I created three files, first one is the main.c second one is the header file solvesudoku.h and finally a file where almost all my code is written is in solvesudoku.c file.

I call my header file with #include "solvesudoku.h" in main.c file as well as in solvesudoku.c file and that way I can use my functions in main.c (which are originally in solvesudoku.c)

In solvesudoku.c:

SET createPuzzle as integer array function

SET Arr\_puzzle as integer array

SET i and j as Integers

DEFINE Integer array function array as 9 by 9 two-dimensional array which stores numbers as follows:

0, 1, 9, 0, 0, 2, 0, 0, 0,

4, 7, 0, 6, 9, 0, 0, 0, 1,

0, 0, 0, 4, 0, 0, 0, 9, 0,

8, 9, 4, 5, 0, 7, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 2, 0, 1, 9, 5, 8,

0, 5, 0, 0, 0, 6, 0, 0, 0,

6, 0, 0, 0, 2, 8, 0, 7, 9,

0, 0, 0, 1, 0, 0, 8, 6, 0,

/\*DISCLAIMER: THIS SUDOKU TABLE IS COPIED FROM THE WEB FOR TESTING PURPOSES IT WILL WORK ON ANY OTHER SUDOKU TABLE. \*/

ALLOCATE memory for Arr\_puzzle

FOR (i = 0, i<9, i++)

ALLOCATE memory for every ith ekement of Arr\_puzzle

FOR (j= 0, j<9)

Copy numbers from array[] to Array\_puzzle[]

J++

ENDFOR

ENDFOR

SET printPuzzleas void function with int\*\* puzzle Parameter

SET I, J, A as integers

FOR (i = 0, a = 1; i < 9; INCREMENT i, INCREMENT a)

FOR (j = 0; j < 9; INCREMENT j)

PRINT out value of my puzzle with coordinate table

SET checkAvailable as Bool function with parameters int\*\* puzzle, int\* row, int\* column

SET I, J as integers

FOR (i = 0; i < 9; INCREMENT i,)

FOR (j = 0; j < 9; INCREMENT j)

IF (puzzle at I and j coordinates is equal to 0)

Value of address Row = i

Value of address Column = j

RETURN True

SET checkBoxas as Bool function with parameters int\*\* puzzle, int row, int column, int val

SET I, J as integers

SET squareRow, squareColumn as integers

FOR (i = 0; i < 9; INCREMENT i,)

Check the values at coordinantes (I, column) are equal to val

If they are equal RETURN False

FOR (j = 0; j < 9; INCREMENT j,)

Check the values at coordinantes (row , j) are equal to val

If they are equal RETURN False

squareRow = row - row % 3; (modulo of 3)

squareColumn = column - column % 3;

FOR (i = 0; i < 9; INCREMENT i,)

FOR (j = 0; j < 9; INCREMENT j)

Check the values at coordinantes (squareRow + I , squareColumn + j) are equal to val

If they are equal RETURN False

RETURN True otherwise

SET solvePuzzle as Bool function with parameter int\*\* puzzle

CALL checkAvailable to see if there are empty squares in the puzzle

CHECK values from 1 to 10 in each of the empty squares one by one by CALLING Function checkbox

IF it finds the value that is Valid

Insert that value to the coordinates found by checkAvailable Function

IF The puzzle can be solved with that input RETURN True

ELSE delete that value to the coordinates found by checkAvailable Function and Incrementit until it finds correct value

Return False // This is main line for recursion as if Checkbox is false it traces the “mistake” and tries out different values

SET copyPuzzle as Integer Array finction with parameter int\*\* puzzle

This basically copies everything from the puzzle to new Puzzle by using the FOR loop and allocating memory for newPuzzle

“FOR (i = 0; i < 9; INCREMENT i,)

FOR (j = 0; j < 9; INCREMENT j)”

SET userChoice as void Function with parameters int\*\* userPuzzle, int\*\* tempPuzzle

//This Function prompts player for imputs and checks if inputs are valid and correct

While(1) //infinity loop

IF (Checkavailable is false)

PRINT “Good Job You Solved the Puzzle!!!”

RETURN TRUE

WHILE(1)

Prompt user to either click ‘q’ to quit or press ‘Enter’ to continue

IF (user presses q)

Solve thepuzzle by CALLING solvePuzzle

PRINT the Solved Puzzle

RETURN

ELSE IF (c != '\n') AND (c != 'q')

Discard user input

ELSE

BREAK

PRINT “please Enter Coordinate for the square you want to insert the value to in the following format X Y:”

SCAN for user input

WHILE(1)

IF the coordinants picked is non-exsistant on the table ask user to insert coordinantes again

ELSE DECREMENT the users values by 1 and BREAK

PRINT “Please insert value from 1 to 9”

SCAN for user inputs

WHILE(1)

IF user value isn’t in the range of 1 to 9 ask again for value

ELSE BREAK;

CALL Function checkbox to see if user inserted correct value

IF (Checkbox returns true)

Insert the value in accordingly in coordinantes chosen by the user

ELSE

Prompt user to try again

BY using the For loop COPY Array userPuzzle to Array tempPuzzle

IF by calling solvepuzzle tempPuzzle can’t be solved

Prompt user to try again

SET userPuzzle at user inserted coordinantes to 0

PRINT userPuzzle by calling the printPuzzle function

RETURN

IN main.c we just call these Functions and assign them to variables

IN solvesudoku.h we CALL each Function we have in solvesudoku.c