Editing Images

Birmingham-Southern College
Anthony Winchester

But first...nested loops

Nested Loop

* Any loop inside another loop is called a nested loop

* The behavior is the same as any other loop, but when the inner loop finishes, we return to the outer loop and depending on conditions, may need to repeat the inner loop from start to end

Let's See What This Does

```
for i in range(4):
    for j in range(2):
        print('i:', i, '\nj:', j, '\ni + j:', i+j)
    print('Inner loop done, but outer loop isn't.')
print('Now we're done!')
```

Back to Images...

What is an Image

* An image is truly a table of pixels where each pixel is a mixture of red, green, and blue

* We can manipulate images programmatically by altering the red, green, or blue values (or all of them) within each pixel

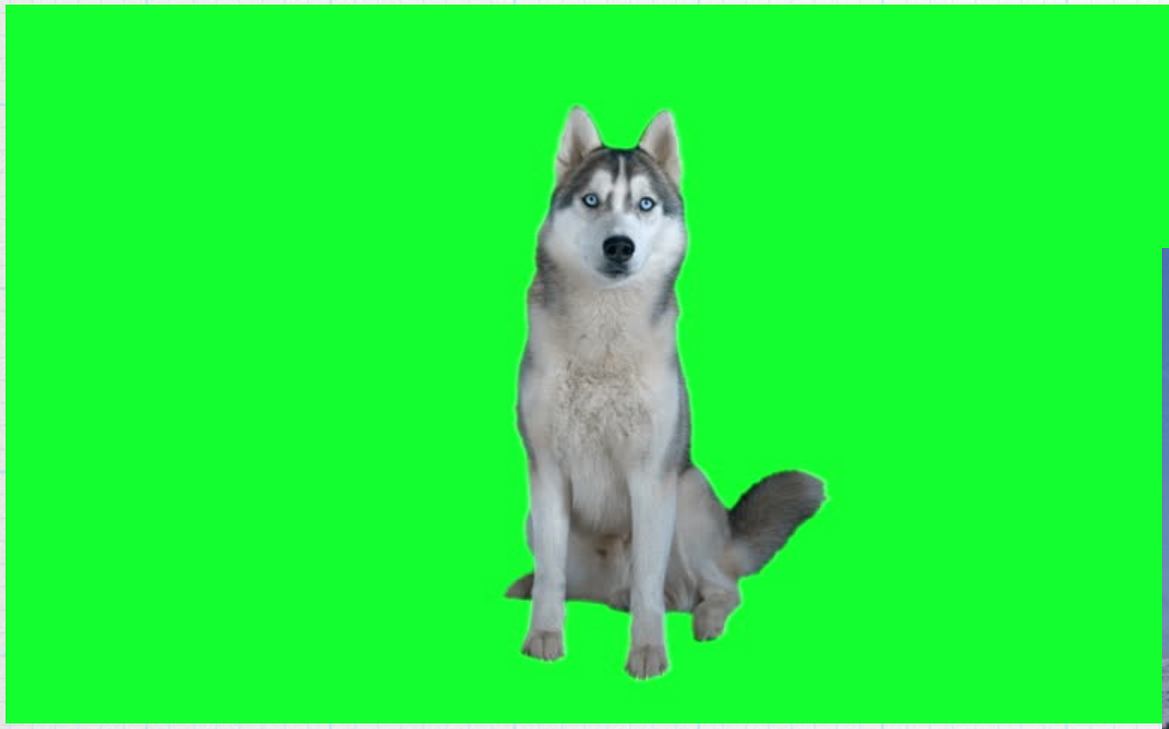
rgb	rgb	rgb	rgb
rgb	rgb	rgb	rgb
rgb	rgb	rgb	rgb
rgb	rgb	rgb	rgb
rgb	rgb	rgb	rgb

What can we do?

Simulate a Sunset

Go From Color to Grayscale

Green Screen!



How do we do this?

