STA314: Assignment 1, Fall 2020

This assignment is ~~Due 16 October at 12pm~~ **Due to the very late posting, the assignment has been extended to 11:59pm on Tuesday 3 November.**

Please submit your answers via Quercus as a *single* `.zip` file. This zip file should include:

- A .Rmd file that compiles to the vignette. (This will be compiled and the output marked)
- One (1) .R file called `functions.R` that contains all of the R functions you wrote. It should *also* contain the function `change_resolution` that is provided. (This will be marked)
- The picture needed to compile the vignette.
- A pdf of the *compiled* vignette. (This will be marked only if the .Rmd does not compile)

The marking scheme is as follows:

- 40% for a vignette that compiles *to a pdf* and demonstrates all 4 functions you are asked to write. (-10% for each missing function, **0% if the vignette does not compile**.)
- 20% for complete documentation (as described in the Week 4 tutorial) for all 4 functions. This should include a simple use case for each function. (5% per function)
- 40% for the vignette. It should clearly step through the process using non-technical language where appropriate. It should demonstrate the challenges with choosing the number of clusters (thread colours). All plots should have clear captions, axis names, and legends.

Please note: The raw markdown file will **not** be marked. If the RMarkdown compiles, the compiled pdf will be marked. In the event that the RMarkdown does not compile, the pdf will be marked. *If there is no pdf and the RMarkdown does not compile, you will loose 80% of your mark.*

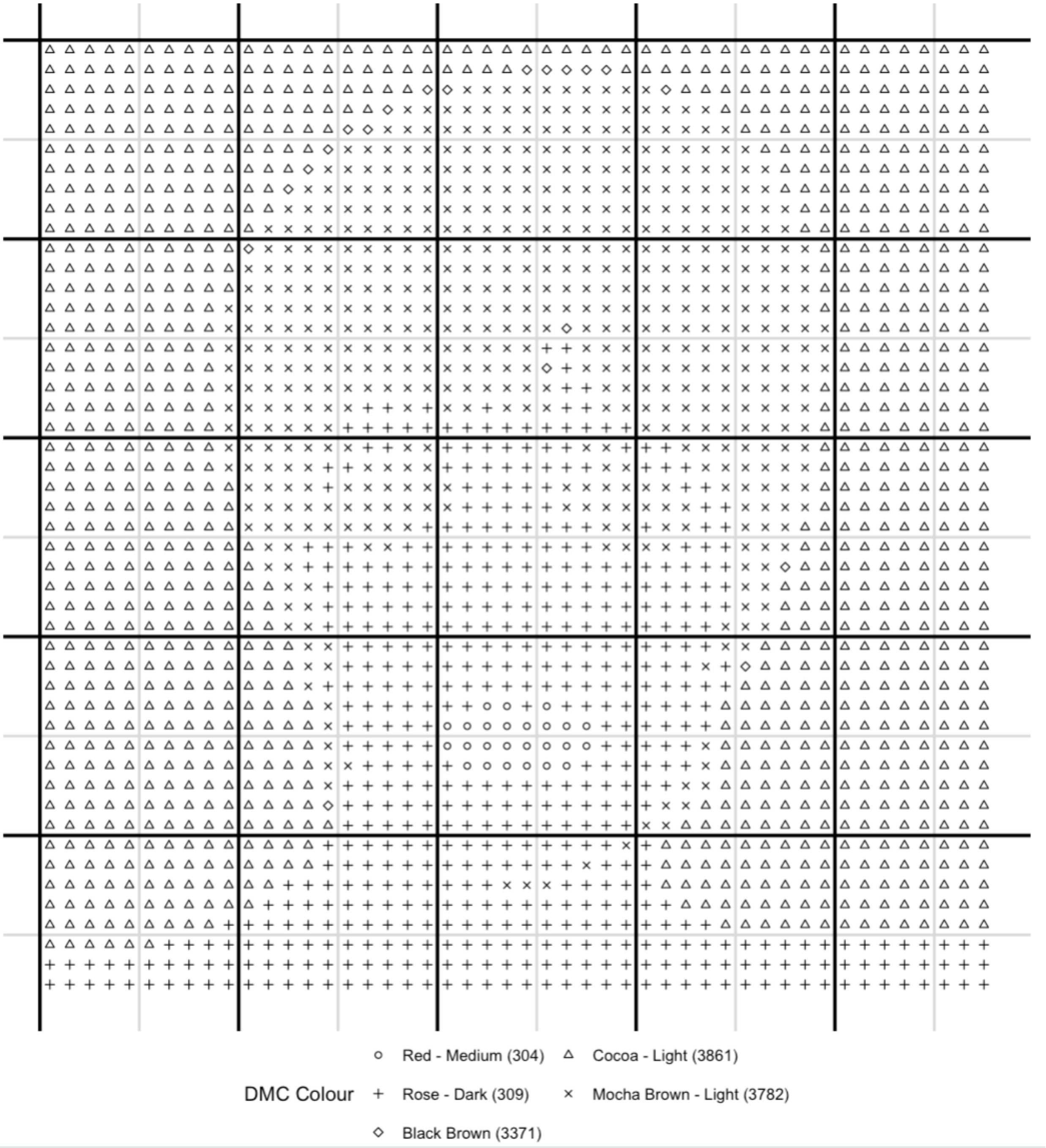
Please note: Please read the instructions *very carefully.* There is a restriction on the packages you are allowed to use in this assignment. A complete list is below, but you're restricted to `tidyverse`, `tidymodels`, `imager`, `dmc` (a package for choosing thread colour), `scales`, `cowplot`, and `sp` (which is used in the `change_resolution` function.)

