CSE 107: Lab 02: Simple Image Manipulations in Python.

Valentinno Cruz

LAB: Thu 4:30-7:20

Yuxin Tian

September 26, 2022

Task 1: Computing the maximum value of an image. Rotating an image.

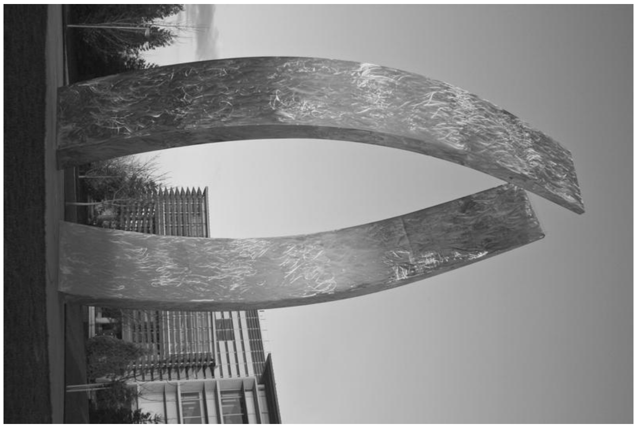
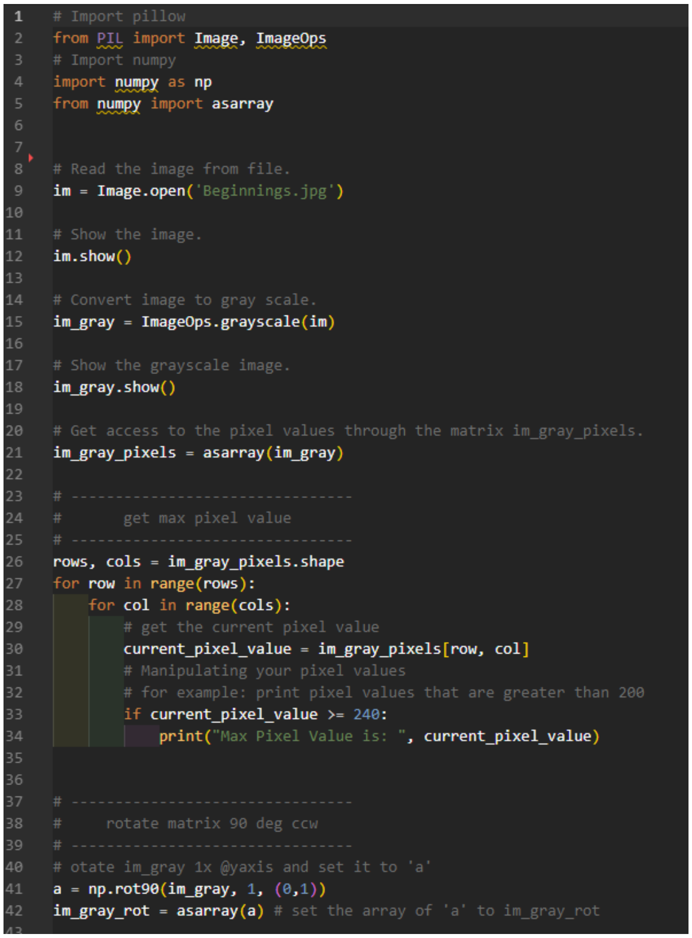


