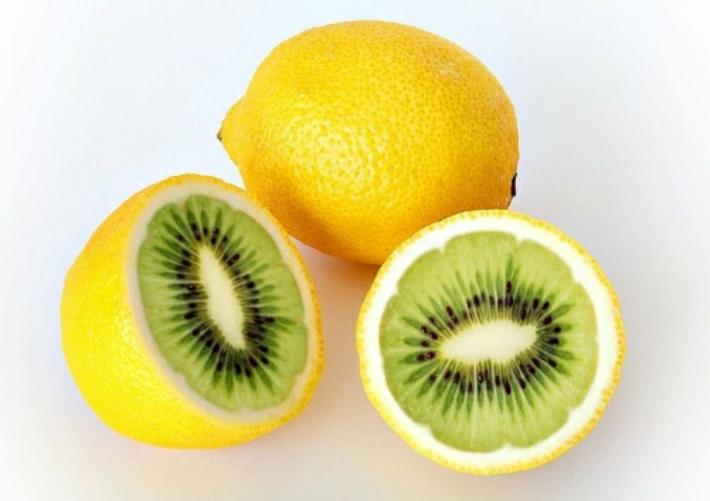
PLTW COMPUTER SCIENCE

Activity 1.4.3

Arrays and Images

Introduction



Have you ever played with special photo effects on a computer? Now that you know that images are really just zeros and ones, you might wonder: how are those special effects created?

Even very routine computer work involves constant manipulation of images. Every time you move a window on the screen, click on a menu, or even just move the mouse, the pixels on the screen change. What algorithms are used to determine the zeros and ones for the graphics card to send to the monitor?

Ready, set, JPG!

Resources

1.4.3 sourceFiles.zip

Reference Card for Pyplot and PIL

Procedure

Part II: Manipulating Pixels

5. You can assign new values to a pixel. Add the following lines of code before the `fig.show()` in your code. Execute the code and examine the figure it creates.

```
###
# Make a rectangle of pixels yellow
###

height = len(img)
width = len(img[0])
for row in range(200, 220):
     for column in range(50, 100):
          img[row][column] = [255,255,0] # red + green = yellow
```

The nested `for` loops iterate through a rectangle of pixels. The outer loop runs through each row, from row 200 through row 219. For each iteration of that outer loop, the inner loop works across the image from column 50 to column 99. A new list of RGB values is assigned to every

one of these pixels.

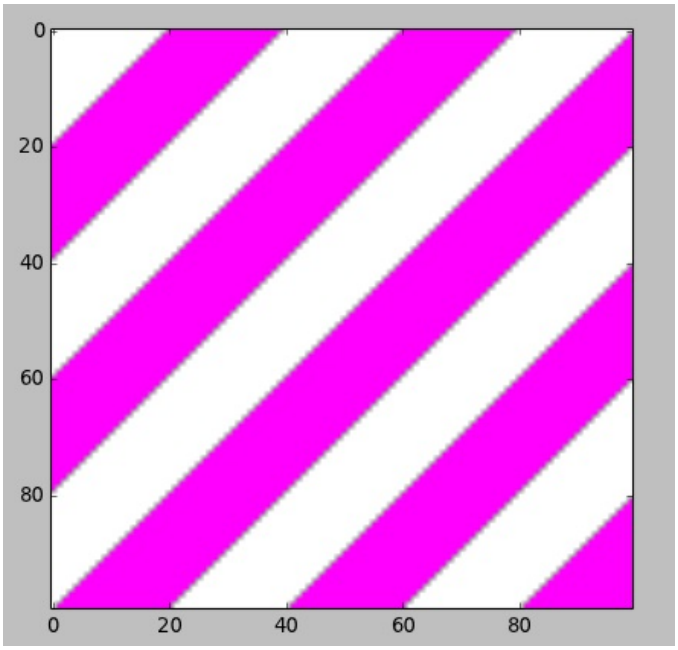
Change the code to create a green rectangle that covers the woman's earring.

6. The assignment in line 30 from the previous step can be placed within an `if` structure so that not all pixels get the new assignment. Try replacing the code for a colored rectangle from the previous step with lines 22-31 of the following code. The conditional in the new line 30 uses the built-in function `sum()` to add together the values of the three RGB pixels. If red + green + blue is more than 500, the pixel was bright and in this image was probably the sky.

```
img = plt.imread(filename)

###
# Change a region if condition is True
###

height = len(img)
width = len(img[0])
for r in range(155):
    for c in range(width):
        if sum(img[r][c])>500: # brightness R+G+B goes up to 3*255=765
            img[r][c]=[255,0,255] # R + B = magenta

###
# Show the image data
###
```

   - Explain the algorithm used by lines 28-31 of this code.
   - Add another section of code to change the color of the woman's earring.
   - Save your *Python* file in the code editor.

7. In this step you will create an algorithm that creates a two-color image. One RGBA color will be transparent (alpha=0) and the other RGBA color will be opaque (alpha=255.) Experiment with the program `make_mask.py` from this Activity's source code as an example to start with. The code provided creates the following image. Modify the code to create an `ndarray` representing an RGBA pattern of your own design. Your array should have three dimensions: [rows][columns][RGBA].



Conclusion

1. Describe what the data in a digital image contains, and describe what it means if a digital image has been "altered."
2. What are some of the differences between a photograph taken with light-sensitive film and a photograph taken with a digital camera? In what ways are they the same?
3. There are ways to send secret information in photographs using the lowest-place-value bits in each color byte. Concealing information in an image is called **steganography**. The 1s place and 2s place of each RGB pixel intensity could be changed to encode the numbers 0 to 63, more than enough for the alphabet.

    - Explain why these 6 bits are of least significance in the image representation.
    - Explain why 6 bits are enough to encode 0 to 63.
    - How would this make the image look different?

4. If you google "Python image analysis," you can find several *Python* libraries that will analyze an image. At least one library has a method or function that can determine how many separate objects are in the image. In very rough terms, describe how you think such an algorithm might acquire this information from the RGB pixel values.