Background: Making a cross-stitch pattern

Your task is to write several R functions that will take an image and create a cross-stitch pattern. This will include thread colour. There are a few challenges you need to get past:

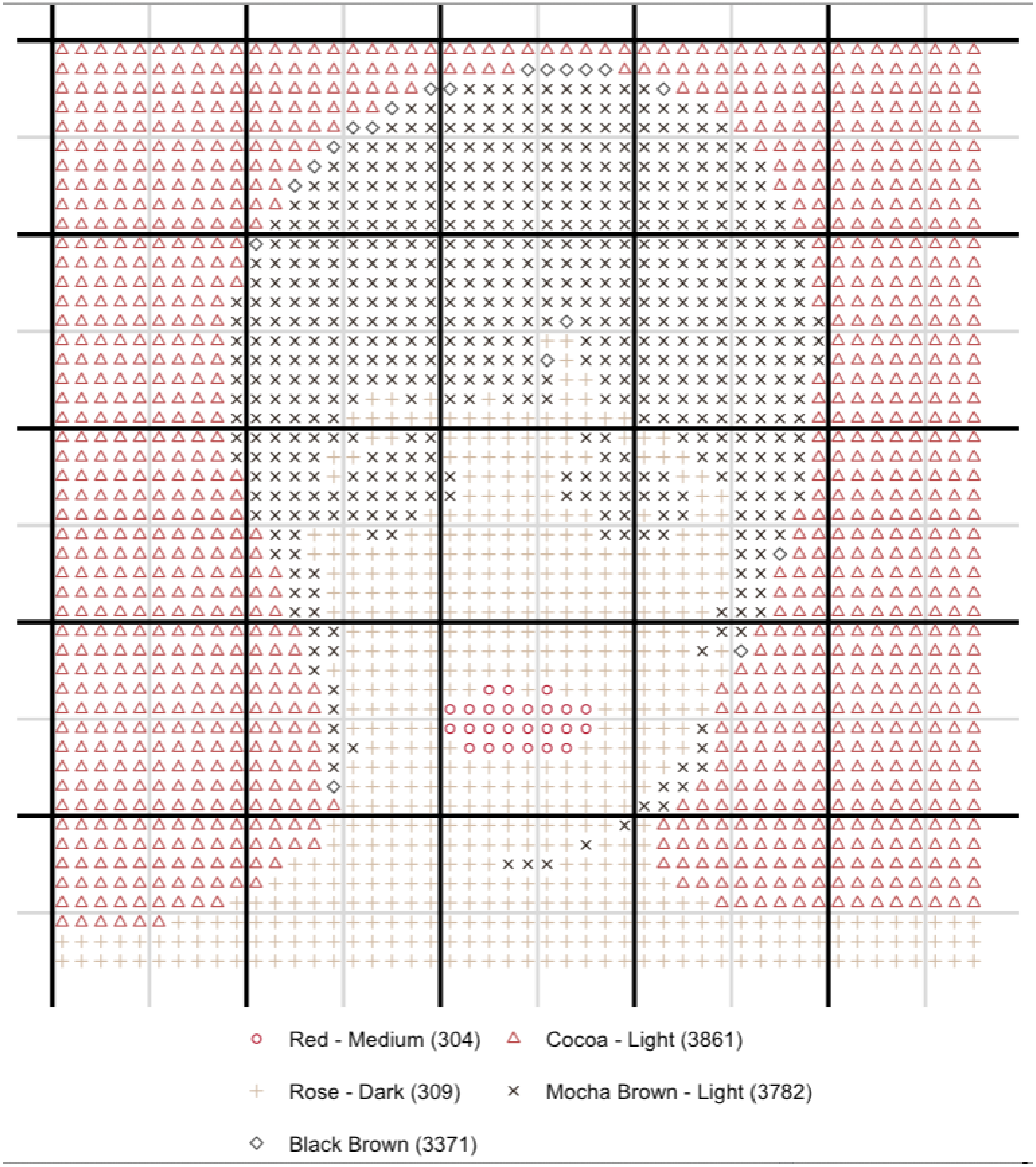
1. Most images have too many colours to conveniently cross-stitch. You will need to use clustering to simplify the image.
2. Not every colour has an associated embroidery floss colour. The `dmc` package by Sharla Gelfand can be found at (https://github.com/sharlagelfand/dmc)[https://github.com/sharlagelfand/dmc] and will be useful for finding the nearest thread to a given colour. This package is not perfect (it is not vectorized and it throws occasional warnings), but it works.
3. You will need to help the user select how many threads they will use.
4. Most images have too many pixels to make a realistic cross-stitch. Use the provided function `change_resolution` to change the resolution of an image in `(x, y, cluster)`. The lower resolution image will use the most common colour in the pixels that are being combined to get the aggregate colour. *Note*: Sometimes this function will drop a small cluster because it is never the most common cluster. That's not an error.
5. You need to make an *actual pattern* that people can use to do cross-stitch.
6. Some images have background colours that do not need to be covered in stitches. Your code should allow for one of the clusters to be "background" and those entries in the pattern should be empty.

Below is a *rough* example of a 47 x 47 Liza Minnelli cross-stitch (image from the week 2 tutorial). One is in colour and the other is in black and white. Both use symbols to differentiate colours and grid lines to help read the pattern. The legend has both the name of the colour and the colour number. *This is intentionally quite rough and should not be seen as the perfect example of a pattern!* In particular, this example does not make use of the backing colour! The clustering that was used to make this pattern had 6 clusters, but after reducting the resolution, one of these clusters did not appear in the final picture.



