LAB 14L

Submit code: **14L**

Problem

The purpose of this assignment is to give you practice with 2D Arrays.

Problem

We are going to start with a 2D integer array which can have a maximum of 12 rows and 12 columns and perform different operations on it.

1. There are 3 files that need to be copied from ~cs410c/public/14L to your local directory – array2D.c, array2D.h and array2D_functions.c The main function is in array2D.c

2. The main function calls each function to perform several operations. Each of the other functions must be coded in array2D_functions.c and their prototype declarations must be placed in the array2D.h file

3. First you fill your array with initial values from an input file. The first line of this file will contain the array dimensions. So for a 9x9 array, this line would be:
   9 9
   Subsequent lines will be the rows of the matrix (in the above case there will be exactly 9 lines of data that follow, each line containing 9 values). Look at the sample input files first to understand how the data files are structured.
   (Another input file might have 8 5 as the first line of data indicating that the data that follows is meant to be represented using an 8x5 array).

4. Use File I/O functions that you are already familiar with to read from the data file into your 2D array. For the purposes of this program you don't have to use EOF to find the end-of-file marker in the data file but simply use the values that you read on the first line to determine how many items you will read to fill up the entire array (one row after another, filling up each column in a row*). In other words, read the first two data values on that first line into two variables which can be used to set up the maximum number of iterations for your "nested for loop" that you will need to have similar to the example covered in class.*

   I have created the function prototype definition for read_array in array2D.h, created a function definition in array2D_functions.c and have the call from main() to this function. Notice how we define a 2D array in the function argument – you must specify at least the second subscript size of the 2D array (C expects at least this size).It needs to return the actual number of rows and columns as output arguments).

5. Next you print the contents of the whole array (just to standard output to the screen, not to an output file) to show a rectangular grid using a function called print_array() (input arguments are the 2D array, number of rows, number of columns). While your printing of the array doesn't have to be exactly like mine, make it as much of a grid format as possible for easy reading.

6. Next I want you to get the average value of all the cells in the array, i.e. sum up all the elements of the array, and divide by the total elements in the array. One way to do this is to navigate through the array one row at a time (for each row sum up all the column values before going on to the next row). Do this in the array_average() function (same number of input arguments as print_array but the function returns a floating point number).

7. Next I want you to replace all the diagonal elements of the grid by doubling its value. If the new value is greater than 100, replace it with 100. I'd like you to do this in the function

replace_diagonal().(again it's the same number of input arguments as print_array(), function returns nothing).

8. I'd like you to call the print_array function again for you to verify your changes from step 7 above

9. The main() function calls the appropriate functions in the following sequence: read data into the array, print the array, get the average value of all the cells in the array and print it, replace diagonal elements of the array only if the array happens to be a square array and print the array again.

10. Sample output is available for the first input data file

11. Submit all three files as 14L:
    submit 14L array2D.c array2D_functions.c array2D.h