USEME

Code Execution

The main method where the Image processing application begins is from:
ImageProcessingApplication.java

In order to run the complete code one can run the following code-

1. Without file as a cmd argument:
   a. javac ImageProcessingApplication.java
   b. java ImageProcessingApplication

2. With file as a cmd argument:
   a. javac ImageProcessingApplication.java
   b. java ImageProcessingApplication -file “../res/Script-Tiger.txt”
3. Passing file during runtime:
   a. javac ImageProcessingApplication.java
   b. java ImageProcessingApplication
   c. run “../res/Script-Tiger.txt”

Please note that if the file path contains spaces in between please pass the complete address inside the quotes. Example of allowed format-
1. “../res/Script-Tiger.txt”
2. ‘../res/Script-Tiger.txt’

Also the address provided for file path is just for reference and can be changed as per user requirement to any other file path both relative and absolute where the scripting commands are stored.

Commands:

All the commands that are passed to the image application processing are in lower case. The application currently supports the following list of commands as a VALID input-
1. load image-path image-name: The image-path should be provided using single or double quotation marks like “image-path” or ‘image-path’. For eg-
   a. load “../res/Koala.png” Koala
   b. load ‘../res/Koala.jpg’ Koala
2. save image-path image-name: Similar to loading the image-path should be provided inside the quotation marks. For eg-
   a. save “../res/Koala-save.ppm” Koala
   b. save ‘../res/Koala-save.png’ Koala
3. red-component image-name dest-image-name: Creates an image with the red-component of the image. Similarly we can generate blue and green images using blue-component image-name dest-image-name and green-component image-name dest-image-name.

For eg - red-component Tiger Tiger-red-component

4. luma-component: Creates an image with the luma greyscale-component of the image. Similarly we can generate value and intensity images using value-component image-name dest-image-name and intensity-component image-name dest-image-name.
   For eg - luma-component Tiger Tiger-luma-greyscale

5. horizontal-flip image-name dest-image-name: To flip the image horizontally.
   For eg - horizontal-flip Tiger Tiger-horizontal

6. vertical-flip image-name dest-image-name:: To flip the image vertically.
   For eg - vertical-flip Tiger Tiger-vertical

7. brighten increment image-name dest-image-name: To change the brightness of the image. The 'increment' can be both positive value or negative value. The values should be an integer input.
   For eg - brighten 10 Tiger Tiger-brighter

8. sepia image-name dest-image-name: To change the tone of the image to sepia.
   For eg- sepia Tiger Tiger-sepia

9. rgb-split image-name dest-image-name-red dest-image-name-green dest-image-name-blue: splits the given image into three images containing different channel grayscale images.
   For eg - rgb-split Tiger Tiger-red Tiger-green Tiger-blue

10. rgb-combine image-name red-image green-image blue-image: Combine three r,g,b grayscale images into one single image containing data of all the existing ones.
    For eg - rgb-combine Tiger-red-tint Tiger-red Tiger-green Tiger-blue

11. blur image-name dest-image-name: It blurs the 'image-name' image using a fixed kernel.
    For eg - blur Tiger Tiger-blur

12. sharpen image-name dest-image-name: It blurs the 'image-name' image using a fixed kernel.
    For eg - sharpen Tiger Tiger-sharpen

    run script-file: Load and run the script commands in the specified file.
    For eg - run "../res/sample-file.txt"

13. compress percentage image-name dest-image-name: It compresses the image in the memory. The value of *percentage* should lie between 0 and 100 including decimal values. Anything beyond it should throw an error message.
    For eg - compress 35.6 Tiger Tiger-compressed-35

14. histogram image-name dest-image-name: Plots the histogram of an image pixel on a 256 X 256 image size.
    For eg - histogram Tiger Tiger-Histogram

15. color-correct image-name dest-image-name: Align the peaks of the r,g,b intensity curves.
    For eg - color-correct Tiger Tiger-Color-Correct

16. levels-adjust b m w image-name dest-image-name: Transforms the image according to a quad equation curve. Here B<M<W and all of them can be decimal values. If the constraint is violated then an error message is shown.
    For eg - levels-adjust 10 100 150 Tiger Tiger-Level-Adjustment-10

17. operation image-name dest-image split p: It splits the image for preview of image on which an operation is performed. It displays the operated image in one half of the preview and the original image on the other. The "split p" is used to define the percentage of the preview in which the operated image is visible.
    For eg - sepia Tiger Tiger-sepia-split split 55

18. -file name-of-script.txt: A script can be passed as a command-line option. Once a valid script is provided, the program exits.
    For eg - java ImageProcessingApplication -file "../res/Script-Tiger.txt"

19. "q" or "quit": It is used to exit the application.

<u>NOTE</u>:

- A combination of commands can be passed one after another in order to perform cascading operations on a single image.
- Most of the commands handle exceptions and display error messages to the view container, hence no assumptions have been made separately.