**Lab for Introduction to Manipulating Pictures**

**Barbara Ericson**

To manipulate a digital picture we will create a picture object and then first get a one-dimensional array of all the Pixel objects in the picture and loop through the array. We will change the red, green, and/or blue values in the pixels in order to change the picture. Color is a class in the package java.awt and it represents color using red, green, and blue values that range from 0 to 255.

Bring up a Color Chooser by typing the following in the interactions pane of DrJava:

> ColorChooser.pickAColor();

Click on the "RGB" tab and use the sliders to make different colors:

1. What are the red, green, and blue values that make the color yellow?
2. What are the red, green, and blue values that make the color green?
3. What are the red, green, and blue values that make the color pink?
4. What are the red, green, and blue values that make the color black?
5. What are the red, green, and blue values that make the color brown?

We can create a picture object by passing it a fully qualified file name as a string:

Picture p = new Picture("f:/mediasources/beach.jpg");

We can get a one-dimensional array of pixels in the picture using

Pixel[] pixelArray = this.getPixels();

When we get all the pixels in a picture as a one-dimensional array we get the pixels in the first row followed by the pixels in the second row, then the third row and so on. We can use a for-each loop to loop through all the pixels in the array of pixels.

for (Pixel pixel : pixelArray)

We can get the red, green, or blue value from a pixel using getRed, getGreen or getBlue

int red = pixel.getRed();

We can set the red, green, or blue value in a pixel using setRed(int value), setGreen(int value), setBlue(int value)

pixel.setRed(red \* 2);

Here is a method to loop through all the pixels in a picture and set the red value to half the original value.

**public void** decreaseRed()

{

Pixel[] pixelArray = **this.**getPixels();

**int** value = 0;

// *loop through all the pixels in the array*

**for** (Pixel pixelObj : pixelArray)

{

// *get the current red value*

value = pixelObj.getRed();

// *decrease the red value by 50% (1/2)*

value = **(int)** (value \* 0.5);

// *set the red value of the current pixel to the new value*

pixelObj.setRed(value);

}

}

1. Write the method increaseRed that will double the red value in the current picture and test it using:

String fName = "f:/mediasources/beach.jpg";

Picture pict = new Picture(fName);

pict.explore();

pict.increaseRed();

pict.explore();

1. Call increaseRed several times on the same picture. Does the amount of red ever go over 255?
2. Write a method clearBlue that sets the blue value in all the pixels in a picture to 0.
3. What a method keepOnlyBlue that sets the red and green values in all the pixels in a picture to 0 (keeps only the blue). Can you still make out what is in the picture? Also write keepOnlyRed and keepOnlyGreen.

Turn in Picture.java.