△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	◇	◇	◇	◇	◇	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	◇	◇	×	×	×	×	×	×	×	×	×	×	◇	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	◇	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	◇	◇	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	◇	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	◇	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	◇	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	◇	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	◇	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	+	+	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	◇	+	×	×	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	+	+	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	+	+	×	+	×	×	+	×	×	×	+	+	×	×	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	+	+	×	×	+	+	+	+	+	+	+	×	×	+	+	+	×	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	×	×	×	×	×	+	+	×	×	×	×	+	+	+	+	+	+	+	+	×	×	+	+	+	×	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	×	×	×	×	+	×	×	×	×	×	×	+	+	+	+	+	×	×	×	×	×	×	+	+	×	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	×	×	+	+	+	+	+	×	×	×	×	×	×	×	+	+	×	×	×	×	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	×	×	×	×	×	×	×	×	×	+	+	+	+	+	+	+	+	+	×	×	+	×	×	+	+	×	×	×	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	×	×	+	+	+	+	+	+	+	+	+	+	×	×	×	×	+	+	+	×	×	×	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	◇	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	×	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	+	◇	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	×	+	+	+	+	+	+	+	○	○	+	○	+	+	+	+	+	+	+	+	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	×	+	+	+	+	+	○	○	○	○	○	○	○	○	+	+	+	+	+	+	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	×	+	+	+	+	+	○	○	○	○	○	○	○	○	+	+	+	+	+	×	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	×	×	+	+	+	+	+	○	○	○	○	○	○	+	+	+	+	+	+	×	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	×	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	◇	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	×	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	×	+	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	△	△	+	+	+	+	+	+	+	+	+	+	+	+	+	×	+	+	+	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	△	+	+	+	+	+	+	+	+	+	+	+	×	×	×	+	+	+	+	+	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	△	△	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	△	△	△	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	△	△	△	△	△	△	△	△	△	△	△	△	△	△
△	△	△	△	△	△	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+

	○ Red - Medium (304)	△ Cocoa - Light (3861)
DMC Colour	+ Rose - Dark (309)	× Mocha Brown - Light (3782)
	◇ Black Brown (3371)	



Tasks

In this assignment, you are limited to using the following packages and base R

```
library(imager)
library(tidyverse)
library(tidymodels)
library(sp)
library(scales)
```

```r
library(cowplot)
#devtools::install_github("sharlagelfand/dmc")
library(dmc)
```

1. Produce **one** R file called `functions.R` that contains the following functions:

a. `cluster_info <- process_image(image_file_name, k_list)`:
   - Input:
     - `image_file_name` - a PNG or JPEG image.
     - `out` - the number of centres in the clustering
   - Output:
     - `cluster_info`: A list or tibble of information derived from the k_means that will be sufficient to be the input to any other function you write. This should be the *only* function that computes a clustering. This should include at least:
       - the original output of the `kclust` calls,
       - the tidied clusters, their associated RGB values and their nearest DMC thread colour information.

b. `scree_plot(cluster_info)`:
   - Produces and plots a scree plot.

c. `colour_strips(cluster_info)` or `color_strips(cluster_info)`:
   - Produces colour strips with the DMC colour closest to the cluster centre colour.

d. `make_pattern(cluster_info, k, x_size, black_white = FALSE, background_colour = NULL)`:
   - Plots the pattern.
   - Input:
     - `cluster_info` - The output of `process_image`
     - `k` - The chosen cluster size
     - `x_size` - The (approximate) total number of possible stitches in the horizontal direction
     - `black_white` - (logical) Print the pattern in black and white (TRUE) or colour (FALSE, default)
     - `background_colour` - The colour of the background, which should not be stitched in the pattern. (Default is to not have a colour)
   - Output:
     - This function should be the *only function* to use `change_resolution(image_df, x_size)`. It should produce a cross-stitch pattern that can be followed, complete with a legend that has thread colour, and a guide grid.

2. *All functions* should be correctly documented.
3. Produce a vignette that demonstrates *all the functions* you have written. It should step the reader through the entire process of producing a cross-stitch pattern from an image image you choose (see notes below). It should explain how to use each function, and interpret the output of each function. Special care should be shown to explaining how to use the scree plot and the colour strips to choose the number of clusters. This vignette should be self-contained and your target audience is a third year statistics student who has not done STA314 and is really into cross-stitch. This vignette **must** compile.

Note on the image in the vignette: Firstly, the image can not be obscene, objectionable, or otherwise problematic.

Beyond that you have two major constraints:

- The image should not be too high-resolution. The example I used was `1024 x 1024`, which was fine. But this is not a great assignment to experiment with high-resolution images.
- The image shouldn't be too exciting or have a very busy background. Not every picture will make a good cross-stitch. You won't be marked down for choosing a bad picture as long as the picture you choose is sufficient to explain the methods. (So, for an example, a picture that is only one or two colours wouldn't be good enough.) The picture doesn't need to be as simple as Liza, but it works best if there is just one main focus of the picture and not too much background.