Figure 1. The beginnings image rotated 90 degrees Clockwise

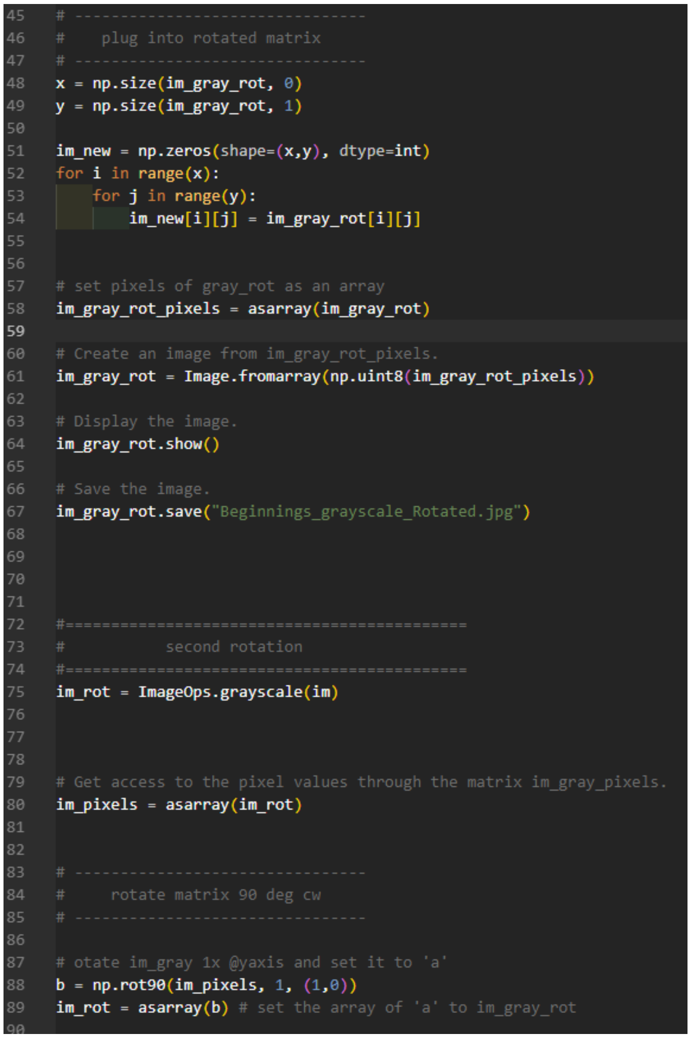
Questions for task 1:

1. What is the maximum pixel value of your grayscale Beginnings image?
   a. **240**
2. What is the maximum pixel value of your clockwise rotated grayscale image?
   a. **240**
3. Should these be the same? Why or why not?
   a. **They should be the same since it is the same amount of pixels**
4. What was the most difficult part of this task?
   a. **Figuring out how to do the tasks using the nested for loops, I was still unable to get the max pixel effectively.**

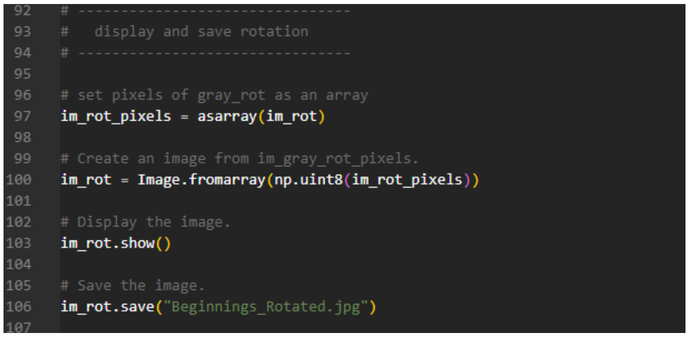
Task1.py



```python
# Import pillow
from PIL import Image, ImageOps
# Import numpy
import numpy as np
from numpy import asarray


# Read the image from file.
im = Image.open('Beginnings.jpg')

# Show the image.
im.show()

# Convert image to gray scale.
im_gray = ImageOps.grayscale(im)

# Show the grayscale image.
im_gray.show()

# Get access to the pixel values through the matrix im_gray_pixels.
im_gray_pixels = asarray(im_gray)

# --------------------------------
#        get max pixel value
# --------------------------------
rows, cols = im_gray_pixels.shape
for row in range(rows):
    for col in range(cols):
        # get the current pixel value
        current_pixel_value = im_gray_pixels[row, col]
        # Manipulating your pixel values
        # for example: print pixel values that are greater than 200
        if current_pixel_value >= 240:
            print("Max Pixel Value is: ", current_pixel_value)


# --------------------------------
#      rotate matrix 90 deg ccw
# --------------------------------
# otate im_gray 1x @yaxis and set it to 'a'
a = np.rot90(im_gray, 1, (0,1))
im_gray_rot = asarray(a) # set the array of 'a' to im_gray_rot
```



```python
# --------------------------------
#    plug into rotated matrix
# --------------------------------
x = np.size(im_gray_rot, 0)
y = np.size(im_gray_rot, 1)

im_new = np.zeros(shape=(x,y), dtype=int)
for i in range(x):
    for j in range(y):
        im_new[i][j] = im_gray_rot[i][j]


# set pixels of gray_rot as an array
im_gray_rot_pixels = asarray(im_gray_rot)

# Create an image from im_gray_rot_pixels.
im_gray_rot = Image.fromarray(np.uint8(im_gray_rot_pixels))

# Display the image.
im_gray_rot.show()

# Save the image.
im_gray_rot.save("Beginnings_grayscale_Rotated.jpg")




#============================================
#            second rotation
#============================================
im_rot = ImageOps.grayscale(im)



# Get access to the pixel values through the matrix im_gray_pixels.
im_pixels = asarray(im_rot)


# --------------------------------
#     rotate matrix 90 deg cw
# --------------------------------

# otate im_gray 1x @yaxis and set it to 'a'
b = np.rot90(im_pixels, 1, (1,0))
im_rot = asarray(b) # set the array of 'a' to im_gray_rot
```



```python
# --------------------------------
#   display and save rotation
# --------------------------------

# set pixels of gray_rot as an array
im_rot_pixels = asarray(im_rot)

# Create an image from im_gray_rot_pixels.
im_rot = Image.fromarray(np.uint8(im_rot_pixels))

# Display the image.
im_rot.show()

# Save the image.
im_rot.save("Beginnings_Rotated.jpg")
```

