**PJ 11 Report My Name: \_Yaowei Lei\_**

**A. The following is my Python source program:**

**// Please copy your source program into here from your Visual Studio IDE.**

**// Your code here must be in color. You must not show screen prints here.**

#--------------------------------------------------------------------------------------------------------.

# Author: Yaowei Lei

# Date: 12/12/2021

# Purpose: CS119-PJ11: Sudoku Game Check – to check a Sudoku game and show its result in detail.

#---------------------------------------------------------------------------------------------------------.

def showGame( S ) :

for i in range (9) :

for j in range (9) :

print (S[i][j], end=" ")

print ( )

def checkGame( S ) :

countBad = 0 # count how many problems being detected

for r in range(9): # 9 rows check with r = 0 to 8

if (not rowOK(S, r) ): # r = 0 to 8 from computer’s view

print("Row ", r + 1 , " has a problem.") # Row 1 to 9 from user’s view

countBad += 1 # increment countBad by 1

for c in range(9): # 9 columns check: 0 to 8, actually they mean column 1 to 9 for user.

if (not columnOK(S, c) ):

print("Column ", c + 1 , " has a problem.") # column 1 to 9 from user’s view

countBad += 1 # increment countBad by 1

for q in range(9): # 9 squares check: 0 to 8, actually they mean square 1 to 9 for user.

if (not squareOK(S, q) ):

print("Square ", q + 1 , " has a problem.") # Square 1 to 9 from user’s view

countBad += 1 # increment countBad by 1

if countBad == 0: # perfect game since nothing is bad

print("Congratulations! You won the game.")

def rowOK ( S, r ) :

goodlist = [1,2,3,4,5,6,7,8,9] # a perfect list of 1 thru 9 sorted order

slist = S[r] # get row r, which can be 0, 1, …, or 8

clist = [ ] # We must make a real copy of the original source list to avoid changing it here.

for element in slist:

clist.append(element) # make a real copy to avoid side effect to the original 9x9 array

clist.sort( ) # sort the list before comparing with goodlist

return (clist == goodlist) # true means OK for row r in S since they are equal

# end of RowOK( )

def columnOK ( S, c ) :

goodlist = [1,2,3,4,5,6,7,8,9] # a perfect list of 1 thru 9 sorted order

slist = [ S[0][c], S[1][c],S[2][c],S[3][c],S[4][c],S[5][c],S[6][c],S[7][c],S[8][c] ] # get column c, which can be 0, 1, …, or 8

clist = [ ] # We must make a real copy of the original source list to avoid changing it here.

for element in slist:

clist.append(element) # make a real copy to avoid side effect to the original 9x9 array

clist.sort( ) # sort the list before comparing with goodlist

return (clist == goodlist) # true means OK for cloumn c in S since they are equal

def squareOK ( S, q ) :

goodlist = [1,2,3,4,5,6,7,8,9] # a perfect list of 1 thru 9 sorted order

slist1 = [ S[0][0],S[0][1],S[0][2],S[1][0],S[1][1],S[1][2],S[2][0],S[2][1],S[2][2] ]

slist2 = [ S[0][3],S[0][4],S[0][5],S[1][3],S[1][4],S[1][5],S[2][3],S[2][4],S[2][5] ]

slist3 = [ S[0][6],S[0][7],S[0][8],S[1][6],S[1][7],S[1][8],S[2][6],S[2][7],S[2][8] ]

slist4 = [ S[3][0],S[3][1],S[3][2],S[4][0],S[4][1],S[4][2],S[5][0],S[5][1],S[5][2] ]

slist5 = [ S[3][3],S[3][4],S[3][5],S[4][3],S[4][4],S[4][5],S[5][3],S[5][4],S[5][5] ]

slist6 = [ S[3][6],S[3][7],S[3][8],S[4][6],S[4][7],S[4][8],S[5][6],S[5][7],S[5][8] ]

slist7 = [ S[6][0],S[6][1],S[6][2],S[7][0],S[7][1],S[7][2],S[8][0],S[8][1],S[8][2] ]

slist8 = [ S[6][3],S[6][4],S[6][5],S[7][3],S[7][4],S[7][5],S[8][3],S[8][4],S[8][5] ]

slist9 = [ S[6][6],S[6][7],S[6][8],S[7][6],S[7][7],S[7][8],S[8][6],S[8][7],S[8][8] ]

nlist = [slist1, slist2, slist3, slist4, slist5, slist6, slist7, slist8, slist9 ]

clist = [ ] # We must make a real copy of the original source list to avoid changing it here.

