TCSS 333 Spring 2015 HW2 Image Manipulation

Due: by midnight Friday Apr. 17 (or with 10% late penalty by midnight Saturday Apr. 18).

In this assignment, you will be working with image files of a particular kind: they will all be bitmap files (with extension .bmp), using 24-bit color and the width and height of the images will always be multiples of 8. Several examples of such files will be posted on Canvas. Your program should also be able to work with other images of the same type. Your goal is to read two image files and from them create two new image files:

-one that is the "double exposure" of the input images (called "blend.bmp")

- one that is an 8x8 checkerboard of the input images (called "checker.bmp")

Your program should start by reading in two files named "in1.bmp" and "in2.bmp". You can find out how .bmp files are formatted by examining them with a hexadecimal editor and comparing what you see to the description at http://en.wikipedia.org/wiki/BMP_file_format. Bitmap files are not all alike and this article provides more information than you need. The most important parts are the two tables labeled "Bitmap file header" and "Windows BITMAPINFOHEADER". The first one explains what is contained in the first 14 bytes of the file and the second one explains the next 40 bytes. If the image is w pixels in width by h pixels in height, then there are a total of w*h pixels. A pixel in a .bmp file is recorded using thee bytes standing for the red, green, and blue color components of the pixel. So an image file with image dimensions 100 by 200 will store 20,000 pixels. That same file will have a total size of:

```
14 bytes    Bitmap file header
40 bytes    Windows BITMAPINFOHEADER
60,000 bytes   width * height * (bytes per pixel) = 100 * 200 * 3
 ----------
 60,054  bytes total
```

In the headers, you will find

size of the BMP file in bytes
bitmap width in pixels
bitmap height in pixels
the image size

The wikipedia article and a hex editor will help you figure out exactly where they're located. You will need to read some of those numbers into int variables so you can use them in your program. The rest of the header information should not be changed, so you can simply read those parts into char arrays where those bytes will be stored until you later write them out to your output files.

Once you have finished reading through the headers (using a number of fread's), you will be ready to read the pixel data that makes up most of the input files, again using fread. You'll put all that pixel data in two large 2D arrays, one for each input image.

After that, you will probably want to create two other 2D arrays that will hold your new images (blend.bmp and checker.bmp). The double exposure image is created by calculating the average red, green, and blue values for each pixel, that is the average between image 1's pixel at row r, column c and image 2's pixel at row r, column c. We assume the two images have the same dimensions! As you calculate these averages, keep in mind that a char is a lot like a small number. By default, the type char can contain either a positive or negative number (-128 to 127). But R,G,B are values from 0 to 255. Therefore instead of using char to store your pixel data, use "unsigned char".

For the checker board, you will be combining the two images to create an 8 by 8 checkerboard. A normal checkerboard is made of alternating black and white squares. The checkboard you create will be made up of alternating squares taken from the matching position in the two original input images. To make this process simpler, we assume the width and height of the input images are both multiples of 8. As a way of figuring out how to assemble the checkerboard, you might consider this. If you number the rows and columns of a checkerboard from 0 to 7, and then label each square with the sum of its row and column, the pattern of even and odd sums is the same as black and white squares:

	0	1	2	3	4	5	6	7
0	0+0 E	0+1 O	0+2 E					
1	1+0 O	1+1 E	1+2 O					
2	2+0 E	2+1 O	2+2 E					
3								
4								
5								
6								
7								

After creating pixel data for your new images, you need to write out two .bmp files by reversing the same process you used to read in the starter file, this time using fwrite instead of fread.

Be ready to use a hexadecimal editor with your generated images, especially if Paint (or similar program) can't open the image. That's likely because the headers have been damaged.

Code quality guideslines from HW1 (indenting, no redundant code, etc) apply to this and all assigments. Refer back to HW1 if you have forgotten them.