind('<Button-1>', lambda x: change_focus('c8_5'))
    c8_6.bind('<Button-1>', lambda x: change_focus('c8_6'))
    c8_7.bind('<Button-1>', lambda x: change_focus('c8_7'))
    c8_8.bind('<Button-1>', lambda x: change_focus('c8_8'))
focus_mouse()

def bind_text():
    global c0_0, c0_1, c0_2, c1_0, c1_1, c1_2, c2_0, c2_1, c2_2
    global c0_3, c0_4, c0_5, c1_3, c1_4, c1_5, c2_3, c2_4, c2_5
    global c0_6, c0_7, c0_8, c1_6, c1_7, c1_8, c2_6, c2_7, c2_8
    global c3_0, c3_1, c3_2, c4_0, c4_1, c4_2, c5_0, c5_1, c5_2
    global c3_3, c3_4, c3_5, c4_3, c4_4, c4_5, c5_3, c5_4, c5_5
```

```
global c3_6, c3_7, c3_8, c4_6, c4_7, c4_8, c5_6, c5_7, c5_8
global c6_0, c6_1, c6_2, c7_0, c7_1, c7_2, c8_0, c8_1, c8_2
global c6_3, c6_4, c6_5, c7_3, c7_4, c7_5, c8_3, c8_4, c8_5
global c6_6, c6_7, c6_8, c7_6, c7_7, c7_8, c8_6, c8_7, c8_8

#       Row 1
c0_0.bind('<Key>',change_text)
c0_1.bind('<Key>',change_text)
c0_2.bind('<Key>',change_text)
c0_3.bind('<Key>',change_text)
c0_4.bind('<Key>',change_text)
c0_5.bind('<Key>',change_text)
c0_6.bind('<Key>',change_text)
c0_7.bind('<Key>',change_text)
c0_8.bind('<Key>',change_text)

#       Row 2
c1_0.bind('<Key>',change_text)
c1_1.bind('<Key>',change_text)
c1_2.bind('<Key>',change_text)
c1_3.bind('<Key>',change_text)
c1_4.bind('<Key>',change_text)
c1_5.bind('<Key>',change_text)
c1_6.bind('<Key>',change_text)
c1_7.bind('<Key>',change_text)
c1_8.bind('<Key>',change_text)

#       Row 3
c2_0.bind('<Key>',change_text)
c2_1.bind('<Key>',change_text)
c2_2.bind('<Key>',change_text)
c2_3.bind('<Key>',change_text)
c2_4.bind('<Key>',change_text)
c2_5.bind('<Key>',change_text)
c2_6.bind('<Key>',change_text)
c2_7.bind('<Key>',change_text)
c2_8.bind('<Key>',change_text)

#       Row 4
c3_0.bind('<Key>',change_text)
c3_1.bind('<Key>',change_text)
c3_2.bind('<Key>',change_text)
c3_3.bind('<Key>',change_text)
```

```
c3_4.bind('<Key>',change_text)
c3_5.bind('<Key>',change_text)
c3_6.bind('<Key>',change_text)
c3_7.bind('<Key>',change_text)
c3_8.bind('<Key>',change_text)

#       Row 5
c4_0.bind('<Key>',change_text)
c4_1.bind('<Key>',change_text)
c4_2.bind('<Key>',change_text)
c4_3.bind('<Key>',change_text)
c4_4.bind('<Key>',change_text)
c4_5.bind('<Key>',change_text)
c4_6.bind('<Key>',change_text)
c4_7.bind('<Key>',change_text)
c4_8.bind('<Key>',change_text)

#       Row 6
c5_0.bind('<Key>',change_text)
c5_1.bind('<Key>',change_text)
c5_2.bind('<Key>',change_text)
c5_3.bind('<Key>',change_text)
c5_4.bind('<Key>',change_text)
c5_5.bind('<Key>',change_text)
c5_6.bind('<Key>',change_text)
c5_7.bind('<Key>',change_text)
c5_8.bind('<Key>',change_text)

#       Row 7
c6_0.bind('<Key>',change_text)
c6_1.bind('<Key>',change_text)
c6_2.bind('<Key>',change_text)
c6_3.bind('<Key>',change_text)
c6_4.bind('<Key>',change_text)
c6_5.bind('<Key>',change_text)
c6_6.bind('<Key>',change_text)
c6_7.bind('<Key>',change_text)
c6_8.bind('<Key>',change_text)

