CS4551 Multimedia Software Systems (Spring 2018)

Homework1 (5%) – Image Quantization

- Due: Electronic submission via CSNS by Sunday, 02/18/2018.
- What to turn in:
  - Submit ***source code*** with necessary files for "compile and run".
  - Do NOT submit data files.
  - You **MUST** provide a ***readme.txt*** file containing all information to help with the grading process.
- If your program produces any compile errors, you will receive 0 automatically no matter how close your program is to the solution.
- You will receive penalty if you do not follow any of the given instructions.
- **You are not allowed to use any Java built-in image class methods, library, or tools to complete this homework.**
- Suggestions:
  - Do not create one mega-size main class.
  - Do not modify existing methods of `Image` class.
  - Feel free to extend `Image` class. You might want to add instance variables and methods to given classes, create subclasses, or add new classes.
  - Test your program with all test data.

0. What your program should do

Name your main application ***CS4551_[YourLastName].java*** (eg. CS4551_Doe.java) and expand the given template program to perform the following tasks.

Receive the input file as command line arguments.

<eg> On Command Prompt

```
java CS4551_Doe Ducky.ppm
```

Display the following main menu to the user and receive the user's input.

```
Main Menu-----------------------------------
1. Conversion to Gray-scale Image (24bits->8bits)
2. Conversion to N-level Image
3. Conversion to 8bit Indexed Color Image using Uniform Color
   Quantization (24bits->8bits)
4. Conversion to 8bit Indexed Color Image using [your selected method]
   (24bits->8bits)
5. Quit

Please enter the task number [1-5]:
```

After performing the selected task, go back to display the menu again.

1. Task 1 - Conversion a 24-bit Color to a 8-bit Gray-scale (10 pts)

Read the 24bit input PPM image. Convert 24-bit color pixels to gray-scale pixels. Use the following equation to compute the gray-scale value from `R`(Red), `G`(Green), and `B`(Blue) of each pixel.

```
Gray = round(0.299 * R + 0.587 * G  + 0.114 * B)
```

After the computation, make sure that `Gray` is an integer ranging 0-255. Truncate if it is not in the range. Display the output and save it into a PPM file.

2. **Task 2 - Conversion a 24-bit Color to a N-level (50 pts)**

Read the 24bit input PPM color image. Convert 24-bit color pixels to N-level pixels. For example, if N=2, it is to convert the input to a bi-level image (i.e. quantize the gray scale value into one of two values, 0 or 255). If N=4, it is to convert the input to quad-level images (i.e. quantize the gray scale value into one of four values, 0, 85, 170 or 255).

Steps:

a. Convert a 24-bit color image into a gray-scale image using the method in Task1.
b. Receive a value for N from the user. Assume that the user will enter 2 (1bit), 4 (2bits), 8 (3bits), or 16 (4 bits) for N.
c. Display and save two output images:
   - Output1: N-level image produced by the threshold method. For example, if N=2, use a threshold value (such as 128) to determine to quantize each gray scale value into 0 or 255.
   - Output2: N-level image produced by the error diffusion method.

3. **Task 3 - Uniform Color Quantization (40 pts)**

Read the 24bit color image. Quantize 24bit colors to 8bit colors using the Uniform Color Quantization method.

Step1: Generate the 8-bit color Look Up Table (LUT) by the Uniform Color Quantization and display the table to the console as shown below.

```
Index      R     G     B
_____________________________
0          16    16    32
1          16    16    96
2          16    16    160
...
5          16    48    96
…
255        .     .     .
```

Step2: Convert each 24-bit pixel of the input image into an 8-bit index value. Save the 8-bit color index values into `[InputFileName]-index.ppm` file. Assign the same index values to R, G, and B.

Step3: Save and display the 8-bit indexed-color image. To do this, you have to retrieve (or get) RGB values from the lookup table (LUT) using the index value of each pixel. Do not overwrite the original input in order to save your output. Save your output into a new image (named `[InputFileName]-QT8.ppm)`.

HW1 Sample Results

4. Extra Credit Task (30 pts)

Implement one of the following 24bit to 8 bit color quantization methods.

1) Non-Uniform Quantization: Modify the Uniform Color Quantization method to use a non-linear method when dividing each axis of the RGB cube. For example, break each axis on a logarithmic scale instead of linear scale.
2) Hybrid Approach with Error Diffusion: Apply the Error Diffusion method to the Uniform Color Quantization when quantizing each color component. For each of R, G, and B channels, quantize a 8bit value to a *n* bits (*n*=3 for R and G, *n*=2 for B) using the error diffusion.
3) Popularity Algorithm
4) Median-cut Algorithm

This task should

- Display the 8-bit color Look Up Table (LUT) generatedby the selected method.
- Convert each 24-bit pixel of the input image into an 8-bit index value. Save the 8-bit color index values into `[InputFileName]-index2.ppm` file. Refer to how to save 8-bit image in PPM format.
- Save and display the 8-bit indexed-color image. To do this, you have to retrieve (or get) RGB values from the lookup table (LUT) using the index value of each pixel. Do not overwrite the original input in order to save your output. Save your output into a new image (named `[InputFileName]-QT8-2.ppm).`

** If you did not implement this task, change the menu options accordingly.*