CIS 201 – Computer Science I
Laboratory Assignment 10

Introduction

In this lab you will manipulate images using classes from the Georgia Tech MediaComp project. See `http://coweb.cc.gatech.edu/mediaComp-plan/101` for more information on this project.

Files from this project can be found in

```
/home/student/Classes/MediaComp
```

and the documentation for the Java class used in this project can be found by following the `Documentation` link on the CS homepage.

Reading and displaying a picture

Copy the file

```
/home/student/Classes/201/Labs/Lab10/Pic.java
```

into a suitable working directory. You should be able to compile and run this file directly, with a copy of the raptor picture displayed on your screen.

File other `.jpg` files on your system (use the `locate` command!), or find ones on the web and download them to your current directory. Modify the program so that the picture to be displayed is given as a command line parameter, in `args[0]`.

Checkpoint 1

Show us some of your pictures!

Negative images

Copy the file

```
/home/student/Classes/201/Labs/Lab10/Pic1.java
```

into your working directory and change the `/* FIXME */` part so that it will use the command line parameter `args[0]` as the filename.

If you compile and run this program, it should display the picture exactly as the previous program, since the "new" pixels defined using `setColor` have the same red, green, and blue values as the original ones.

To make a "negative image", you will want to change the values of red, green, and blue values so that they go from 255 to zero instead of from zero to 255. Specifically, given the value `r` for red, find a simple expression that will evaluate to 255 if `r` is zero that will evaluate to zero if `r` is 255, and that will handle values inbetween as well, as illustrated in the following table

old `r`	new `r`
0	255
1	254
...	...
255	0

Then do the same thing for green and blue. Your actual parameters `r`, `g`, and `b` in the `new Color(...)` constructor should be replaced by these expressions.

Run your modified program with several pictures of your choosing.

(If you wish, you can use the grayscale conversion as described in class to produce a black-and-white "negative" image instead of the color one that we are describing here.)

Checkpoint 2

Show us your work at this point. Your displayed pictures should look like "negatives".

Working with pixels in the (x,y) coordinate system

Instead of treating the image pixels as a one-dimensional array, we can also look at them in the (x,y) coordinate system, where (as we do in FANG), the origin $(0,0)$ is at the top-right corner. The height and width of the picture can be obtained using methods defined in the `Picture` class.

Copy the file

```
/home/student/Classes/201/Labs/Lab10/Pic2.java
```

into your working directory and examine it using an editor. You can see that the program produces a picture where all of the reds are removed (*i.e.*, the red values are all zero).

Use this idea to modify the program so that the resulting picture mirrors the left-hand side onto the right-hand side of the picture. You can do this by scanning only the left half of the picture (with the *x* values going from 0 to half of the width) and changing the color of the corresponding right pixel using something like

```
rpx.setColor(lpx.getColor());
```

where `rpx` is the right pixel and `lpx` is the left pixel. (Of course, these aren't the actual variables you'll use.)

Checkpoint 3

Show us your work at this point.

Prisoner

Make another copy of this program named `Pic3.java`. Change it so that it puts the image "behind bars" by putting black vertical bars strategically across the image.

To accomplish this, first define a method with the following documentation and signature:

```
/**
 * draw a black vertical bar on a picture
 * @param p  - the Picture
 * @param bw - the width (in pixels) of the bar to be drawn
 * @param x  - the starting x-coordinate of the bar
 */
public static void drawBar(Picture p, int bw, int x)
```

To test your `drawBar` moethod, make a call to `drawBar` in your `main` method to see that it works with a simple case. Then write a `for` loop that will draw several vertical bars, all of the same width, across your picture.

Checkpoint 4

Show us your work at this point.

Try something!

Well, try something of your own choosing.

Checkpoint 5

And show us!