Hack 6.0

Computer Science I

Department of Computer Science & Engineering
University of Nebraska–Lincoln

Introduction

Hack session activities are small weekly programming assignments intended to get you started on full programming assignments. Collaboration is allowed and, in fact, *highly encouraged*. You may start on the activity before your hack session, but during the hack session you must either be actively working on this activity or *helping others* work on the activity. You are graded using the same rubric as assignments so documentation, style, design and correctness are all important.

Problem Statement

In this hack you'll get some more practice writing functions that utilize pass-by-reference (pointers), error handling and enumerated types. There are several different ways to model colors including RGB and CMYK. RGB is generally used in displays and models a color with three values in the range $[0, 255]$ corresponding to the red, green and blue "contribution" to the color. For example, the triple $(255, 255, 0)$ corresponds to a full red and green (additive) value which results in yellow. CMYK or Cyan-Magenta-Yellow-Black is a model used in printing where four colors of ink are combined to make various colors. In this system, the four values are on the scale $[0, 1]$. Write functions to convert between these models.

1. Write a function to convert from an RGB color model to CMYK. To convert to CMYK, you first need to scale each integer value to the range $[0, 1]$ by simply computing

$$r' = \frac{r}{255}, \quad g' = \frac{g}{255}, \quad b' = \frac{b}{255}$$

and then using the following formulas:

$$k = 1 - \max\{r', g', b'\}$$
$$c = \frac{(1 - r' - k)}{(1 - k)}$$
$$m = \frac{(1 - g' - k)}{(1 - k)}$$
$$y = \frac{(1 - b' - k)}{(1 - k)}$$

Your function should have the following signature:

```
int rgbToCMYK(int r, int g, int b, double *c, double *m, double *y, double *k)
```

Identify any and all error conditions and use the return value to indicate an error code (0 for no error, non-zero value(s) for error conditions). Note that one edge case is black, when $(r, g, b) = (0, 0, 0)$ which would lead to a division by zero in the formulas. The equivalent CMYK values are $(0, 0, 0, 1)$.

2. Write a function to convert from CMYK to RGB using the following formulas.

$$r = 255 \cdot (1 - c) \cdot (1 - k)$$
$$g = 255 \cdot (1 - m) \cdot (1 - k)$$
$$b = 255 \cdot (1 - y) \cdot (1 - k)$$

Results should be rounded. Your function should have the following signature:

```
int cmykToRGB(double c, double m, double y, double k, int *r, int *g, int *b)
```

Identify any and all error conditions and use the return value to indicate an error code (0 for no error, non-zero value(s) for error conditions).

Instructions

- You are encouraged to collaborate any number of students before, during, and after your scheduled hack session.
- Design at least 3 test cases for each function *before* you begin designing or implementing your program. Test cases are input-output pairs that are known to be correct using means other than your program.
- You may (in fact are encouraged) to define any additional "helper" functions that you find useful.
- Include the name(s) of everyone who worked together on this activity in your source file's header.

- Place your prototypes and documentation in a header file named `colorUtils.h` and your source in a file named `colorUtils.c`.
- A testing file, `utilsTester.c` has been provided that uses cmocka (https://cmocka.org/), a unit testing framework for C. We have already written several (17) test cases for you. Using these examples, implement your test cases using cmocka for your two functions. You should have at least 3 test cases for each function for a total of 23 test cases.

  The starter file should be sufficient to demonstrate how to use cmocka, but the full documentation can be found here: https://api.cmocka.org/. A `makefile` has also been provided to help you easily compile your files. Note that cmocka is already installed on the CSE server. If you compile on your own machine, you will have to install and troubleshoot cmocka yourself.
- Turn in all of your files via webhandin, making sure that it runs and executes correctly in the webgrader. Each individual student will need to hand in their own copy and will receive their own individual grade.