**CS 4323 Design and Implementation of Operating Systems I**

**Assignment 03: Full Marks 100**

**(Due Date: 10/14/2019, 11:00 PM CST)**

The assignment must be done individually.

In this assignment, you will be using the concepts of multithreading using pthreads. Let us first understand the task and then we will see the implementation details.

How many of you have heard of watermarking or seen the application of watermarking. It is commonly used in the TV channels, where the channel logo is kept mostly in the left top of the screen. This type of watermarking is called visible watermarking, where the channel logo is very much visible to the user’s eye. There is invisible watermarking also, where the watermarked image is not visible and is conceived inside the image. Let us briefly understand the concept of watermarking (i.e. visible watermarking, to be specific).

Watermarking is the act of hiding a message related to the signal within the signal itself. The signal can be image, music video or any other video. It is similar to steganography, where both hides a message inside another signal. The difference between them is the message which is being hidden. In the case of steganography, the message to be hidden within a signal has no correlation with the signal, while watermarking tries to hide the message related to the signal within the signal. Watermarking is generally used to identify the ownership of the signal. Consider, if you have produced a masterpiece digital work and if it is copied by somebody, then how would you claim that the content belongs to you? If you have inserted watermark in your masterpiece work, then you can legally claim that the work belongs to you. So, watermarking provides a way to verify the right ownership of any content. It is also used to provide authenticity and integrity of the signal (interested students can explore more about this).

As said earlier, watermarking can be applied to any kind of signal (image, audio, video). However, for this assignment, we will consider the simplest application of watermarking in picture. Any watermarking scheme consists of three main parts:

* Watermark: The message which is to be hidden in the signal
* Encoder: the insertion algorithm used to hide the message in the signal
* Decoder: the extraction algorithm used to extract the watermark from the watermarked signal (i.e. the signal where hides the message)

This assignment will only focus on the encoding scheme. There are many complicated algorithms, but we will focus on the simplest one: Least Significant Bit (LSB) Coding.

Before explaining the encoding algorithm, brief explanation of the image. Below are the two images, we will be using for the assignment.



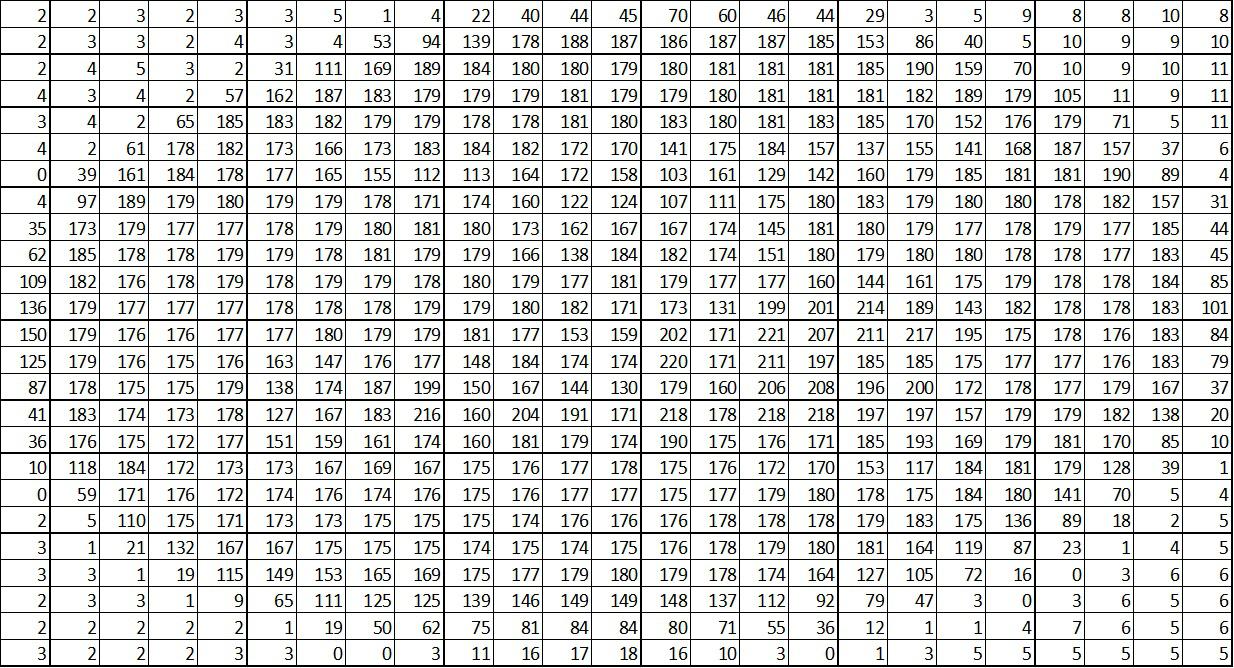
|  |  |
| --- | --- |
| **Fig. Signal (Original Image where watermark** | **Fig. watermark (OSU Logo)** |
| **is to be inserted)** |  |

