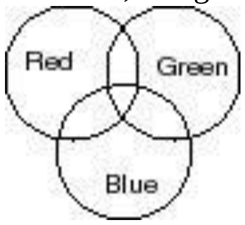
Problem 1 - Color wheels.

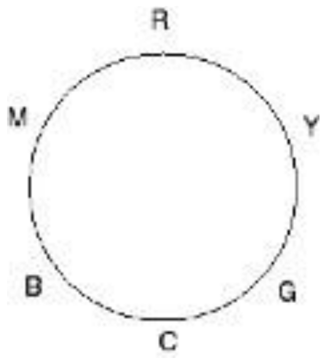
i. Create the color wheel we discussed in class, which looks like three circular dots of unit intensity. One is red, one green, one blue:

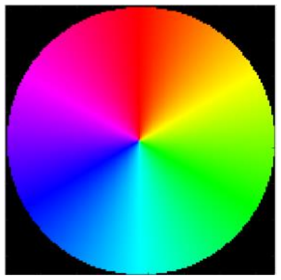


You can display this by creating a color structure out of a three-dimensional array such as a[256 256 3], and then displaying it using the command image(a). Verify that the mixed colors in the regions of overlap look the way you think they should.

ii. Now, modify the color values by letting the red, green, and blue intensities in turn each be 0.5 instead of 1 (note you will be creating three new images). Observe how the colors in the overlap regions change.

iii. Create a hue map as we presented on the board in class, showing all possible fully saturated colors as a circular disk (see figure below.) Hint: make the calculations in HSB space, and convert to RGB for display using the MATLAB command hsv2rgb.





Problem 2 - RGB images.

Download the data files lab10prob2r, lab10prob2g, and lab10prob2b, which are each 580 x 435 pixel images representing the red, green, and blue channels of a color picture. Combine the three channels into a 3-D matrix and display.

Assume that the entries in each channel range from 0-255. Stretch the data so that the image looks pleasing on your display, and submit the image.

Problem 3 - False color images.

Download the data files lab10prob3r, lab10prob3g, and lab10prob3b, which are now 1024 x 1024 pixel images representing the red, green, and blue channels of a color picture. Combine the three channels into a 3-D matrix and display.

i. Scale all three channels using a single transformation to make the total intensity of the image have a mean of 140 and a standard deviation of 60. Display and submit.

ii. Now, repeat (i) but this time scale each channel independently to have parameters 140, 60. Comment and illustrate how the independent scaling improves your ability to discriminate features in the image.

Problem 4 - Indexed color.

i. Open and display the single-channel (b+w) image in lab10prob4data. The image size is 580 x 435. ii. Using indexed color methods, display the image as shades of blue rather than black/white.

iii. Define a new color table such that the intensity from 0-255 varies the color smoothly according to the following definition:

Pixel value	0	100	200	255
Color	Black	Green	Yellow	White

Display the new image and submit. Here you should design your color table so that the colors vary from black to full-intensity green for pixels 0-100, from green to yellow from 100-200, and from yellow to white from 200-255.

Problem 5 – Saving our Eyes!

Recent vision science research has revealed that blue light emitted from laptops and cell phones can cause eye strain, mental fatigue, and can affect our circadian rhythms, preventing us from getting restful sleep. Indeed, it’s possible you’re working on this lab late at night, and the blue light is affecting your future sleep quality at his very moment! To mitigate this, one can download applications such as *f.lux* to reduce the amount of blue light in your computer’s display. In this question, we’ll explore how that works.

i. Read in an image of your choice and display it. “Filter” out the blue light and display the new image.

Ok easy, this is great! My eyes feel less strained already. Still, when we transition so suddenly from looking at a full color image to the blue-removed image, the blue-removed image might look very strange and displeasing. If we slowly transition between images, maybe our eyes will adjust and we won’t notice as much that the blue is missing. This is a good opportunity to practice stitching together image frames into a movie, which is what you’ll be doing for the final project.

ii. The VideoWriter class is one of a few MATLAB tools we can use to create a movie out of a sequence of image frames. Using a for loop, create 100 image frames that decrease the amount of blue in the image from full value in the first frame to no blue in the 100th frame. Then save those frames into an image with a frame rate of 25fps (so that in total, your movie will be four seconds long). Some starter code is included below to help you out. You can also refer to the last example in the MATLAB VideoWriter class documentation.

```
%Total number of frames in your movie numFrames
= 100;
 for k =
1:numFrames
    %TODO: create an image with less blue than before

    %TODO: display the image

    %use the function "getframe" to capture
    %this image as a frame structure and add it to a list, M
    M(k) = getframe;
end
v = VideoWriter('FluxFade','MPEG-4'); %Name your video
v.FrameRate = 25; %Set the frame rate to 25 frames per second
open(v) writeVideo(v,M); close(v);
```