5CS021 – Numerical Methods and Concurrency **LAB REPORT – Week7**

Attempt all questions

1. Apply a negative filter on an image file. This is the process of reversing the RGB values. For example, if the Red value of the pixel is 100, the inverse is 155. NOTE: 0 is minimum and 255 is maximum.
2. Write a program that reads an image file into a 1D array and converts it into a grayscale image. A color of gray is one in which Red=Green=Blue. Large values are white and small values are black. An easy way to make an grayscale image out of color is to set each color value to the average of all three: gray=(R+G+B)/3; R=gray; G=gray; B=gray;
3. Write a program that reads an image file and stores its data in a 1D array. Then, ask the user how many pixels to change to black from the right and bottom edges of the image. For example, if the original image size is 10 x 10, and the user specifies changing 2 pixels from the right and 3 pixels from the bottom to black, the new image will have black pixels in those areas while keeping the rest of the image unchanged. Finally, save the modified image as a new file.
4. Write a program that reads an image file and stores its data in a 1D array. Then ask the user to enter a number from -255 to 255 (to either darken or brighten the image). Then add or subtract that number from each pixel value (except for transparency) and make sure the pixel value stays within the range 0 to 255, so a certain validation check must be done.

Optional Question (No need to do handwritten):

1. Write a program that reads an image file and stores its data in a 1D array. Then, ask the user how many pixels to crop from the right and the bottom edges of the image. For example, if the original image size is 10 x 10, and the user specifies cropping 2 pixels from the right and 3 pixels from the bottom, the new image size should be 8 x 7. Finally, save the cropped image as a new file.

2. Write a program that reads an image file and stores its data in a 1D array. Then, perform a 90-degree clockwise rotation of the image by rearranging the pixels in the array. To do this, calculate the new position of each pixel in the rotated image, where the x-coordinate becomes the y-coordinate and the y-coordinate is adjusted based on the width of the image. Allocate memory for the rotated image with swapped width and height, copy the pixel data to the new positions, and save the rotated image as a new file.

   For example for a 3x3 image: (0,0) (0,1) (0,2)
   (1,0) (1,1) (1,2)
   (2,0) (2,1) (2,2)

   Should become: (2,0) (1,0) (0,0)
   (2,1) (1,1) (0,1)
   (2,2) (1,2) (0,2)