Grade

Camilo Ortiz & Kelley Baumann

Overall grade
We think we deserve 109 points based on our lean and efficient code and extensive use of previously unseen Java and Python libraries to create a novel program. We cover these in more detail below.

Functionality: 16/20
Our project is robust, except if the user selects a non-.jpg or .png file as their image. In our front page's instructions, we specify this such that the user doesn't do this. It is impervious to keyboard clicks, but some computations take a little while to compute, so the user has to be a little patient when the program slows down a bit.

Design: 19/20
Although we were dealing with complicated functions, such as communicating between Java and Python, we maintained robust and streamlined code. We utilized inheritance and static variables to communicate between pages, and made them our own bootstrapped "state variables". No code in our SRC folder is unnecessary and we didn't overcomplicate easy solutions.

Creativity: 18/20
We brought together elements of our favorite image-altering softwares, including the Meme Fryer (link: https://deepfriedmemes.com/) and facial recognition common to Snapchat filters and Apple's Mac Photo Booth to create something new of our own. Meme/image distortion software does exist online, but we have yet to find one that incorporates facial recognition.

Sophistication: 20/20
Although the code quantity of our final project would not appear to reflect several weeks of lab assignments, we believe that the features we included in our final design pushed us to learn about and implement concepts (JFileChooser, JButton, OpenCV, etc.) we had never seen before. As such, the effort sustained to complete the project does accurately reflect a multi-week group project.

For example, in this project we had to fit many different pieces together, from Python runtimes to handling threads such that there wouldn't be an error when Java tries to read a file before our Python program has finished writing it.

Broadness: 16/20
We were able to implement six of the eight categories required for broadness:

[1] New Java libraries: We utilized a number of previously unseen Java libraries in our program, including JFileChooser, JButton, ActionEvent, ActionListener, ImageIO, BufferedImage, and AffineTransform. We also used OpenCV, a Python library, for facial recognition.

[2] Subclassing: The way our pages communicate with one another is thanks to subclassing. They are all inherited from the same Page class that has static variables and that's how the Main class knows what page to display. Some methods are shared (such as the resize method). Removing inheritance would completely decimate our program.

[5] Built-in data structures: To keep track of head/eye rectangle coordinates, we used a Vector<> of Integer arrays. We chose a Vector because we wouldn't have to worry about implementing a resize() method when the array got too large

[6] File input/output: This program is truly founded on the ability of the user to upload their own image, distort it, and then save the new version to their own directory again. All of the file input and output is facilitated by buttons that open pop-up windows that allow users to navigate to find desired images and directories.

[7] Randomization: One of our features, the "Random" button, runs a python distortion program on the image that uses random numbers [-50, 51) to translate pixels a random amount, giving the user a new image every time.

[8] Generics: We use generics in our built-in data structures. Vectors are generic, and we initialize them as Vector<Integer[]>.

Code Quality: 20/20

To reiterate what we stated in the design section, our code is lean and efficient. We also utilized a number of Java and Python libraries that drastically improved the capabilities of our project. We have left comments throughout the code to help you (and others) understand our process.