Hack 7.0

Lists & Arrays
Computer Science I – Java

Department of Computer Science & Engineering
University of Nebraska–Lincoln

Introduction

Hack session activities are small weekly programming assignments intended to get you started on full programming assignments. Collaboration is allowed and, in fact, *highly encouraged*. You may start on the activity before your hack session, but during the hack session you must either be actively working on this activity or *helping others* work on the activity. You are graded using the same rubric as assignments so documentation, style, design and correctness are all important.

Exercises

To get more practice working with Java `List`s, you will write several methods that involve operations on `List` of integers. In particular, implement the following.

1. Write a method that, given a `List` of integers and an integer x determines if it contains x anywhere within the `List`. It should return `true` if it does, `false` otherwise.

   ```
   public static boolean contains(List<Integer> list, int x)
   ```

2. Write a method that, given a `List` of integers and an integer x determines if it contains x within the range of the two provided indices i, j (inclusive of both indices). It should return `true` if it does, `false` otherwise.

   ```
   public static boolean containsWithin(List<Integer> list, int x, int i, int j)
   ```

3. Write a method that, given a `List` of integers, and a "new size" creates a new deep copy of the `List`. However, instead of its original size, the size of the new `List` should be the new size. If the new size is less than the old size, only the first `newSize` elements should be copied over. If the new size is greater than the original size, then the new `List` should be padded out with zeros.

   ```
   public static List<Integer> paddedCopy(List<Integer> list, int newSize)
   ```

4. Write a method that, given a `List` of integers reverses the elements in the `List`. For example, if the original `List` was `[10, 15, 5, 25, 0]` the new array should be `[0, 25, 5, 15, 10]`.

   ```
   public static void reverse(List<Integer> list)
   ```

5. Write a similar method that creates and returns a new, deep copy of the given `List` of integers but with its elements in reverse order.

   ```
   public static List<Integer> reverseCopy(List<Integer> list)
   ```

Image Manipulation

You'll get more practice with 2-dimensional arrays by writing several methods to manipulate images. In Java, you typically use a built-in representation of an image, but we've adapted this class and instead represent an image as a 2-dimensional array of `RGB` "pixels" (you've worked with this class in previous labs and hacks). We've written a few helper methods to load and save images to a variety of image file formats (jpg, bmp, gif, png, among others).

You can manipulate individual pixels as you would any array element. For example:

```
RGB image[][] = ...;
RGB tempPixel = image[0][0];
...
image[i][j] = tempPixel;
```

An image is represented by an $h \times w$ (height by width) 2D array of these image can be represented as a two dimensional array of these `RGB` elements.

We've provided a library of methods to load and save a file (you'll need to RTM) and specified several method signatures for methods you need to implement.

- `copyImage()` should produce a deep copy of the given image.
- `flipHorizontal()` should flip the image horizontally as depicted in Figure 2.
- `flipVertical()` should flip the image vertically as depicted in Figure 3.

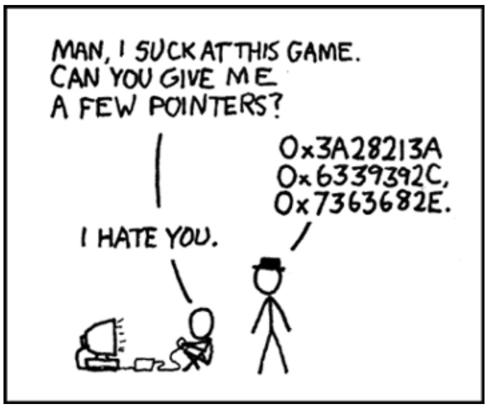
MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A 0x6339392C, 0x7363682E.
I HATE YOU.

Figure 1: Original Image

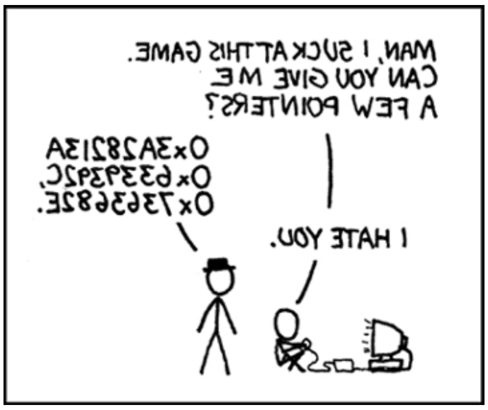

Figure 2: Flipped Horizontally

- `rotateClockwise()` should produce a new image that is rotated 90 degrees clockwise. This function must produce a new image because an $h \times w$ sized image that has been rotated will be a $w \times h$ image. This operation is depicted in Figure 4.

Instructions

- For the warm-up, place all your methods with documentation in a java source file named `ListUtils.java`
- In addition, you'll want to create a main test driver program that demonstrates at least 3 cases per function to verify their output. You need not hand in this test file, however.
- For the Image Manipulation section, we have provided starter code in two source files: `ImageUtils.java` and `ImageDriver.java`. The first source file contains the starter code for the methods you must write. The driver file contains a main that you can use to test your program on an image of your choice.

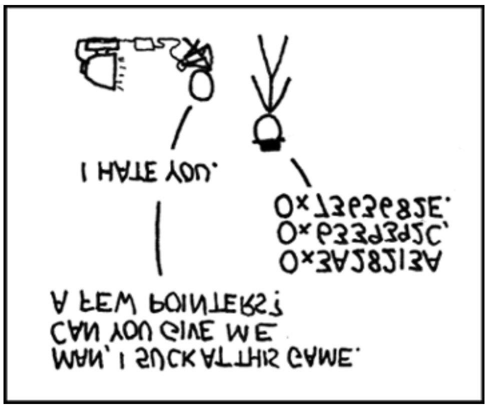
MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A 0x6339392C, 0x7363682E.
I HATE YOU.

Figure 3: Flipped Vertically

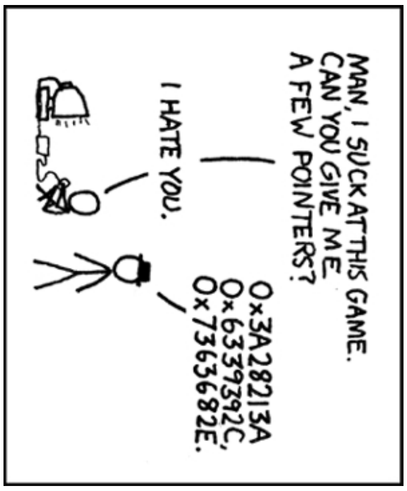
MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A 0x6339392C, 0x7363682E.
I HATE YOU.

Figure 4: Rotated Clockwise

- You should test all your functions with an image (load it, manipulate it and save it) of your choice.
- As a first step, you should add documentation to all your functions. Use this as an opportunity to discuss how the functions should work and to *whiteboard* your designs and solutions with other students.
- You are encouraged to collaborate any number of students before, during, and after your scheduled hack session.
- You may (in fact are encouraged) to define any additional "helper" methods that may help you.
- Include the name(s) of everyone who worked together on this activity in your source file's header.
- Turn in all of your files via webhandin, making sure that it runs and executes correctly in the webgrader. Each individual student will need to hand in their own

copy and will receive their own individual grade.