Quiz: Steganography Extraction and Duplication

Programming Exercise Guide

Before attempting this quiz you should write the following JavaScript programs.

Problem 1

You should modify the steganography program you wrote earlier in this module to add a function named `extract` to extract the image that is hiding in the lower half of the 8 bits (lower 4 bits) for each RGB part of each pixel. The function `extract` has one parameter `image`, and it returns a new image, which is the image that was hidden in the lower half of the bits for each RGB part of each pixel.

In particular you need to do the following:

1. Work an example with one pixel. Suppose the red value of a pixel is 203. Convert that number to binary. 203 is 128 + 64 + 8 + 2 + 1 = 11001011 in binary. You need to move the lower four bits, 1011, to the upper four bits and clear out the lower four bits, to 10110000. What decimal number is that? This should be the red value of this pixel for the hidden image. Try another number, say 86.
2. Write down the steps you took, find patterns and then come up with an algorithm to solve this problem.
3. Once you have an algorithm, plug in 203 and 86 to see if you get the correct new red value.
4. Then convert your algorithm to code.
5. Test your algorithm on the image below and also on images that you hide inside images.

Here is an example. Consider the two images of Owen and Robert which are cropped to be the same size.


Here we hide Robert inside of Owen's picture in the lower half of the bits, or 4 bits.

And here is Robert extracted from Owen's picture.

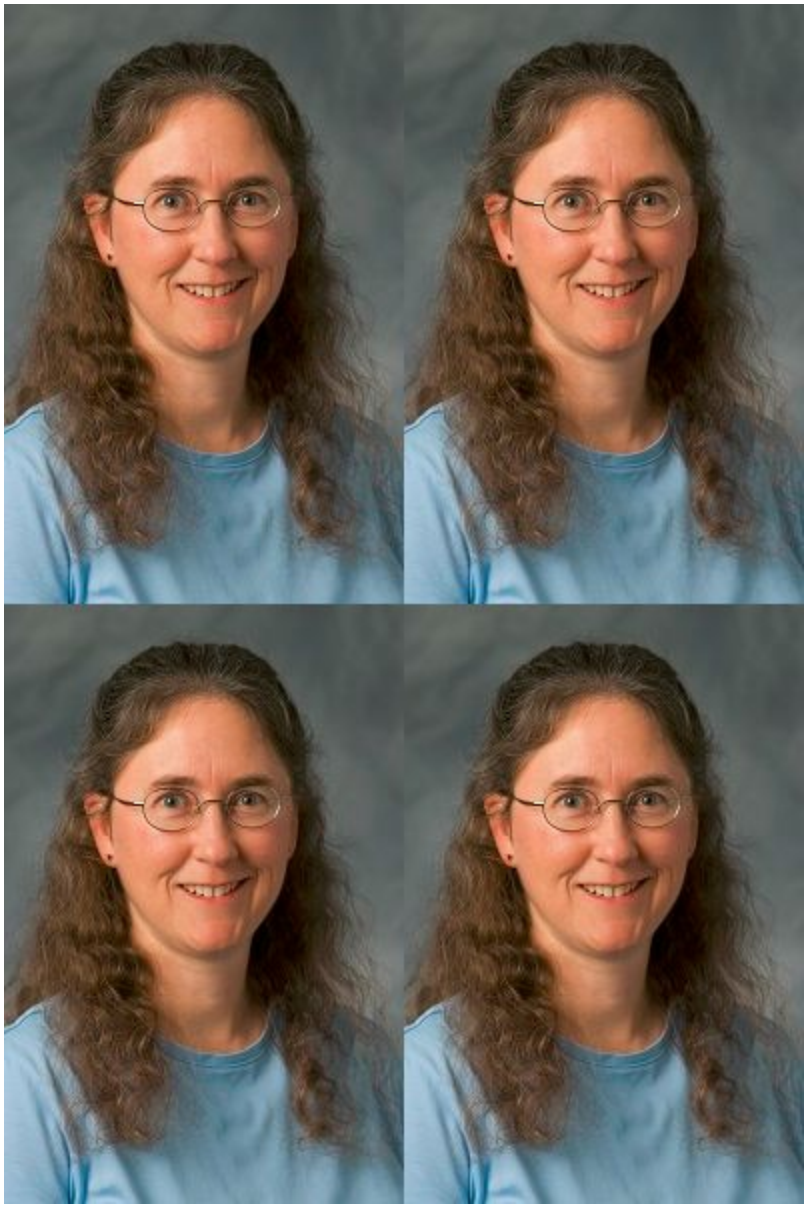
Notice there is a little bit of loss in Owen's picture and also in Robert's picture.

Problem 2

Write the function `duplicate`, which has one parameter `image` and returns a new image that is double the size of the original image and has four copies of the image.

For example, consider the image of Susan.

After running `duplicate` on her image the resulting image is:



To solve this problem here are some hints:

1. Work an example with one pixel. Start with a pixel and its (x,y) location from anywhere in the new duplicate image and figure out where that pixel should get its color from in the original image. Be sure to try an example pixel from each of the four pictures that appear in the duplicate image. Think about looping over the pixels in the new image and determine where each pixel should get its color from in the original image.

2. Write down the steps you took, find patterns, and then come up with an algorithm to solve this problem. Think about how this problem is similar to the `enlarge` function in one of the lessons.
3. Once you have an algorithm, plug in the location of some pixels to see if you get the correct answer.
4. Then convert your algorithm to code.
5. Test your algorithm on the image above and also on some of your own images.