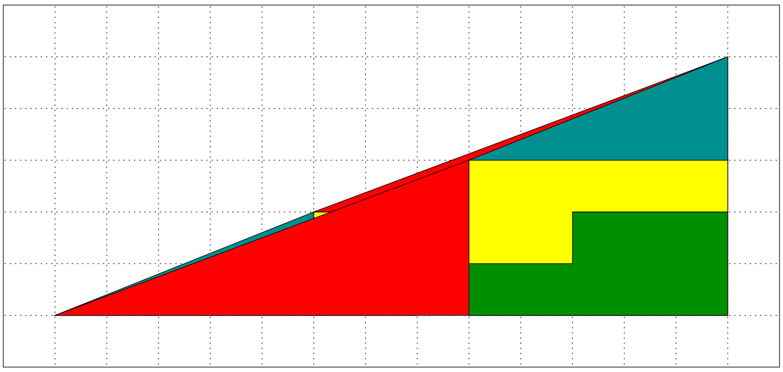
1. You can find larger (colour) images of this puzzle and the figure below in the class web pages.

   (a) When you look at the top figure in the question, it looks like the pieces fit together to form a triangle. This isn't the case. The slope of the hypotenuse of the black triangle is $2/5 = .4$, while the slope of the hypotenuse of the gray triangle is $3/8 = .375$. If the top and bottom figures are superimposed you can see there is a thin sliver of difference between the areas.



   The area of this sliver is precisely the area of the extra square.

   (b) Humans are very bad at making slope judgements. You are being fooled here because you cannot detect the slope difference in the triangles.

2.
```
#  For both cases we need a rectangular grid of squares
#  I'll take the grid to be n-by-n with n = 6, but it's
#  easy to replace this one value to get more smoothly
#  varying colours.
#
#  I generate the squares of the grid running left to
#  right and bottom to top.

n = 6
x = rep(1:n, n)
y = rep(1:n, rep(n, n))

#  For the first example, I generate equally spaced hues
#  around the colour wheel.  I avoiding wrapping the hues
#  by generating 7 values and dropping the last one.

h = rep(seq(1, 0, length = n+1)[1:n], n)
s = rep(seq(0, 1, length = n), rep(n, n))

#  The actual plot is just a single call to "rect".
#  I make a second call to draw a black boundary
#  around the box.

plot.new()
plot.window(xlim=c(0, n), ylim=c(0, n), asp = 1)
rect(x-1, y-1, x, y, col=hsv(h = h, s = s), border = NA)
rect(0, 0, n, n, border="black")

#  Here is the second example.  It is virtually identical.
#  The default hue in "hsv" is red, but I use blue.

v = rep(seq(0, 1, length = n), n)

plot.new()
plot.window(xlim=c(0, n), ylim=c(0, n), asp = 1)
rect(x-1, y-1, x, y, col=hsv(h = 2/3, s = s, v = v),
     border = NA)
rect(0, 0, n, n, border="black")
```