Homework #4

CS 210, **Fundamentals of Computer Science I**
Instructor: Tatiana Harrison

Instructions

This homework assignment covers methods, which we have started discussing (chapter 5), and assumes understanding of the concepts that are covered in chapters 1 through 4. Several of the book questions are from chapter 4 of the textbook, and are meant to help you review (and master) *for* and *while* loops. The questions from the back of the chapter are worth the point values indicated below (a total of 30 points). The two programming tasks are worth a total of 50 points. Check the course web page for TA office hours. Submit your answers to the book questions, and your .java file zipped, to Canvas.

End-of-chapter Questions – 30 points
Take "Homework 4 Q&A" Canvas quiz

Programming Task – Celsius Fahrenheit Converter - 40 points

For this programming task you will write a program that has a method that will convert Fahrenheit to Celsius, or vise-versa. This program is VERY SIMILAR to what you will do in Lab 6.

To complete this task:

1. In jGRASP, create a new java file, and save it as `TemperatureConverter.java`.

2. Write a method that:
   - Has the method modifiers `public static`
   - Has the name `convertTemp`
   - Receives two parameters: the first is of type *double*, which you should call `temperature`, and the second is of type `String,` which you should call `temperatureScale`
   - The return of the method should be *void*
   - The body of the method should have an *if,* `else-if,` *else* statement. Depending on the value of the argument `temperatureScale`, the method should convert the value saved in the variable *temperature* from Fahrenheit to Celsius if `temperatureScale` is *f*, or convert temperature from Celsius to Fahrenheit if `temperatureScale` is *c*. If `temperatureScale` is neither *f* nor *c*, the method should print an error message. The skeleton of the *if* statement is the following:

```
if ( /* check if temperature scale is equal to the letter f */ ){

   // code that converts from Fahrenheit to Celsius
   // and prints the result to the screen

} else if (/* check if temperature scale is equal to the letter c */){

   // code that converts from Celsius to Fahrenheit
   // and prints the result to the screen

} else {

   // code that outputs a message indicating that an incorrect
   // option was selected

}
```

- Use the following conversion formula, to convert the value that is input to the method:

Celsius = 5 / 9 * (Fahrenheit - 32)

Fahrenheit = 32.0 + (Celsius * 1.8)

- Declare a variable to hold the calculation result, and use the *System.out.println* method to print to the screen the output of the calculation.

1. Write a main method, that:

- Creates a `Scanner` to read input from the keyboard.
- Prompts the user for a number, and saves the user's input into a variable of type `double`.
- Prompts the user to indicate whether he/she has entered a Fahrenheit or Celsius temperature, and saves the user's input into a variable of type `String`.
- Invokes the `convertTemp` method, that you've already written.

A sample invocation of the program is shown in the following three boxes:

```
Hello. This program will convert Fahrenheit to Celsius, or vise-versa.
To get started please enter a temperature: 45
You've entered 45.0
Did you submit Fahrenheit or Celsius?
Type f for Fahrenheit, or c for Celsius: c
45.0 degrees Celsius is equal to 113.0 Fahrenheit
```

Sample invocation, to convert Celsius to Fahrenheit

```
Hello. This program will convert Fahrenheit to Celsius, or vise-versa.
To get started please enter a temperature: 78
You've entered 78.0
Did you submit Fahrenheit or Celsius?
Type f for Fahrenheit, or c for Celsius: f
78.0 degrees Fahrenheit is equal to 25.555555555555557 Celsius
```

Sample invocation, to convert Fahrenheit to Celsius

```
Hello. This program will convert Fahrenheit to Celsius, or vise-versa.
To get started please enter a temperature: 63
You've entered 63.0
Did you submit Fahrenheit or Celsius?
Type f for Fahrenheit, or c for Celsius: g
Sorry, you've entered a bad choice.
```

Sample invocation, when user provides a bad option

Programming Task – Invoking methods - 30 points

For this programming task, you'll be given a java file `ManipulateImage.java`, that contains several predefined methods that you don't have to (SHOULDN'T) edit. You'll only need to complete two methods that are partially given to you in the java file. This programming task is meant to familiarize you with writing and invoking methods.