#       Row 8
c7_0.bind('<Key>',change_text)
c7_1.bind('<Key>',change_text)
c7_2.bind('<Key>',change_text)
```

```
    c7_3.bind('<Key>',change_text)
    c7_4.bind('<Key>',change_text)
    c7_5.bind('<Key>',change_text)
    c7_6.bind('<Key>',change_text)
    c7_7.bind('<Key>',change_text)
    c7_8.bind('<Key>',change_text)

    #       Row 9
    c8_0.bind('<Key>',change_text)
    c8_1.bind('<Key>',change_text)
    c8_2.bind('<Key>',change_text)
    c8_3.bind('<Key>',change_text)
    c8_4.bind('<Key>',change_text)
    c8_5.bind('<Key>',change_text)
    c8_6.bind('<Key>',change_text)
    c8_7.bind('<Key>',change_text)
    c8_8.bind('<Key>',change_text)
bind_text()

def clear_list_all():
    global number_list
    global c0_0, c0_1, c0_2, c1_0, c1_1, c1_2, c2_0, c2_1, c2_2
    global c0_3, c0_4, c0_5, c1_3, c1_4, c1_5, c2_3, c2_4, c2_5
    global c0_6, c0_7, c0_8, c1_6, c1_7, c1_8, c2_6, c2_7, c2_8
    global c3_0, c3_1, c3_2, c4_0, c4_1, c4_2, c5_0, c5_1, c5_2
    global c3_3, c3_4, c3_5, c4_3, c4_4, c4_5, c5_3, c5_4, c5_5
    global c3_6, c3_7, c3_8, c4_6, c4_7, c4_8, c5_6, c5_7, c5_8
    global c6_0, c6_1, c6_2, c7_0, c7_1, c7_2, c8_0, c8_1, c8_2
    global c6_3, c6_4, c6_5, c7_3, c7_4, c7_5, c8_3, c8_4, c8_5
    global c6_6, c6_7, c6_8, c7_6, c7_7, c7_8, c8_6, c8_7, c8_8
    for i in range(9):
        for j in range(9):
            cy_y = f'c{i}_{j}'
            exec(f'{cy_y}.delete("all")')

    number_list = [[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]]
def clear_list_sp():
    global number_list, f_number
    global c0_0, c0_1, c0_2, c1_0, c1_1, c1_2, c2_0, c2_1, c2_2
    global c0_3, c0_4, c0_5, c1_3, c1_4, c1_5, c2_3, c2_4, c2_5
    global c0_6, c0_7, c0_8, c1_6, c1_7, c1_8, c2_6, c2_7, c2_8
    global c3_0, c3_1, c3_2, c4_0, c4_1, c4_2, c5_0, c5_1, c5_2
    global c3_3, c3_4, c3_5, c4_3, c4_4, c4_5, c5_3, c5_4, c5_5
    global c3_6, c3_7, c3_8, c4_6, c4_7, c4_8, c5_6, c5_7, c5_8
    global c6_0, c6_1, c6_2, c7_0, c7_1, c7_2, c8_0, c8_1, c8_2
    global c6_3, c6_4, c6_5, c7_3, c7_4, c7_5, c8_3, c8_4, c8_5
    global c6_6, c6_7, c6_8, c7_6, c7_7, c7_8, c8_6, c8_7, c8_8
    cy_y = f'c{f_number[0]}_{f_number[1]}'
    exec(f'{cy_y}.delete("all")')
    number_list[f_number[0]][f_number[1]] = 0
def solve_sudoku():
    global number_list
    if SudokoSolver.check_correct(number_list) != 0:
        SudokoSolver.solve(number_list)
    else:
            tkinter.messagebox.showerror('ERROR!!!!',"You Cant Put same
number in one box and same row & col !!!")
    display_numbers()

button_solve = tk.Button(win, text = 'solve', command = solve_sudoku)
button_clear_all = tk.Button(win, text = 'clear board', command =
clear_list_all)
button_clear = tk.Button(win, text = 'delete', command = clear_list_sp)

button_clear_all.grid(row = 4, column = 1)
button_clear.grid(row = 4, column = 2)
button_solve.grid(row = 4, column = 0)

menu_file = tk.Menu(win)
edit_bar = tk.Menu(menu_file, tearoff = False)
edit_bar.add_command(label = 'clear board', command = clear_list_all)
edit_bar.add_command(label = 'solve', command = solve_sudoku)
menu_file.add_cascade(label = 'delete', menu = edit_bar)

display_numbers()
win.config(menu = menu_file)
win.mainloop()
```

SudokoSolver.py