for element in nlist[q]:

clist.append(element) # make a real copy to avoid side effect to the original 9x9 array

clist.sort( ) # sort the list before comparing with goodlist

return (clist == goodlist) # true means OK for cloumn c in S since they are equal

# MAIN PROGRAM: ================================================.

# Nice to show row index 0 to 8 and column index 0 to 8 for readability of your code

S1 = [ # Game 1

# column index: 0 1 2 3 4 5 6 7 8

[1,2,3,4,5,6,7,8,9], # row index 0 || User’s row 1

[2,3,4,5,6,7,8,9,1], # 1 2

[3,4,5,6,7,8,9,1,2], # 2 3

[4,5,6,7,8,9,1,2,3], # 3 4

[5,6,7,8,9,1,2,3,4], # 4 5

[6,7,8,9,1,2,3,4,5], # 5 6

[7,8,9,1,2,3,4,5,6], # 6 7

[8,9,1,2,3,4,5,6,7], # 7 8

[9,1,2,3,4,5,6,7,8] ] # 8 9

S2 = [ # Game 2

# column index: 0 1 2 3 4 5 6 7 8

[1,2,3,4,5,6,7,8,9],

[4,5,6,7,8,9,1,2,3],

[7,8,9,1,2,3,4,5,6],

[2,3,4,5,6,7,8,9,1],

[5,6,7,8,9,1,2,3,4],

[8,9,1,2,3,4,5,6,7],

[3,4,5,6,7,8,9,1,2],

[6,7,8,9,1,2,3,4,5],

[9,1,2,3,4,5,6,7,8] ]

S3 = [ # Game 3

# column index: 0 1 2 3 4 5 6 7 8

[1,2,3,4,5,6,7,8,2],

[4,5,6,7,8,9,1,2,3],

[7,8,9,1,2,3,4,5,6],

[2,3,4,5,6,7,8,9,1],

[5,6,7,8,9,1,2,3,4],

[8,9,1,2,3,4,5,6,7],

[3,4,5,6,7,8,9,1,2],

[6,7,8,9,1,2,3,4,5],

[9,1,2,3,4,5,6,7,8] ]

S4 = [ # Game 4

# column index: 0 1 2 3 4 5 6 7 8

[1,2,3,4,5,6,7,8,9],

[4,5,6,7,8,9,1,2,3],

[7,8,9,1,2,3,4,5,6],

[2,3,4,5,6,7,8,9,1],

[5,6,7,8,9,1,2,3,4],

[8,9,1,2,3,4,5,6,7],

[3,4,5,6,7,8,9,1,2],

[6,7,8,9,1,8,3,4,5],

[9,1,2,3,4,5,6,7,8] ]

preGame = [S1, S2, S3, S4]

print ("Welcome to the Sudoku Game Check of \"Yaowei lei\"!")

n = 1 # line number for each separation line for readability

print(n,"=============================================================");n=n+1;

for g in range(4) :

print("Your game",g+1,"is as follows: " )

showGame(preGame[g]) # to print 9x9 game board

print( )

print("Your game",g+1,": ") # show the check result

checkGame(preGame[g]) # to check 9 rows/columns/squares. Total 27 checkings.

print( )

print(n,"=============================================================");n=n+1;

print("Thank you for playing the Sudoku Game Check of \"Yaowei lei\"!")

print (n,"============================================================"); n+=1;