**Figure 1. Images used for the assignment**

Both of these images have a resolution of 250-by-250 pixel. Each pixel contains image information i.e. the intensity value is given as numbers.

Consider smaller images and corelate the concept to the above image:

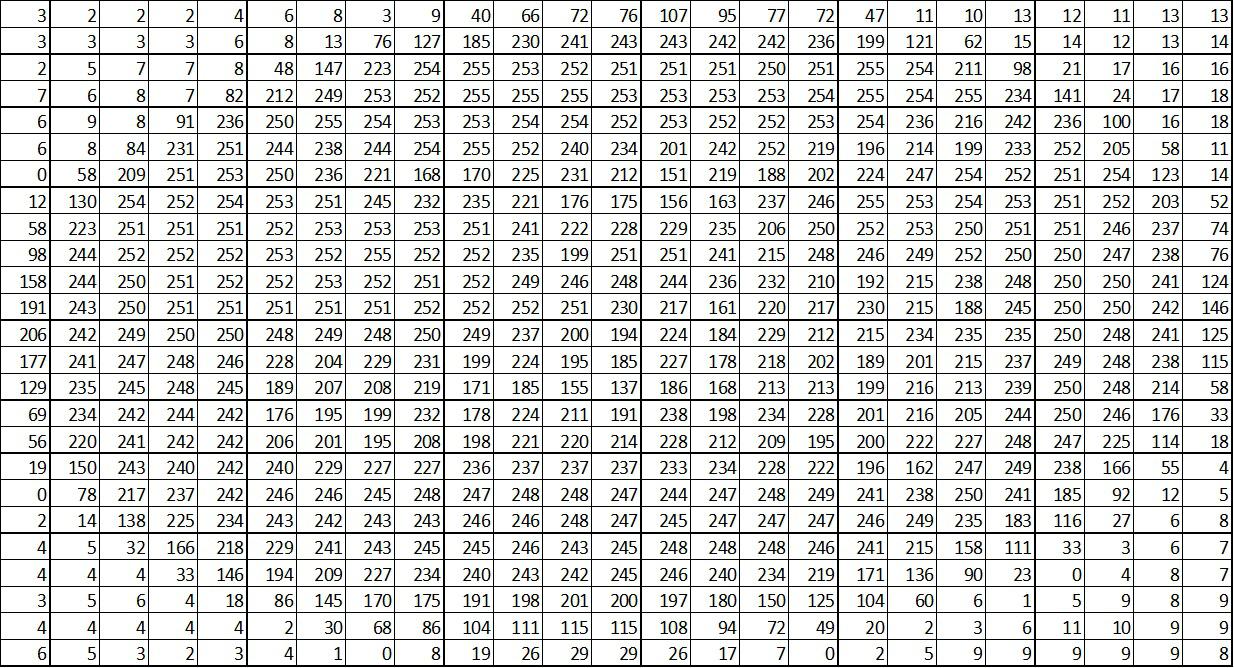
 This is a grayscale image meaning that each pixel contains 8 -bits data. Thus, each pixel contains a value in the range of 0 – 255. This image data is given by:



