Homework 4

Due Monday, July 7, at 9 AM (GMT-7)

CSc 345 – Summer 2014

Instructor: Qiyam Tung

Instructions

1. This is an individual assignment. You must do your own work.
2. If you are having difficulty and need to ask a question you can:
   (a) Ask questions in class.
   (b) Stop by my office hours (or make an appointment).
   (c) Post a question on Piazza.
   (d) Post a private question on Piazza if the question is too specific.
3. I highly encourage you to post your test cases (input and output) on Piazza.

Sorting Images (100 points possible)

In computer vision and image processing, we are often concerned with extracting useful information from images. For example, in object detection, the distribution of colors is used as evidence for the existence of a particular object (e.g. tiger). Other applications include identifying the dynamic range of the image.

In this assignment, your task is to sort the the intensity values of 10×10 (100 pixels) subimages of the original image. It is often helpful to be able to visualize such representations, so your output should include the sorted subimages of the original image (see Figure 1).

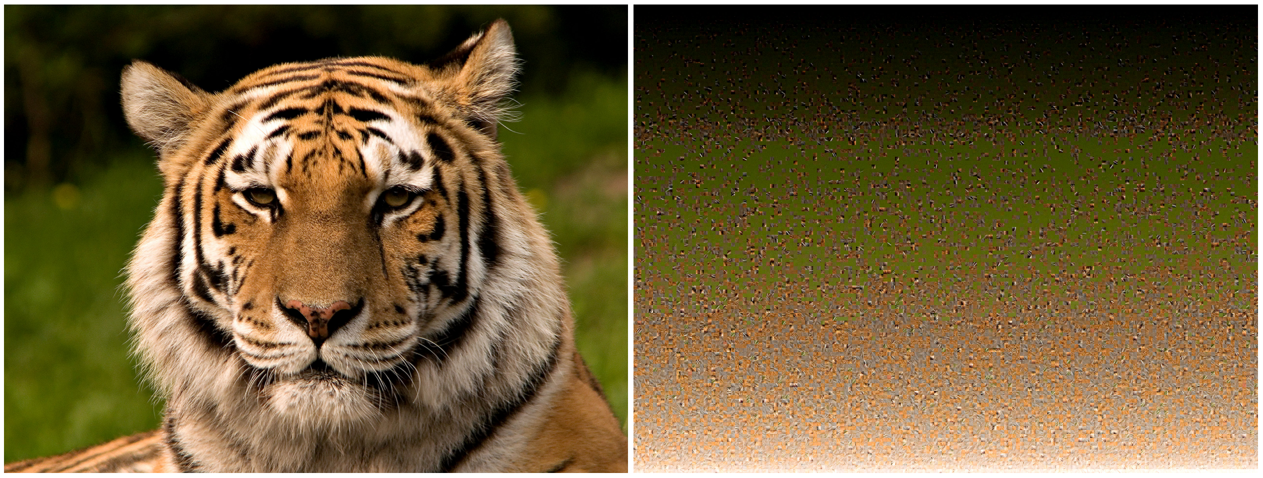

Figure 1: Given an image, you should sort it by intensity as 10×10 blocks (image of tiger from Wikipedia.com)

The Assignment

Write a Java program named cs345_prog1.java so that it sorts an image.

Specifically,

1. Divide the image into 10×10 blocks of pixels (or subimages).
2. Calculate the average intensity of each of these subimages.
3. Sort them while minimizing the number of writes.
4. Place the sorted blocks back into the image (sorted from top to bottom, left to right).

In this assignment, "writes" means the number of times you modify the array you're trying to sort. For example, `array[i] = array[j]` is one write, but `swap(array, i, j)` is two, since swapping two elements requires writing back to the array twice.

If you're not doing an in-place sort (using more than $O(1)$ memory), then include the count for the number of writes you are using for your dynamically allocated structures (e.g. the output array for Mergesort or Radix/Counting sort).

Your goal in "minimizing" the number of writes is to simply beat Insertion Sort. There are many ways to do this.

The Program

Compilation Instructions

You must create a main Java program called cs345_prog1.java. You may add additional helper classes if you wish, but there is no need to get too creative. Specifically, your Java classes should compile with the single command

```
bash4.2: javac *.java
bash4.2:
```

Sample Run

The arguments to your program should be a list of images (whose dimensions will be divisble by 10) and write the sorted image out to file (in .png format). You should also print the number of writes to stdout.

The following are example runs on our implementation of the assignment using Insertion Sort.

```
bash4.2: java cs345_prog1 tiger.jpg
Sorting tiger.jpg as tiger_sorted.png
  Num writes: 138990900
bash4.2:
```

Here is an example of a run with multiple images.

```
bash4.2: java cs345_prog1 tiger.jpg aurora.jpg amoeba.jpg
Sorting tiger.jpg as tiger_sorted.png
  Num writes: 138990900
Sorting aurora.jpg as aurora_sorted.png
  Num writes: 6627442
Sorting amoeba.jpg as amoeba_sorted.png
  Num writes: 1466970016
bash4.2:
```

Calculating Intensity

To do sorting, we need to be able to convert the RGB color to its appropriate intensity. Use the following formula to compute the intensity

$$I = 0.2989R + 0.5870G + 0.1140B \tag{1}$$

Use the `BufferedImage` class to read and write Note that you should store intensity as a double, while the RGB values are stored in `BufferedImage` as integers. `BufferedImage` stores its RGB values in a single 32-bit `int` value. That means you need to do the following computation to extract the values

```
b = (rgb >> 0)  & 0xFF;
g = (rgb >> 8)  & 0xFF;
r = (rgb >> 16) & 0xFF;
```

Handling Errors

Your program should do appropriate error handling. Specifically,

- Print the usage statement when not given any images. That is,

```
      bash4.2: java cs345_prog1
      Usage: java cs345_prog1 image1 [image2 [image3 [...]]]
      bash4.2:
```

- Print an error message and exit when one of the filenames given cannot be read. This should occur before any image is sorted and written to file
- Print an error message if one of the images is not divisible by 10 and exit. Again, this should occur before sorting and writing to file.

Documentation

See the code requirements document.

Grading

We will be grading your program based on the sorted images you output (it should be exactly the same as ours) as well as the number of writes (which should be less than or equal to our implementation of Insertion Sort).

20% of your grade will go to documentation.

What to Turn in

Turn in all your Java classes: `cs345_prog1.java` and any other helper classes that you may have.

Miscellaneous

In order to do this project, you may find the following classes useful.

- `java.awt.image.BufferedImage`: image class.
- `java.util.*`: collection of List data types.
- `java.util.Collections`: for the swapping function.
- `java.lang.Comparable`: for comparing subimages.
- `javax.imageio.ImageIO`: for reading/writing images to file.

Refer to the Java documentation page for more information: http://docs.oracle.com/javase/7/docs/api/.

You may NOT use any library functions for sorting. You need to write the sorting function you wish to use. Given that you have the pseudocode and the book's implementation of the basic sort, this really shouldn't be an issue. Additionally, you need to keep track of the number of writes, so simply calling a library sorting function wouldn't allow you to compute the count.

You should test your program on `lectura.cs.arizona.edu` as that is where we will be testing your programs. While Java known to be portable, that does not guarantee that it will run perfectly on different machines due to different system settings.