Image Editing Code

```
#Import the Image library that will allow us to access images
from PIL import Image

#Open an image from a file
img = Image.open('tundra.jpg')

#Get all of the pixels (stored in a two dimensional array - matrix)
pixels = img.load()
```

```
#Loop through all of the pixels
for i in range(img.size[0]): #Loops through columns

    for j in range(img.size[1]): #Loops through rows
        #Get the rgb value of the pixel at (i,j)
        r,g,b = img.getpixel((i,j))

        #Set the red of pixel (i,j) to 0 but leave the green and blue alone
        pixels[i,j] = (0, g, b)

#Save the image in a new file
img.save('tundra2.jpg')
```

Image Editing Code

* The code on the previous page changed the red value of the pixel to 0 for every pixel in the image

* How would we create grayscale?
* How would we simulate a sunset?
* How would we do a green screen?
* How would we eliminate red eye?

Grayscale

* Using the same loop as in the example, we would only change one line...the line that changed red to 0

* For Grayscale, you calculate the average value for each pixel and then replace the red, green, and blue values with that average

Grayscale Code

```
#Import the Image library that will allow us to access images
from PIL import Image

#Open an image from a file
img = Image.open('tundra.jpg')

#Get all of the pixels (stored in a two dimensional array - matrix)
pixels = img.load()

#Loop through all of the pixels
for i in range(img.size[0]): #Loops through columns

    for j in range(img.size[1]): #Loops through rows
        #Get the rgb value of the pixel at (i,j)
        r,g,b = img.getpixel((i,j))

        #Calculate average intensity of pixel, value must be an int
        intensity = int((r + g + b)/3)

        #Set the red, green, and blue values to the average intensity
        pixels[i,j] = (intensity, intensity, intensity)

#Save the image in a new file
img.save('tundraGrayscale.jpg')
```

Simulate a Sunset

* To simulate a sunset, you reduce the blue and green values by 30% but leave red alone

Creating a Negative

* To create a negative, we need the opposite color
  * Replace the red with 255 - red
  * Replace green with 255 - green
  * Replace blue with 255 - blue

You Try

* Test out the given code, removing all red
* Try creating a grayscale image
* Try creating a negative image