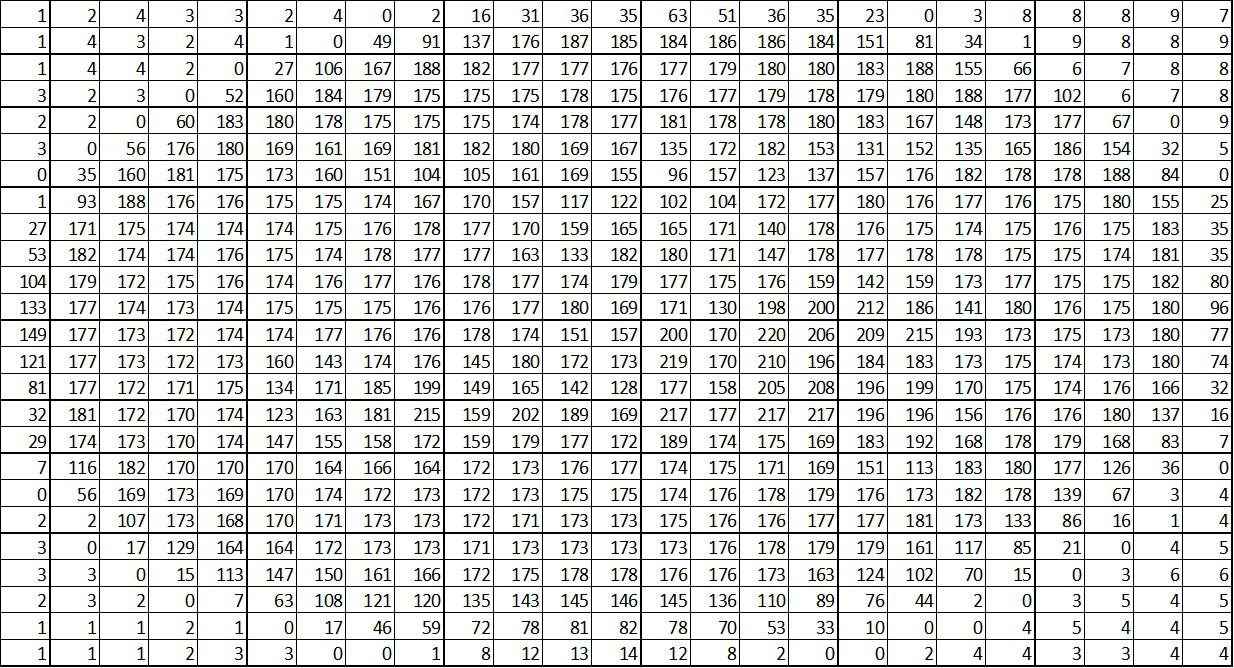
This is a 25-by-25 matrix i.e. number of row = number of column = 25. So, all grayscale image is a 2-D array.

This is true color image i.e. each pixel has red, green and blue component (in short RGB). Each color component (R, G, B) is represented by 8 -bits i.e. the range of possible value for each color component is between 0 and 255. This image has the following values:

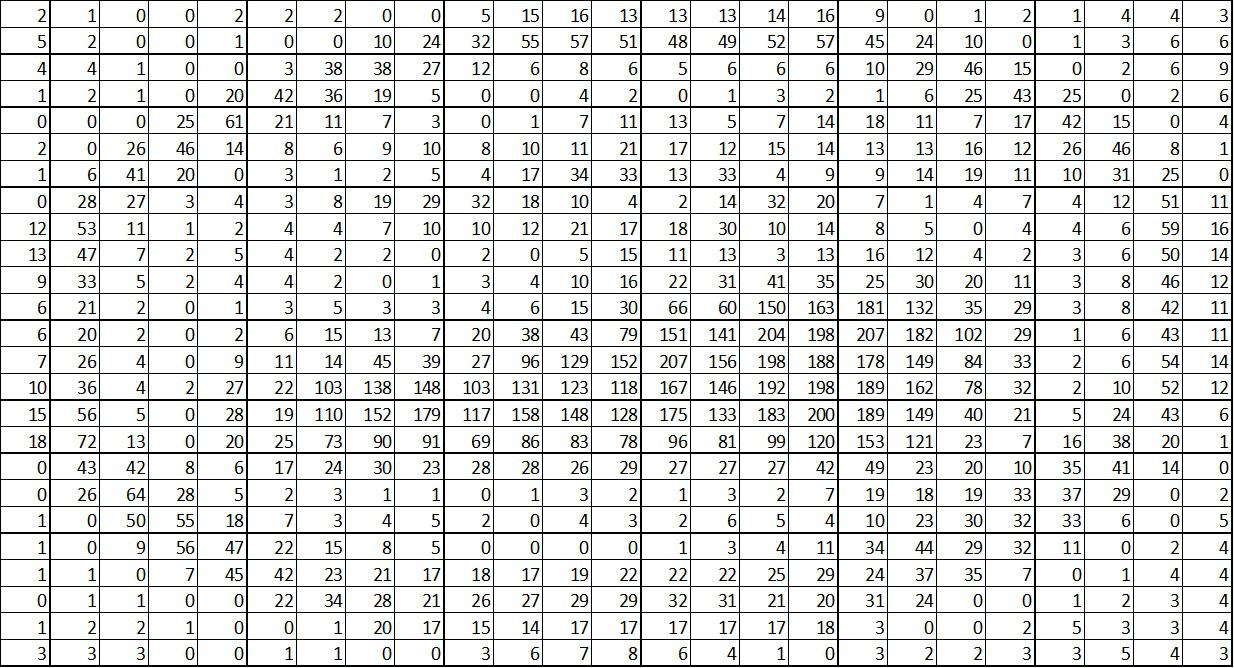
Red component values:



