Junior Knights - Course 2 Week 10 Challenge

Challenge A

We can hide a message in a picture as follows:

Remember that 0 = black and 255 = white. Thus, if we change a pixel value by 1 or 2, our intensity changes very, very little, almost imperctible to the human eye. Each number from 0 to 255 is represented in the computer with eight 0s and 1s. For example, the number 215 is stored in the following way in the computer:

11010111

If we store 208 in the computer instead of 215, it's stored as follows:

11010000

Notice that the last four bits (this is the term for 0s and 1s) are set to 0. It turns out that this gray scale setting (208), doesn't look so different than the gray scale setting 11010111. Thus, we could hide our OWN message in these last four 0s and 1s, embedded in a picture. We can hide one character in two pixel values. Here is an example:

Consider storing the pixel values 113 and 187. These are stored as

01110001 and 10111011, respectively.

Let's say we wanted to store the character 'Y' which has the internal value of 89. In binary, this is written as 01011001. Thus, we can store the first 4 bits in the first pixel and the second four bits in the second pixel:

0111<u>0101</u> and 1011<u>1001</u>

Here is how one can extract the data:

Let pix1 be the first picture value and pix2 be the next picture value. This formula is how to extra the one character encoded in the two pixels:

```
char decode = (char)(16*(pix1%16) + pix2%16);
```

If the message is shorter than the picture, then the decoded characters will "garbage" values of some sort.

Task: The picture frog_secret.pgm stores a secret message in this manner. Try to decode it!!!

Challenge B

The file frog.pgm has been encrypted using the affine cipher. Thus, for each pixel value x in between 0 and 255, it has been mapped to the value (ax + b)%255. When doing this, the image obtained is frog-encoded.pgm.

Using the same keys a and b, we encrypted another picture and stored the result in junk.pgm.

Your job is to first figure out the correct decrypting keys. These are the keys, that when applied to frog-encoded.pgm give you the image frog.pgm.

Once you figure these keys out, apply the same transformation to junk.pgm to reveal the secret image.

Hint: Rather than using ALL of the picture data, you can probably get away with using just two pixels worth of comparative data.

Good luck!