Other Ideas to Try

Green Screen

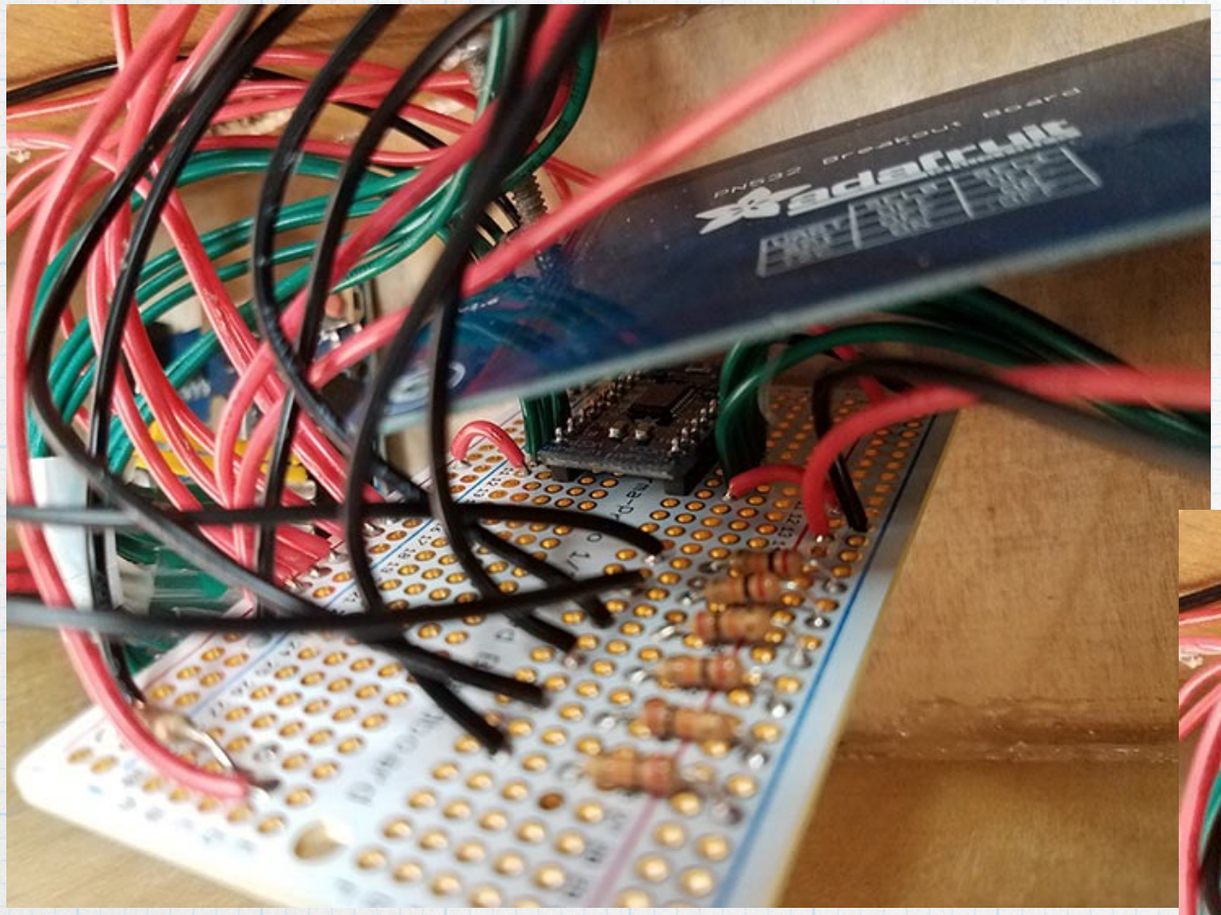
* You have to load two images for green screens (must be the same size)

* You look at each pixel in the image with the green screen, if the green in that pixel is greater than 245, you replace it with the same pixel of the other image

Green Screen

* Load both images into memory
* Loop through the image with the green screen, get the r, g, b for each pixel in the green screen image
* Get the r, g, b for each pixel in the background image
* If the green in the pixel from the green screen image is greater than 245 (example), replace the r, g, b of that pixel with the r, g, b of the background image pixel
*