Green component values:



Blue component values:



This is a 25-by-25- by-3 matrix i.e. number of row = number of column = 25 and you have three color components (Red, Blue and Green. In short, RGB). So, all true color image is a 3-D array.

The left image in figure 1 is a true color image of size 250-by-250-by-3 while the right image in figure 1 is a gray color image of size 250-by-250. You are given 4 txt files along with this assignment:

* redComponent.txt
* greenComponent.txt
* blueComponent.txt
* OSU.txt ( gray color image)

The first three .txt files represent the three color components (R, G and B) for the left image (scenery image). The last .txt file represent the gray color component for the right image (OSU logo).

All together you have to use 4 threads: three threads (one thread per each color component) to process the true color image and one thread to perform task on the grayscale image. The steps to be followed by each thread are as follows:

1. Each thread should read the data from the corresponding color component .txt file. o First three threads perform read for the truecolor image

o The last thread (i.e. 4th thread) performs read for the gray color image

1. The first three threads now removes the least significant bits (LSB) from each cell in their matrix i.e. if the LSB is 0 then it remains 0 but if the LSB is 1 then it is converted to 0.

For e.g.: consider the red component of the true color image as:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 129 | 46 | 180 | 127 | 53 |
| 12 | 35 | 245 | 255 | 24 |
| 9 | 0 | 212 | 123 | 63 |
| 23 | 45 | 190 | 35 | 145 |
| 1 | 46 | 32 | 26 | 150 |

This matrix gets changed into:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 128 | 46 | 180 | 126 | 52 |
| 12 | 34 | 244 | 254 | 24 |
| 8 | 0 | 212 | 122 | 62 |
| 22 | 44 | 190 | 34 | 144 |
| 1 | 46 | 32 | 26 | 150 |

All the 3 threads perform this operation for their respective matrix.

1. The 4th thread needs to convert the corresponding 2-D matrix (i.e. gray color image data) into binary form i.e. the values in this matrix should be either zero or one (this is black & white image). The way to do this is:

o If any cell value is between 0 and 127 (inclusive), then make it to 0.

o Remaining values i.e. in the range between 128 and 255 (inclusive), it is converted into 1.

For e.g.: Given the matrix (i.e. gray color image data):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 129 | 46 | 180 | 127 | 53 |
| 12 | 35 | 245 | 255 | 24 |
| 9 | 0 | 212 | 123 | 63 |
| 23 | 45 | 190 | 35 | 145 |
| 1 | 46 | 32 | 26 | 150 |

The corresponding black-and-white image data is:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 |

The 4th thread’s task end here.

1. Each 3 threads (i.e. first 3 threads) need to add the corresponding matrices obtained from step 2 with the matrix obtained from step 3. It needs to be ensured that the minimum

value can be only 0 and maximum value in any cell can be 255. Any value lesser than 0 has to be converted to 0 and any value larger than 255 has to be converted into 255.

