Assignment 1 – Images, Classes, Vectors and Singletons – Putting it all together in C++.

CLEMSON School of COMPUTING

Introduction

In this assignment you will demonstrate the following skills:

1. Use of classes and classes containing objects as members.
2. Use of a container class (vector) to keep track of data.
3. Use of file I/O streams.
4. Use of a singleton class design pattern.
5. Use of makefiles

Part I – Problem at hand.

This project is done in C++ in *teams of 2, or individually*. Teams of more than 2 students are not allowed. Your teammate should be from the same lecture section of the class. You will not be assigned into groups, so if you would like to work with a partner, you will have to find one on your own, and you will share the same grade, so please *choose wisely*. You are also not allowed to discuss your solution with other teams/students, except your own partner (if you have one). Only one person of the team should submit the files, and all files (except makefile) should have author/s names in comments on the very top of the file.

You will work with .ppm and .pgm files. You will read data from a provided .ppm (as well as .pgm) file and write that data to another .ppm (and .pgm respectively) file, which will effectively create a copy of the original. To verify that you have done it correctly, both the copy and the original can be open in an image editing software (like GIMP) and should look identical. GIMP is a free open source software that can be downloaded free and is available for all platforms (PC, Mac, and Linux). It is very powerful, supports many file extensions, and can convert between different image formats. We will be using GIMP while grading.

Though some implementation requirements are provided to you, you will have to flesh out the rest of the details of your design on your own. Below is the suggested course of action.

1. Do online research and learn about .ppm and .pgm files and their structure. Please keep in mind that pixel values in files with magic numbers P2 and P3 are in ASCII format, and in files with P5 and P6 in byte format (also called raw, or binary). To print to a file in binary format you need to research the ostream options, specifically ios:: binary.

2. Work in the modular fashion – design and test one class/function at a time, then add more code and test, etc. Do not write pages of code and then try to debug, as this may be very difficult and frustrating.

3. It is recommended that you implement .ppm file processing first, and then add .pgm processing, as they will be very similar.

4. Feel free to look up examples online, but *all code you write should be your own, written by you and your teammate.* You cannot use code from any other source, even if you cite it. Plagiarism will be addressed according to Clemson University academic integrity policy. Please do your own work.

Part II – Implementation Requirements

1. You will implement a class ColorPixel (*colorPixel.h*) that will contain the red, green, and blue values of each color pixel and class GrayPixel (*grayPixel.h*) to contain values of the gray pixel. You will need to provide parameterized constructors, destructors, and getter methods for each class. If you have a parameterized constructor, you will not need setter methods to save the pixel values, as the only time you will need to set values of the pixel is when you are creating an instance of that pixel. This will be a very small class with most, or all functions inlined.

2. You will create a class Ppm and a class Pgm. Each class will have a header file (named *ppm.h* and *pgm.h*) and an implementation file (*ppm.cpp* and *pgm.cpp*). You will need member variables for storing magic number, height, width, max_value for each pixel, (other members, if needed), and a vector of ColorPixel or GrayPixel objects, correspondingly. Please remember that you have to handle both ASCII and binary formats of .ppm and .pgm files, so you will need to create the necessary methods for that. It is up to you to figure out the best way to do it.

3. Your driver will be a singleton class. Singleton class is a design pattern that allows only one instance of that class to be instantiated. (please research singleton classes online and carefully study the code examples to understand how they work). This class will be called Replicator (and will live in r*eplicator.cpp*), since it replicates image files. It will contain a static public pointer to an instance of class Replicator, and will have private constructor/s. You will use a static method Instance() to create an instance of a Replicator class, and you will access all the class members/methods via that static public pointer.

4. You will need to create a makefile to build your program. Make sure to provide the functionality to remove object files (*clean*).

5. You will create a tarball containing all the source code and the makefile, but *not* the object code. Please run *clean* before creating a tarball. You do not need to submit image files. Graders will use their own image files to test your program.

6. Submit to canvas before the deadline.

NOTE: If you submit the archive that cannot be open, or is corrupt, you will not be given the opportunity to redo the work and resubmit. You have one shot to get it right.

How to create your own .ppm and .pgm files in GIMP (so you can use them to test your program)

Two *.ppm* and two *.pgm* files (ASCII and raw format) are provided for you. They can be found in Canvas under the assignment1 link: chips.ppm, chips.pgm, potatochips.ppm, potatochips.pgm. BUT if you would like to test your program with some more interesting/exciting files, you can convert your favorite image to *.ppm* or *.pgm* format in GIMP. Open you image in GIMP, then click on Files-> Export As, give it a name on the very top, then on the bottom click on Select File Type (by Extension), scroll down, select .ppm (or .pgm). After that it will bring up a box with options to save in ASCII or raw format. Select format and save it.

Part III. Where to Get Help

1. First of all, try to use textbook, online sources and search engine of your choice to look at code examples, programming concepts explanation, etc. You need to learn to be independent and to be able to use available resources to help yourself.

2. You can also get help with this assignment from the course teacher or TAs. Though TAs may not be familiar with the details of the assignment or design requirements, they can help you with programming questions and debugging.

3. Many questions can be answered in e-mail. Please do not e-mail code unless you were instructed to do so. Do this ahead of time, not at the last moment. You are still responsible for submitting your assignment on time, whether you got your question answered or not at the last moment.

4. **Most importantly**: make a plan A and a plan B. You should always have a plan A and a plan B!! Unexpected circumstances happen, and it is better to be safe, than sorry. Start working right away, DO NOT WAIT TILL THE LAST MOMENT. Make frequent backups!!!!