Picture Lab Bonus 2

<< [Picture Lab Bonus](https://mathorama.com/apcs/pmwiki.php?n=Main.PictureLabBonus) | [Picture Lab](https://mathorama.com/apcs/pmwiki.php?n=Main.PictureLab) | [Picture Lab Activity 1](https://mathorama.com/apcs/pmwiki.php?n=Main.PictureLabActivity1) >>

Steganography

Steganography is the science of hiding information in a picture. You can hide a black and white message inside a color picture by first changing all the red values in the original color picture to be an even value (by subtracting one if odd). Make a picture of the same size out of the message that will be hidden. Then loop through both the original picture and the message picture, setting the red value of a pixel in the original picture to odd (by adding one to it) if the corresponding pixel in the message picture is close to the color black. Write an encode method that takes the black and white picture message and changes the current picture to hide the message picture inside of it. Then also write a decode method that returns the picture hidden in the current picture.

Attach:SteganographyFigure.png

Method to add to your PictureTester class

/\*\* Method to test encode and decode \*/

public static void testEncodeAndDecode()

{

Picture beach = new Picture("beach.jpg");

beach.explore();

Picture message = new Picture("msg.jpg");

beach.encode(message);

beach.explore();

Picture decoded = beach.decode();

decoded.explore();

}

The encode Method to complete in your Picture class

The encode method should go to every pixel, and do two things:

1. if the pixels's red value is odd, make it even by subtracting 1.
2. if the messagePict pixel is close enough to black (remember the colorDistance method we used in Activity 9?), then set the pixel's red value to an odd number by adding one.

/\*\* Hide a black and white message in the current

\* picture by changing the red to even and then

\* setting it to odd if the message pixel is black

\* @param messagePict the picture with a message

\*/

public void encode(Picture messagePict)

{

Pixel[][] messagePixels = messagePict.getPixels2D();

Pixel[][] currPixels = this.getPixels2D();

Pixel currPixel = null;

Pixel messagePixel = null;

/\*\* Your code here \*/

}

The decode Method to complete in your Picture class

The decode method will create and return a new Picture messagePicture that should reveal in black all the odd red values that were encoded in the encode method above.

If the currPixel has an odd red value, then the corresponding pixel on the newly made messagePicture should be set to black, and white otherwise (remember the setColor method from Activity 9?).

/\*\*

\* Method to decode a message hidden in the

\* red value of the current picture

\* @return the picture with the hidden message

\*/

public Picture decode()

{

Pixel[][] pixels = this.getPixels2D();

int height = this.getHeight();

int width = this.getWidth();

Pixel currPixel = null;

Pixel messagePixel = null;

Picture messagePicture = new Picture(height,width);

Pixel[][] messagePixels = messagePicture.getPixels2D();

/\*\* your code here \*/

return messagePicture;

}