In these labs you should work through the problems on the sheet and type your answers into a Microsoft Word document. When you have completed the tasks, print the document and hand it in.

Your answers can benefit from having small sections of R output and graphs copied and pasted into your word document. It can be useful to limit the width of output produced by R so that it fits into your document. You can change the width of output produced by R with a command like: `options(width=50)`.

The labs will count for 1 mark in the current assignment. They will be graded on a 0-1 basis. Don't forget to put your name and student ID on the document.

R has a variety of functions which allow you to control colour. These functions take numerical arguments and return character strings which are accepted as colour descriptions by R graphics functions.

1. The function `gray` takes values in the interval $[0, 1]$ and returns shades of gray.

   (a) You can generate a "ramp" of 11 grays ranging from black to white as follows:

   ```
   vals = seq(0, 1, length = 11)
   gray(vals)
   ```

   What do the `grays` values look like?

   (b) Plot the grays as follows:

   ```
   barplot(rep(1,11), col = gray(vals),
           space = 0, border = NA)
   ```

   You should see the effect of Mach banding in the image.

   (c) The gray's produced by the `gray` function are not perceptually evenly spaced. Try the command

   ```
   barplot(rep(1,11), col = gray(vals^0.65),
           space = 0, border = NA)
   ```

   and describe how the image has changed.

   (d) Using the `barplot` function is cheating. Write your own function to display colour ramps. The ramp will consist of a set of n rectangles, the ith one have its left side at $x = i - 1$, its right side at $x = i$, its base at $y = 0$ and its top at $y = 1$. You should be able to plot all the rectangles with a single call to `rect`. (You will also need to make calls to `plot.new` and `plot.window`.)

2. The function `rgb` takes three separate values in the interval $[0, 1]$ and interprets them as levels of red, green and blue primaries.

   (a) Make a colour ramp of 11 values by varying the value of the red primary from 0 to 1, holding the values for the green and blue at 0.

```
rgb(seq(0, 1 , length = 11), 0, 0)
```

Examine the resulting values.

(b) Plot your red colour ramp using the function you created in 1(d). if your function doesn't work, you can find one by following the R Code link from the class web page.

(c) Try creating a red–white, white–blue colour ramp.

3. The function hsv can be used to create a colour schemes by specifying colours in HSV coordinates.

   (a) Create a rainbow using `hsv` and plot it using your colour ramp function.

   ```
   n = 1001
   rainbow = hsv(seq(2/3, 0, length=n), 1, 1)
   ```

   (b) You may get a better rainbow if you apply a "gamma" correction to compensate for the nonlinearity of the monitor you are looking at. Try the following rainbow.

   ```
   n = 1001
   rainbow = hsv(seq(2/3, 0, length=n), 1, 1, gamma = 0.5)
   ```

4. The function `hcl` in the `colorspace` library can be used to obtain equal-impact colours.

   (a) Try creating and plotting an equal impact palette of colours. You can do this as follows.

   ```
   library(colorspace)
   rainbow = hcl((0:11/12)*360, 59, 75)
   ```

   (b) Try using a ramp of 1001 colours using `hcl` and see how this compares with the result of using `hsv`.