```
list1 = [
        [0,0,6,8,4,0,0,0,0],
        [2,0,1,0,6,0,0,0,7],
        [0,3,9,0,0,0,0,1,0],
        [0,0,0,0,9,8,3,0,0],
        [0,6,0,0,0,0,0,9,0],
        [0,0,7,3,2,0,0,0,0],
        [0,4,0,0,0,0,1,3,0],
        [7,0,0,0,1,0,8,0,4],
        [0,0,0,0,3,5,7,0,0]
        ]

list2 = [
        [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]
        ]

def draw_list(lst):
        counter2 = 1
        counter1 = 1
        for element in lst:
        for a in element:
        if counter1 % 9 == 0:
                print(a)
                counter2 = 1
        else:
                if counter2%3 == 0:
                print(a,end = ' ')
                else:
                print(a,end='')
                counter2 += 1
        counter1 += 1
```

```
def solve(lst):
    for i in range(0,9):
    for j in range(0,9):
    if lst[i][j] == 0:
        psb = check_possible(lst,i,j)
        # print(f'{i+1}x{j+1} = {psb}')
        if len(psb) > 0:
        a = 0
        lst[i][j] = psb[a]
        # print(f'{i+1}x{j+1} = {psb[a]} chosen')
        while solve(lst) == 0:
        # print(f'{i+1}x{j+1} = {lst[i][j]} not a good choice')
        lst[i][j] = 0
        a = a+1
        try:
            lst[i][j] = psb[a]
            # print(f'{i+1}x{j+1} = {psb[a]} chosen')
        except IndexError:
            #print(f'{i+1}x{j+1} = {psb[a-1]} not a good one')
            # print(f'{i+1}x{j+1} = {lst[i][j]} not a good choice')
            lst[i][j] = 0
            return 0
        else:
        # print(f'{i+1}x{j+1} = {lst[i][j]} not a good choice ')
        lst[i][j] = 0
        return 0


def check_possible(lst,row,column):
    numbers = [1,2,3,4,5,6,7,8,9]
    for i in range(0,9):
    for j in range(0,9):
    if i == row:
        if lst[i][j] != 0:
        try:
        numbers.remove(lst[i][j])
        except ValueError:
        pass
    for i in range(0,9):
    for j in range(0,9):
    if j == column:
        if lst[i][j] != 0:
        try:
```

```
                numbers.remove(lst[i][j])
                except ValueError:
                pass
        i1 = row // 3
        j1 = column // 3
        for i in range(i1*3, i1*3+3):
        for j in range(j1*3,j1*3 + 3):
        if lst[i][j] != 0:
                try:
                numbers.remove(lst[i][j])
                except ValueError:
                pass
        return numbers



def check_correct(lst):
        for i in range(0,9):
        row_vector = []
        for j in range(0,9):
        if lst[i][j] != 0:
                row_vector.append(lst[i][j])
                if row_vector.count(lst[i][j]) > 1:
                return 0
        for j in range(0,9):
        column_vector = []
        for i in range(0,9):
        if lst[i][j] != 0:
                column_vector.append(lst[i][j])
                if column_vector.count(lst[i][j]) > 1:
                return 0

        for i in range(0,9):
        if i % 3 == 0:
        chunk_vector = []
        for j in range(0,3):
        if lst[i][j] != 0:
                chunk_vector.append(lst[i][j])
                if chunk_vector.count(lst[i][j]) > 1:
                return 0
        for i in range(0,9):
        if i % 3 == 0:
        chunk_vector = []
        for j in range(3,6):
```

```
if lst[i][j] != 0:
        chunk_vector.append(lst[i][j])
        if chunk_vector.count(lst[i][j]) > 1:
        return 0
for i in range(0,9):
if i % 3 == 0:
chunk_vector = []
for j in range(6,9):
if lst[i][j] != 0:
        chunk_vector.append(lst[i][j])
        if chunk_vector.count(lst[i][j]) > 1:
        return 0
```