# End of MAIN PROGRAM ============================================.

# End of Program ########################################

**B. The following is the console output of my 4 games:**

**// One way to copy the console output is to press Ctrl+Alt+PrtScn.**

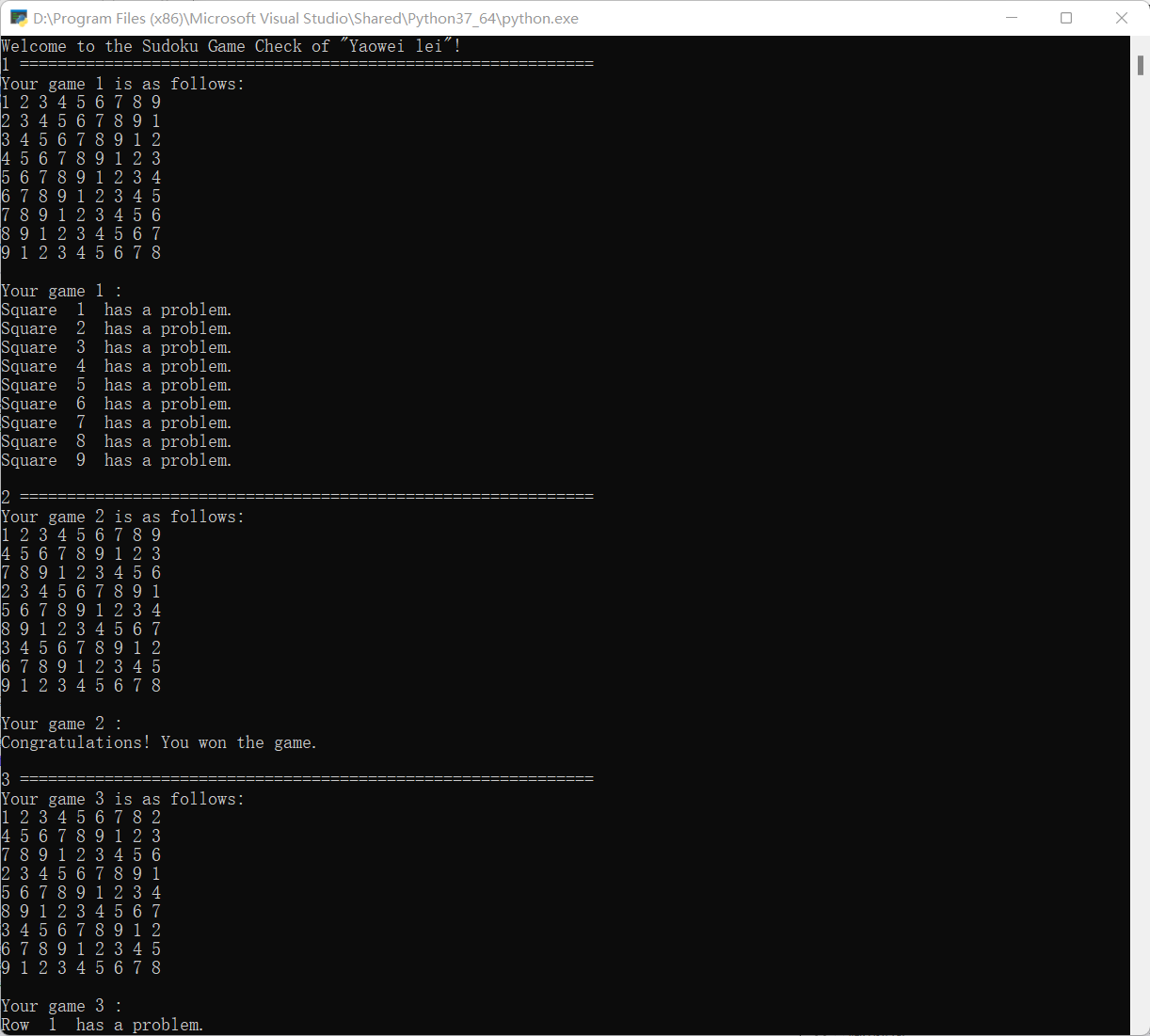
**// Another way to copy is to use the snipping tool. To paste the image is to press Ctrl+v.**

