This exercise is intended to ensure that you grasp the concepts taught in Lectures 17 and 18. It should take you no more than a couple of hours to complete.

Desktop applications that allow elements to be coloured often provide the notion of a palette. In addition to assigning a particular RGB colour value to elements, the user can also assign symbolic colours, drawn from a separately constructed palette. This allows the user to maintain uniformity across a document, and sometimes to make general changes without having to adjust each element individually. For example, in Powerpoint, a palette is called a "slide color scheme", and has a fixed collection of symbolic colours, called "Background", "Text and lines", "Shadows", etc. If you change the colour associated with "Shadows", then all elements that were coloured as "Shadows" will be changed. In Keynote, the colour chooser has a tab that gives a collection of palettes, each with its own set of colours. The user can create new colours and new palettes, but can only add and delete colours in a palette. Since a colour can't be changed, it's not possible to update the colours of a document without modifying each element individually. Photoshop has a scheme similar to Keynote, but with a single collection of swatches that the user can add to. In short, there are many different applications with different treatments of colour palette.

1. Pick an application that has colour palette functionality, and construct an object model for it in the following stages:

   (a) Draw the diagram, and mark multiplicities.
   (b) Explain briefly the salient features.
   (c) List the key operations that are supported, and say which relations and sets they read or modify.

Optional exercise for the ambitious: instead of considering just one application, consider a pair of applications with similar color palette functionalities, and do this for both of them, using object modelling to highlight the differences between the two.

2. Explore the implementation of this scheme (or one of the schemes, if you did the optional exercise), in the following stages:

   (a) Draw two diagrams for two different, plausible ways in which the abstract object model might be represented in code.
   (b) Using the diagrams, compare the two representations and explain their advantages and disadvantages.

You don't need to consider how the palette is made persistent in the file system; just consider the state of the application while running.