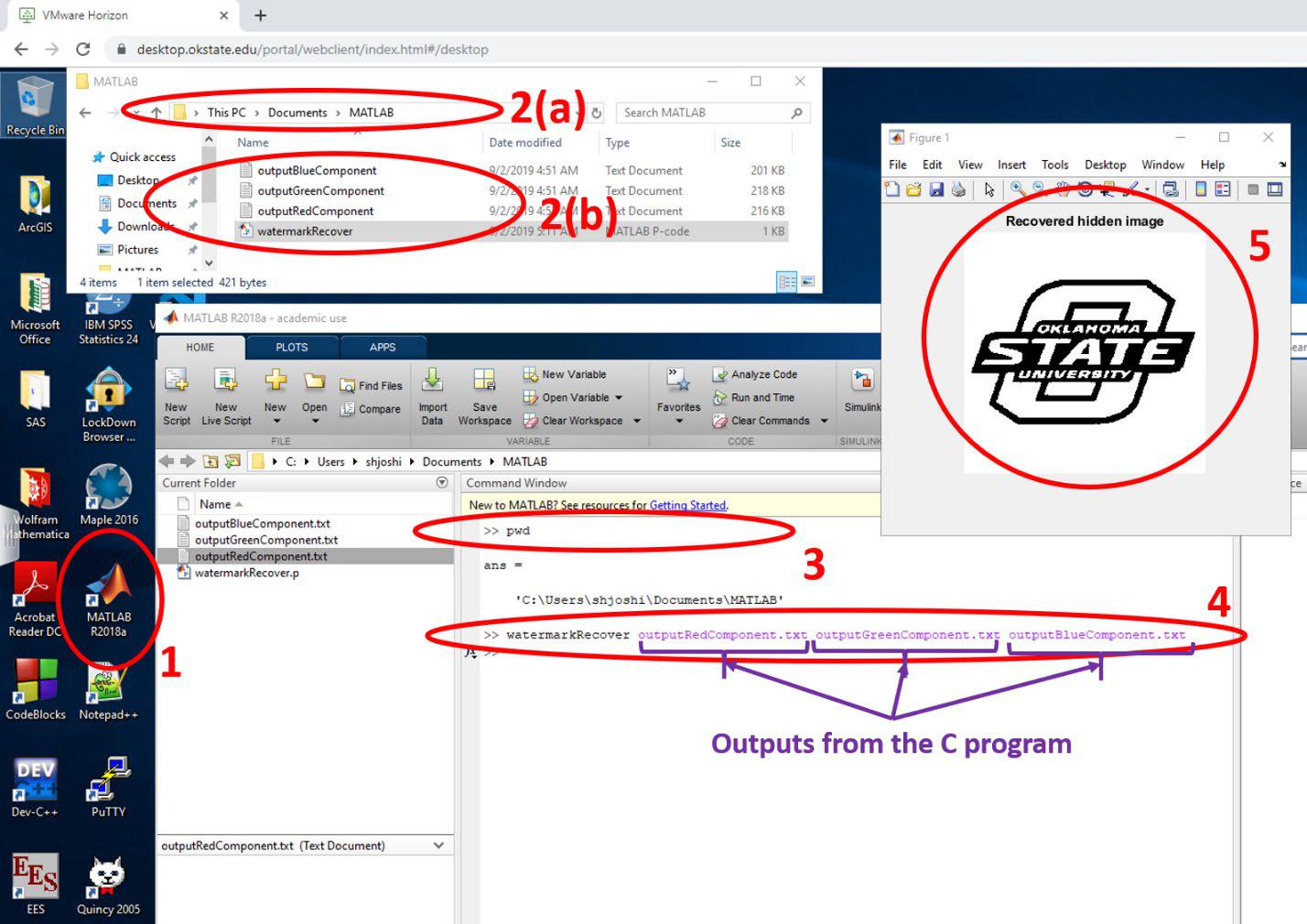
1. Each thread needs to save the final matrix in the .txt file as:
   * outputRedComponent.txt for red component
   * outputGreenComponent.txt for green component
   * outputBlueComponent.txt for blue component

This is it. You have successfully inserted the watermark in the image.

The following in not the part of the assignment. If anyone is interested to check if their final outputs are correct or not then they can check by following the steps given below:

* Visit the okstate virtual lab at: <https://desktop.okstate.edu/portal/webclient/index.html#/>
* Run the MATLAB
* Along with this assignment, you have been given watermarkRecover.p file. Copy this .p file along with your outputs from your c file into Documents > MATLAB folder.
* In the command window of MATLAB, type:

watermarkRecover outputRedComponent.txt outputGreenComponent.txt outputBlueComponent.txt



If you see output (as shown above, by number 5), then your program is correct.

**Submission Guidelines:**

* You should submit your programming assignment as a single .c file: Assignment03\_LastName\_FirstName\_01.c (where \*\* means assignment number and XX means question number). Example: Assignment03\_Andrew\_Simon\_01.c
* Your code should include the header information, which should include:

1. Name o CWID o Email
2. Date

followed by a brief description of the program.