The program that you will complete is an image manipulation program. It contains several methods that are used to read, manipulate, and write out black and white image files of type *jpg*. Your program will prompt the user to type the name of the input jpg image file (you'll be given `bobcat.jpg` to test your program). If a user supplies the letter *q* in response to the file input question, the program will end. If a valid file name is provided, the program prompts the user to indicate what method should be invoked to manipulate the image, and then what the output file should be named. One of several options can be invoked to manipulate the image. The manipulation choices will be (*v*) invert, (*h)* hide, (*u*) unhide, (*b*) blackout, (*w*) whiteout, and (*i*) to get information. The methods to perform invert, hide, unhide, and blackout, are provided in the java skeleton file. You'll need to complete the method that is called when the option *i* (information) is invoked, and you'll need to complete the main method as well. Figure 1 shows a sample invocation/interaction with the program `ManipulateImage`, and Figure 2 shows the files that the program generated when it was invoked using the commands in Figure 1.

```
What image do you want to edit? Type q to quit: bobcat.jpg
What action? Invert (v), blackout (b), hide (h), unhide (u), whiteout (w), information (i): i
The image bobcat.jpg has a height of 232 pixels.
The image bobcat.jpg has a width of 217 pixels.
The image bobcat.jpg is of type 5

What image do you want to edit? Type q to quit: bobcat.jpg
What action? Invert (v), blackout (b), hide (h), unhide (u), whiteout (w), information (i): v
What is the name of the output image? bobcatInvert.jpg
The image bobcat.jpg has been invereted, and saved to bobcatInvert.jpg

What image do you want to edit? Type q to quit: bobcat.jpg
What action? Invert (v), blackout (b), hide (h), unhide (u), whiteout (w), information (i): b
What is the name of the output image? bobcatBlackout.jpg
The image bobcat.jpg has been blackened, and saved to bobcatBlackout.jpg
```

```
What image do you want to edit? Type q to quit: bobcat.jpg
What action? Invert (v), blackout (b), hide (h), unhide (u), whiteout (w), information (i): w
What is the name of the output image? bobcatWhiteout.jpg
The image bobcat.jpg has been whited, and saved to bobcatWhiteout.jpg

What image do you want to edit? Type q to quit: bobcat.jpg
What action? Invert (v), blackout (b), hide (h), unhide (u), whiteout (w), information (i): h
What is the name of the output image? bobcatHide.jpg
The image bobcat.jpg has been 'hidden', and saved to bobcatHide.jpg

What image do you want to edit? Type q to quit: bobcatHide.jpg
What action? Invert (v), blackout (b), hide (h), unhide (u), whiteout (w), information (i): u
What is the name of the output image? bobcatUnhide.jpg
The image bobcatHide.jpg has been 'un-hidden', and saved to bobcatUnhide.jpg

What image do you want to edit? Type q to quit: q
Okay, buh-bye
```

Figure 1: Sample invocation/interaction with program ManipulateImage

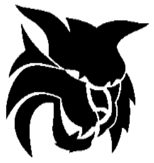	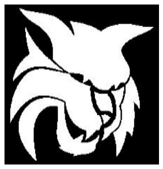			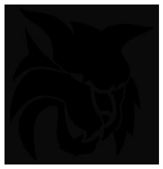	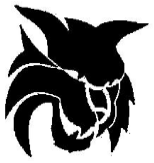
bobcat.jpg	bobcatInvert.jpg	bobcatWhiteout.jpg	bobcatBlackout.jpg	bobcatHide.jpg	bobcatUnhide.jpg

Figure 2: The original image (bobcat.jpg), and the output images of the program ManipulateImage when it is invoked with the commands shown in Figure 1.

Step-by-step instructions:

1. Download the files `bobcat.jpg` and `ManipulateImage.java` from the schedule page of the course website, and save it to the location where you are saving your work. Open the `ManipulateImage.java` file in jGRASP.

2. The first part of the main method has been written for you. Read what's there. Familiarize yourself with the code. Especially, pay attention to the following statements:

```
    BufferedImage inputImage = loadImage(fileName);
    BufferedImage outputImage = null;
```

   This has been written for you. <u>Do not modify it.</u> The first statement declares a variable named `inputImage` of type `BufferedImage`, which will contain the data that is the image that you want the program to manipulate. The second statement declares a reference variable named `outputImage` of type `BufferedImage`, and will refer to the data that is the image.

3. In the file `ManipulateImage.java`, where it says, COMPLETE IF CONDITIONAL, write code that checks if a user has entered the letter *q*, in which case issue a break statement, which will cause the while loop to terminate, and hence the program to end. Refer to section 4.8 of the textbook if you need to review break statements, or review the lecture slides.

4. In the file `ManipulateImage.java`, where it says, METHOD INVOCATION SECTION, complete the code, so that the correct method is invoked, based on what option the user inputs:

   - If user supplies *v*, then set the value of `outputImage` to be the output of the method `invertImage(inputImage);`
   - If the user supplies *b*, then set the value of `outputImage` to be the output of the method `blackoutWhiteoutImage(inputImage, "b");`
   - If the user supplies *h*, then set the value of `outputImage` to be the output of the method `hideImage(inputImage);`
   - If the user supplies *w*, then set the value of `outputImage` to be the output of the method `blackoutWhiteoutImage(inputImage, "w");`
   - If the user supplies *i*, then invoke the method `printImageInfo(inputImage);`

   To help you out, here is the code that you would need to write when the user supplies the letter "v" as the option:

   ```
   outputImage = invertImage(inputImage);
   ```

5. Complete the method `printImageInfo`. Instructions on how to do this are in the comments for that method in the java file.

6. Be sure to test your program. If there are no syntax errors, then the program at this point is fully functional. Be sure that the images that are output/generated by the program are identical to those that are shown in Figure 2.