LAB 1

Exercise 1

As I move my right eye towards the screen the black dot exits my field of vision. It mainly disappears once my nose physically touches the screen. A very similar phenomenon happens when I flip the image and redo the exercise with my left eye. As my nose reaches the screen I lose sight of the black dot.

Exercise 2

I chose the image that looks both like two people facing each other, or a vase in the middle.

In my sketch I used the “beginShape()” function and just set up a lot of curveVertex()’s to roughly draw out one side of the vase. I then copied them, but “width-“ in front of the x values and pasted it into a new “beginShape()”. This effectively mirrored the original face/side of vase.

I then to make them explicit, used the keyPressed() function so that after a key was pressed, it would toggle a boolean true and false. Then it ran through two if statements to highlight either the vases or the faces in yellow.

To be fair the sketch looks more like a silhouette of the Mad Hatter from Alice in Wonderland but it works.

I also changed the DOM in the sketch.js file so that when you highlight any of the two it says what it is too.

LAB 2

Exercise 1

In my sketch, no matter how fast it spins, I seem to perceive a reddish brown tint towards the more central line and a greenish-blue tint towards the outer edge.

An interesting thing that did happen though is that my screen started glitching and the spinning pattern remained on the screen somewhat. I had to restart. It was bizarre.

Exercise 2

In my code I made use of the p5 random() function’s ability to pick a random index from an array – so I just had an array “words = [“RED”, “GREEN” ,…] and used that as it seemed much more efficient than using if statements and ints. Therefore I did not need to replace any 3s with fours, just to

replace the contents of the words[] array. And yes it is difficult to read the words when the colours are different. And vice versa.

Exercise 3

I just did this in the same sketch as 2 as I was just returning values, so it wasn't going to impact the sketch at all. I wrote a modulo function that dealt with the problem of "-1 % 4" not working – so if you input a minus then it just adds the modulo to it until it becomes positive. Otherwise it just returns "input % mod"

The RGB conversion is just the same using the p5 PI, TAU constants, with the values eventually normalised back to degrees, or 0-100%

The HSB conversion normalises the HSB values to radians, 0-1, 0-1, does the conversion and then normalised the RGB to 0-255, rounded up to integers using ceil().

LAB 3

Exercise 1

I ended up with lines instead of a checkerboard but the effect was the same – the colours melded together to form a light orange. My guess was about 255, 175, 0, though it ended up being 255, 185, 0. I'm not really sure what causes this discrepancy – probably my own personal perception of colour.

Exercise 2

When I look from the inverted image to the greyscale image I can just vaguely see the uninverted colours on the greyscale image. I cycled through the pixels in a for loop and changed each one, using createGraphics to do different things to each image.