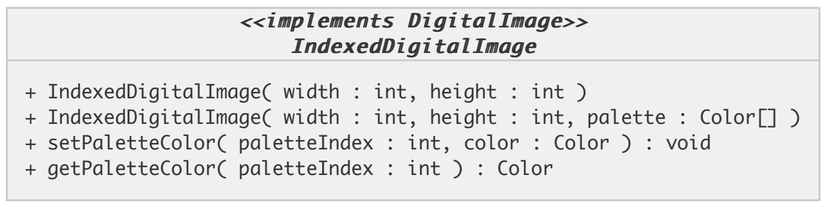
Color and Image Representation

Problems

1. ### YIQ Color Model
   a. Equation 3.4 in the text gives the transformation matrix for converting between the RGB and YIQ color spaces. The inverse of this matrix yields the transformation from YIQ to RGB. Give the transformation matrix for converting from YIQ to RGB.
   b. Convert the normalized RGB color <.25, 1, .75> to the YIQ color space.
   c. Convert the normalized YIQ color <.25, 1, .75> to the 8-bit (non normalized) RGB color space.

2. ### Color Metrics
   a. Give a formal proof that the maximum L_1 distance in HSB space is $(1 + 2\sqrt{2})$.
   b. Give a formal proof that the maximum L_2 distance in HSB space is $\sqrt{5}$.
   c. Give two colors C1 and C2 such that the L_1 distance between them in HSB space is maximal.
   d. Give two colors C1 and C2 such that the L_2 distance between them in HSB space is maximal.

3. ### Indexed Digital Image
   Implement the DigitalImage interface using a palette-based indexing technique. Your class must be named **IndexedDigitalImage** and conform to the UML class diagram shown below. Your implementation must have an 8bpp color depth.



4. ### DigitalImageIO
   Complete the DigitalImageIO class shown below. This class allows `DigitalImage`s to be loaded from and saved to files that follow the NETPBM file format. In particular, this problem requires that you support ASCII-encoded PPM files (magic number P3); you do not need to support binary-encoded files for this problem.

   The specific sub-class of `DigitalImage` returned by the read methood is controlled by the `type` parameter. The `type` parameter corresponds to one of the four DigitalImage implementations described in the textbook in addition to the `IndexedDigitalImage` solution to the previous homework question. Also note that if the image type is `INDEXED` then the loaded image *may* be an inexact approximation to the image described in the PPM file. *Make sure to follow the specification exactly.*

   - **MULTIDIM_ARRAY** : ArrayDigitalImage.java
   - **LINEAR_ARRAY** : LinearArrayDigitalImage.java
   - **PACKED** : PackedPixelImage.java
   - **INDEXED** : `IndexedDigitalImage.java`

```
public class DigitalImageIO {
  public enumeration ImageType { INDEXED, PACKED, LINEAR_ARRAY, MULTIDIM_ARRAY };
  public static DigitalImage read( File file, ImageType type) throws IOException, IllegalFileFormatException {...}
  public static void write( File file, DigitalImage image ) throws IOException {...}
}
```

5. ### ImageConverter
   Complete the ImageConverter class shown below. The `toBufferedImage` method accepts a `DigitalImage` object and produces an equivalent `BufferedImage`. The `toDigitalImage` method accepts a `BufferedImage` object and produces an equivalent `DigitalImage` object.

```
import java.awt.BufferedImage;
public class ImageConverter {
  public static BufferedImage toBufferedImage(DigitalImage src) {...}
  public static DigitalImage toDigitalImage(BufferedImage src) {...}
}
```

Additional Requirements

1. You must submit all your work using GitLab using a project named `cs454`. Place all of your code and documents into a folder named `hw1`. The written portions of this assignment must be included as a **PDF** document that is also uploaded to the GitLab repository. *NO OTHER DOCUMENT TYPE WILL BE ACCEPTED.*

Testing

1. I've written a test routine that you can apply to your code. The tester accepts a **png** or **jpg** image as input and uses your code to create and then read a ppm file. The test program uses command-line arguments; several examples of which are listed below.
   - java hw1.Tester man.png man1.ppm PACKED man1.png
   - java hw1.Tester man.png man2.ppm INDEXED man2.png
   - java hw1.Tester man.png man3.ppm LINEAR_ARRAY man3.png
   - java hw1.Tester man.png man4.ppm MULTIDIM_ARRAY man4.png
2. You can create and view **PPM** files using GIMP
3. If you are new to GitLab, please follow this GitLab tutorial.