**// The console display must not be too wide, otherwise it will be too hard to read once pasted.**

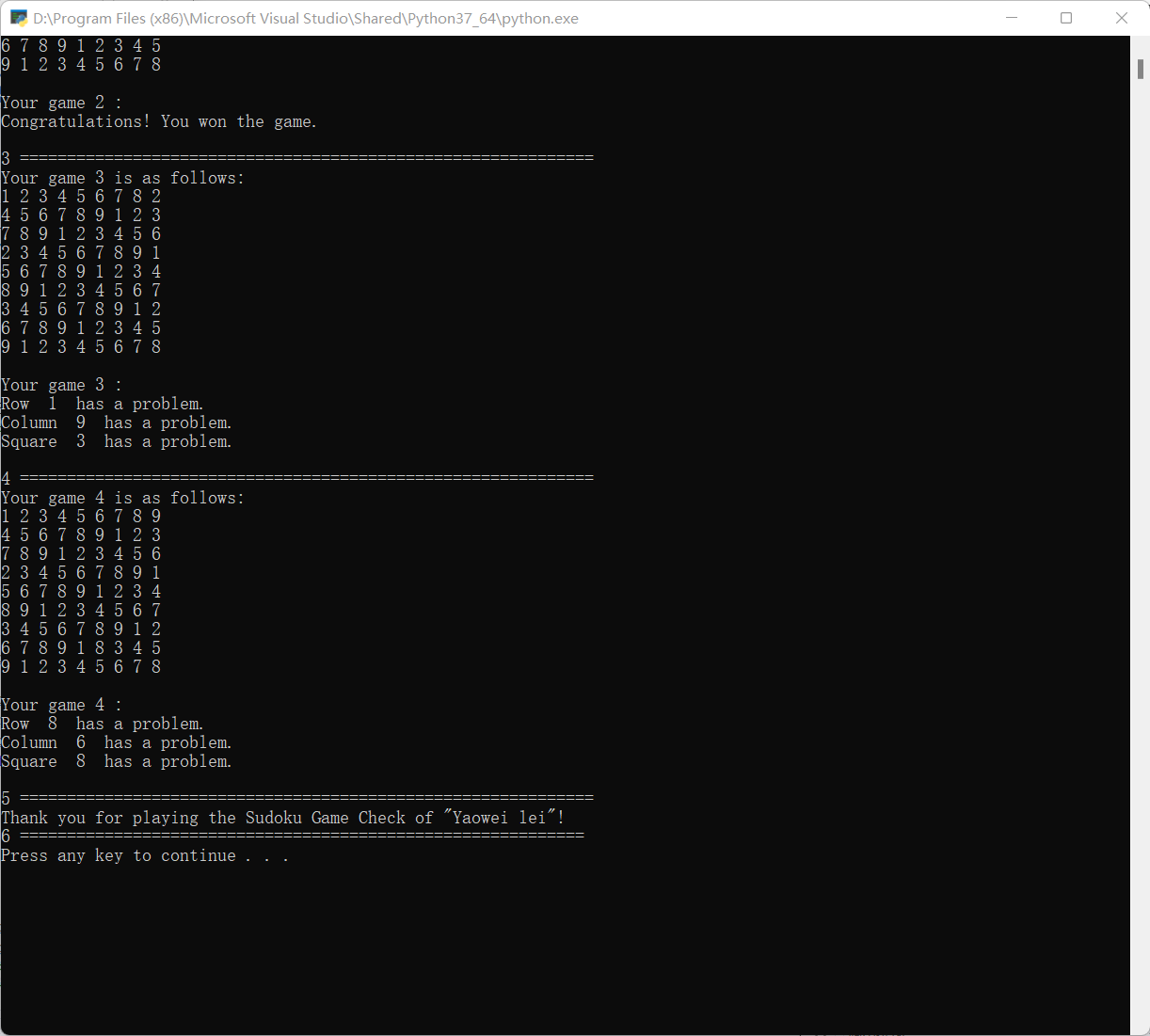
**// Please make sure your console is long enough to show all your output lines to be captured.**

**// Please copy your console output and paste into here:**

**Game 1 and Game 2:**

****

**Game 3 and Game 4:**

****