CS 240 – Advanced Programming Languages

Assignment 1 – Imperative Programming in C – 100 points

The primary goal of this assignment is to help you familiarize with **Imperative Programming.** You will learn to:

- use simple structures.
- Dynamically allocate memory.
- Write functions.
- Implement pointes and references.
- Read from and write to files.

You can write your program either on EDORAS server or CLION IDE. CLION IDE is recommended.

In this program, you will read and manipulate grayscale images. Download the given starter_code.c and implement the missing functions.

Program outline:

1. Your C program should read any given grayscale images in pgm format. Grayscale images have pixel values ranging from 0-255. This pixel values can be treated as unsigned characters.
2. The image data should be loaded as a 2D matrix.
3. The 2D matrix should be manipulated to threshold the given image and convert it to a binary image.
4. The 2D matrix should be manipulated to rotate the given image left.
5. The manipulated 2D matrices should be converted back to grayscale image. This image should be viewable.

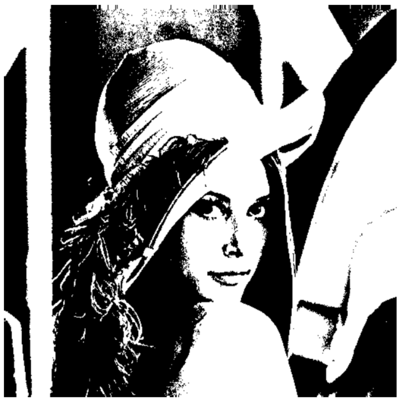
Your tasks: (READ CAREFULLY)

1. Do not alter the main() function. Four functions are declared and called in the given program. Your task is to write the function definitions for the provided functions.
2. PixelGray** readPGM(const char* filename, int* width, int* height);
    a. This function should open the file with provided file name.
    b. Handle errors!
    c. Get the dimensions of the image.
    d. Dynamically allocate the storage (2D matrix) for the image dimension.
    e. Read from the file and fill the 2D matrix.
    f. Close the file.
    g. Return the 2D matrix.
3. void writePGM(const char* filename, PixelGray** matrix, int* width, int* height);
    a. This function should create a new file with the file name provided. This file should be opened for writing.
    b. Handle errors!
    c. Copy data from the provided 2D matrix to the file.
    d. Close the file.
4. PixelGray** threshold(PixelGray** matrix, int* width, int* height);
    a. This function should create a new 2D matrix. Dynamically allocate the storage (2D matrix) for the new matrix.
    b. Thresholding
        i. If ***original value > 80*** -- store 255
        ii. Otherwise – store 0
    c. Return the new 2D matrix.
5. PixelGray** threshold(PixelGray** matrix, int* width, int* height);
    a. This function should create a new 2D matrix. Dynamically allocate the storage (2D matrix) for the new matrix.
    b. Rotate the original 2D matrix and store the result as the new matrix.
        i. Rotation can be achieved by **matrix transpose** (swapping rows & column).
    c. Return the new 2D matrix.

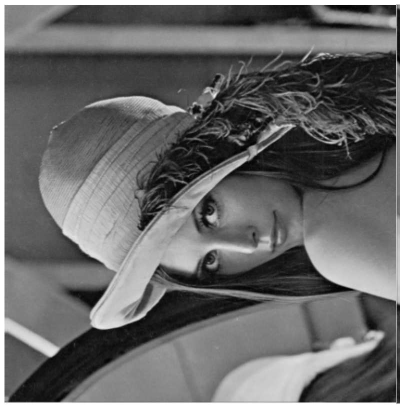
Input: lenna.pgm

Sample input

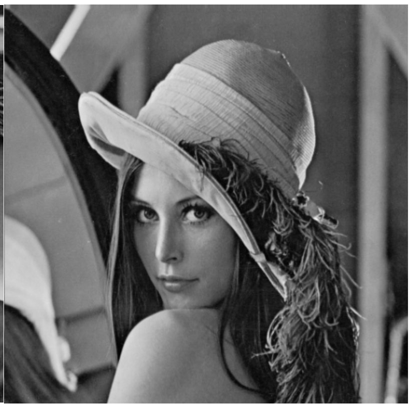
Output:



threshold.pgm



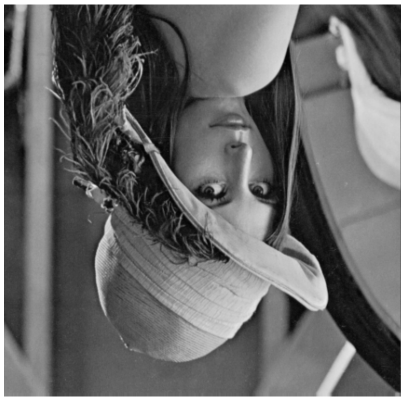
rotate.pgm



rotate_again.pgm

Submissions:

1. Completed code (.c) file should be turned in on Gradescope. Your code should not have any dependencies. No external library should be used. OPENCV functions cannot be used.
2. A simple 2-page report (pdf) should be submitted to canvas (Don't spend more than 20 minutes)
   a. Indicate progress. Did you complete the assignment? What obstacles did you face.
   b. Snapshot of output images for given lenna.pgm file and any grayscale image of your choice. Feel free to alter the threshold value to get good binary image.
   c. Why did rotating twice give you the original image? (as shown in previous page) Why not an up-side down image (as shown below). Explain your answer in 3-4 sentences.



   d. Did you collaborate with anyone? Give credit to your classmates/TAs/Online references who helped with logic/ideas. You cannot show your code to others and/or obtain code from others (including online sources).

Your codes will be tested for plagiarism.

1. Grade scope will compare your codes with all submissions across sections, online resources and AI generated codes.
2. TAs and the instructor reserve the right to inquire any student to explain the code submitted.