CS 3005: Programming in C++

Portable Pixmap Format

The Portable Pixmap Format (PPM) is an open definition digital image file format. It defines how to store data in a file such that another program can read the file and correctly display the image that was stored.

PPM Image File Format

A PPM image has a width, a height, and a maximum color value. The width is the number of pixels in a row. The height is the number of rows in the image. The maximum color value is the highest number that can be stored in one channel of a pixel. Each PPM image is allowed to have a different maximum color value. This allows the image file to decide what number represents 100% brightness. In our implementation, the maximum color value will need to be at least 1 but no more than 255.

As you work on this assignment, it will be useful to know the way a PPM image is stored in a file. The file header information is text (ASCII) words, separated by white space. After the newline character that follows the maximum color value, the rest of the data in the file is binary (not-ASCII), with one byte per color channel, or three bytes per pixel. There are no newlines or other white space characters in the pixel data.

```
P6 WIDTH HEIGHT MAX_COLOR_VALUE\n
BINARY REPRESENTATION OF COLORS FOR EACH PIXEL IN THE SAME ORDER AS THE COLOR FILE
```

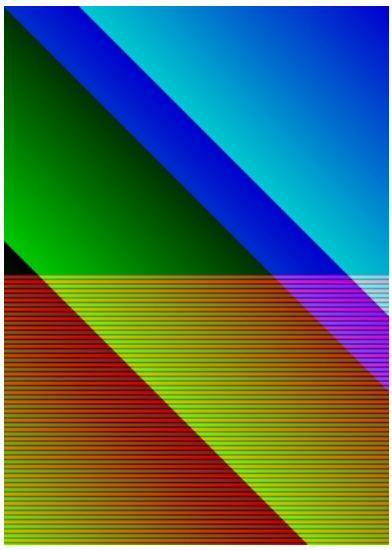
Assignment

In this assignment, you will update the project from the previous assignment. You will add code to be able to store the additional information needed for a `PPM` image, in addition to the `Image` information. You will also create a program to create a PPM image using an algorithm to set the pixel values, using an image size specified by the user. Finally, the program will save the image to a PPM format file, using the name specified by the user.

Potential Session

```
$ ./image_file
Image height? 350
Image width? 250
Output filename? assignment3.ppm
$ ls -l assignment3.ppm
-rw-rw-r-- 1 cgl cgl 262515 Sep 17 19:03 assignment3.ppm
```

This image appears like this:



Download the ppm file

Programming Requirements

The following files must be updated or created and stored in the `src` directory of your repository.

Create `PPM.h`

This file must include the declaration of the `PPM` class. The `PPM` class must inherit from the `Image` class that already exists, so you don't have to recreate all of the data storage code.

The `PPM` class must have an integer data member to store the maximum color value.

Your `PPM` class must have the following methods.

- `PPM( );` The default constructor. A default PPM has max color value of 1, and a default constructed `Image` portion.
- `PPM( const int& height, const int& width )`; The max color value should be set to 1. The `Image` portion should be initialized with the `height` and `width` parameters.
- `int getMaxColorValue( ) const;` Returns the maximum color value of the PPM.
- `bool valueValid( const int& value ) const;` Checks if `value` is a legal color value for this image. Legal means at least 0 and no more than the maximum color value. Returns `true` if it is legal, `false` otherwise.
- `void setMaxColorValue( const int& max_color_value );` Change the maximum color value of the PPM. Only values 1 to 255, inclusive should be accepted. If the value is not accepted, make no changes.
- `void setChannel( const int& row, const int& column, const int& channel, const int& value );` If `value` is valid (use the `valueValid` method), then call `Image::setChannel()` passing in the parameters. If `value` is not valid, do nothing.
- `void setPixel( const int& row, const int& column, const int& red, const int& green, const int& blue );` Set all three channels for the specified pixel. Should use `setChannel` to do the work.
- `void writeStream(std::ostream& os) const;` Writes the PPM data to the output stream `os`. Uses the format mentioned above. The first line of data is ASCII text, and the rest is binary data.

Create `PPM.cpp`

This file must implement all of the methods of the `PPM` class declared in `PPM.h`.

Update `image_menu.h`

Add the following function declarations to the file.

- `void writeUserImage( std::istream& is, std::ostream& os, const PPM& p );`
- `void stripedDiagonalPattern( std::istream& is, std::ostream& os, PPM& p );`
- `int assignment3( std::istream& is, std::ostream& os );`

Update `image_output.cpp`

This file must include the implementations for these functions:

- `void writeUserImage( std::istream& is, std::ostream& os, const PPM& p );` Uses `getString` to ask the user for the "Output filename? ", opens the output file in binary mode, writes the PPM object to the file's stream (using `writeStream`), and closes the file stream.

Update `image_drawing.cpp`

Implement the following functions:

- `void stripedDiagonalPattern( std::istream& is, std::ostream& os, PPM& p );`

Description of the `stripedDiagonalPattern()` Function

This function will use `getInteger` to ask the user for the height of an image and again for the width of an image. The questions must be asked in the order demonstrated in the potential session with identical prompts.

Sets the height and width of the PPM parameter, using the height and width specified by the user. Sets the maximum color value of the object to the sum of the height and the width, divided by 3. If this value is larger than 255, sets the maximum color value to 255 instead. For examples: if the height and width are 300 and 200, then the maximum color value will be (300 + 200)/3 = 166. However, if the height and width are 500 and 400, then the maximum color will be 255, because (500 + 400)/3 = 300 which is greater than 255.

If a pixel is in the top half of the image, sets the red channel of the pixel to 0. If a pixel is in the bottom half of the image and the row number is a multiple of 3, sets the red channel of the pixel to 0. If a pixel is in the bottom half of the image, and the row number is not a multiple of 3, sets the red channel to the maximum color value.

Sets the green channel of a pixel to the *remainder* of `(row + width - column - 1)` divided by one more than the maximum color value. Read that sentence carefully to get the math correct.

Sets the blue channel of a pixel to 0 if the column number is less than the row. Otherwise, sets the blue channel to the maximum color value.

When determining top half vs bottom half or left half vs right half, consider that `x/2` results in an integer, when `x` is an integer. The top half of the image includes all rows who are strictly less than half of the height. The left half of the image includes all columns who are strictly less than half of the width.

(Note: if the user gives nonsense values for height, width or filename, the image saved will not make sense, and it's their fault, not yours.)

Update `controllers.cpp`

Implement the following functions:

- `int assignment3( std::istream& is, std::ostream& os );` Creates a default constructed `PPM` object. Calls `stripedDiagonalPattern` and `writeUserImage`. Returns 0.

Create `image_file.cpp`

This file must include the implementations of the following functions:

- `int main( );` This function should call `assignment3`, passing in `std::cin` and `std::cout` as the input and output streams to use. This function should return what `assignment3` returns.

Update `Makefile`

This file must now also include the rules to build the program `image_file`. The following commands should work correctly.

- `make hello` - builds the hello program
- `make questions_3` - builds the questions_3 program
- `make ascii_image` - builds the ascii_image program
- `make image_file` - builds the image_file program
- `make` - builds all programs

Additional Documentation

- C++ Reference
- Examples from class
- Writing Data to Streams
- Windows PPM Viewer
- Linux PPM Viewer: `eog file.ppm`

Show Off Your Work

To receive credit for this assignment, you must

- use git to add, commit and push your solution to your repository for this class.
- successfully pass all unit tests and acceptance tests

Additionally, the program must build, run and give correct output.