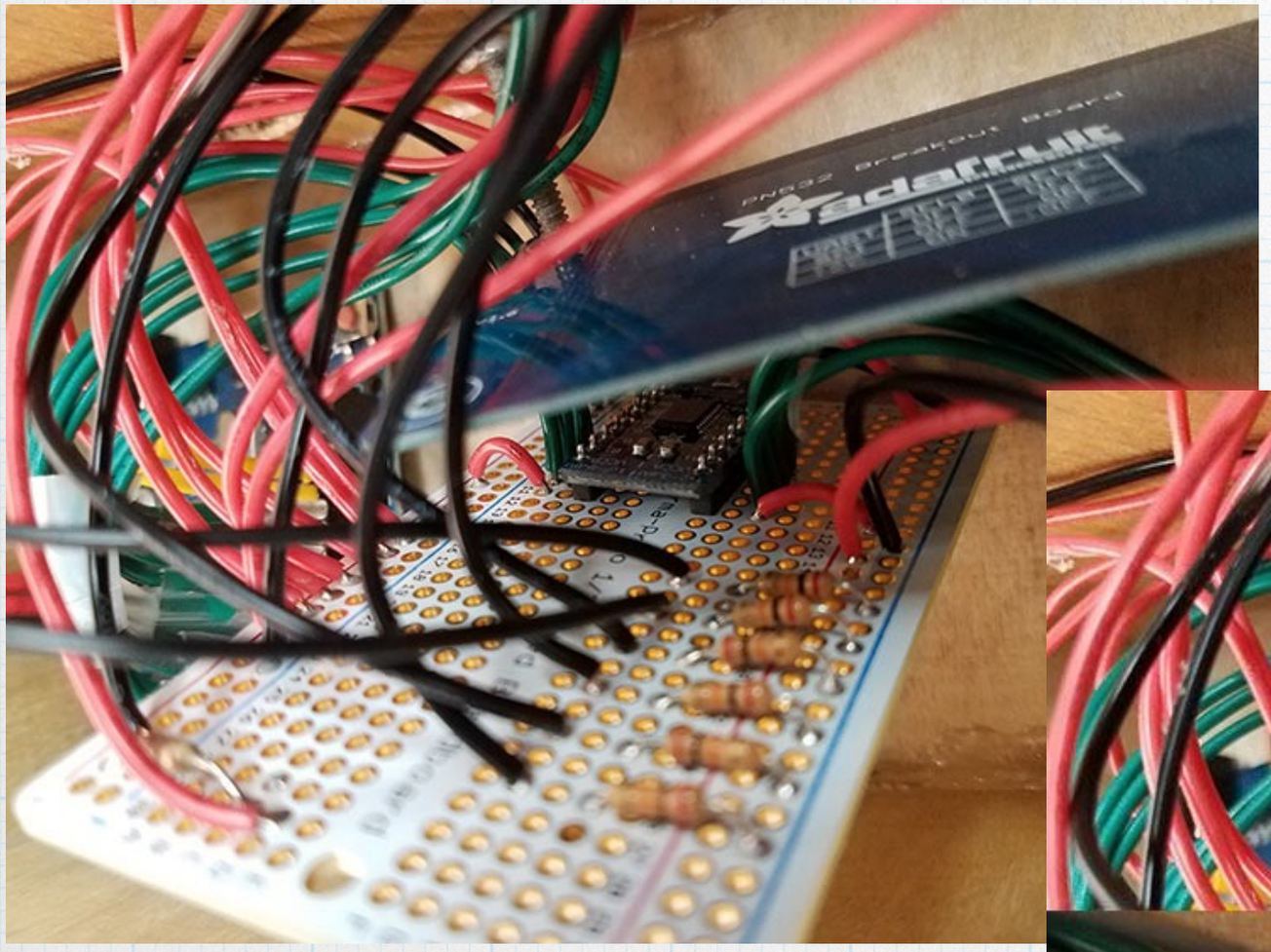
Mirroring an Image



Mirroring an Image

* You'll need a target variable that stores the halfway point in the rows
* Loop through all of the columns (horizontal), but only loop through the first half of the rows (vertical)
* Get the r, g, b of each pixel and store it in the target pixel
* You then have to increment the target position

Mirror and Flip



Mirror and Flip

* You'll need a target variable that stores the last pixel (bottom-right)
* Loop through all of the columns (horizontal), but only loop through the first half of the rows (vertical)
* Get the r, g, b of each pixel and store it in the target pixel
* You then have to decrement the target position

Sepia

Sepia

* When looping through each pixel, convert it to grayscale first
* Then, for each pixel, do the following:
    * Tint the shadows: if red is less than 63, change the red to red * 1.10 and blue to blue * 0.9
    * Tint the midtones: if the red is greater than 62 and less than 192, change the red to red * 1.15 and blue to blue * 0.85
    * Tint the highlights: if red is greater than 191, change red to red * 1.08 and blue to blue * 0.93. Also, if red is greater than 255, change the red to 255 (we can't have a value higher than 255).

Final Project

* Use these ideas to create a photo portfolio of ten images that are manipulated by code. A theme would be helpful. You can add removal of specific colors, such as red eye removal.

* There is more to this project, check Moodle for the full write-up.

But wait...we were talking about functions

Try it out...

* Let's restructure our program so that our editing techniques are each within their own function