Task 2: The inverse of the Watertower image.

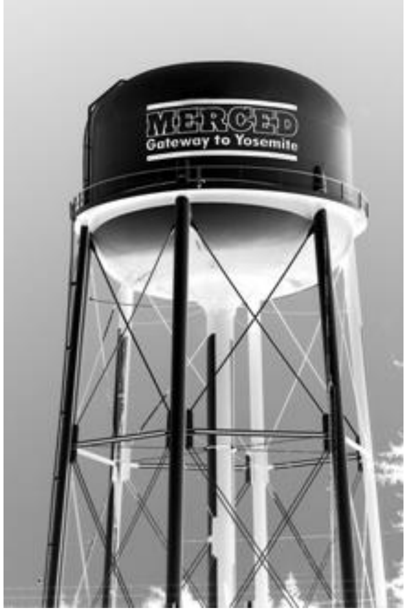
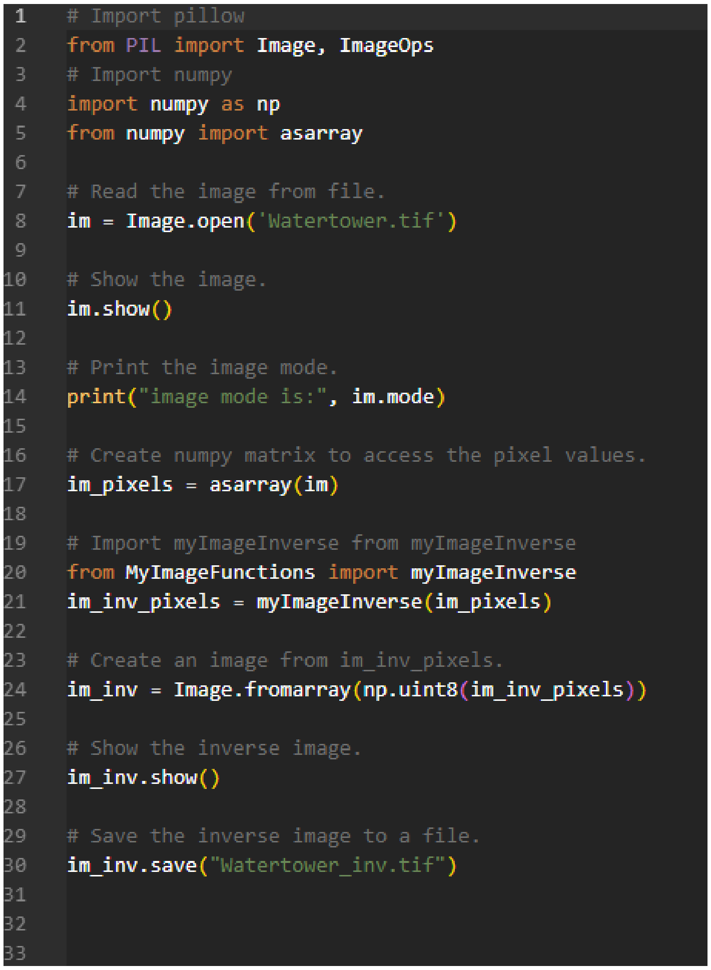


Figure 2. The inverse of the Watertower Image

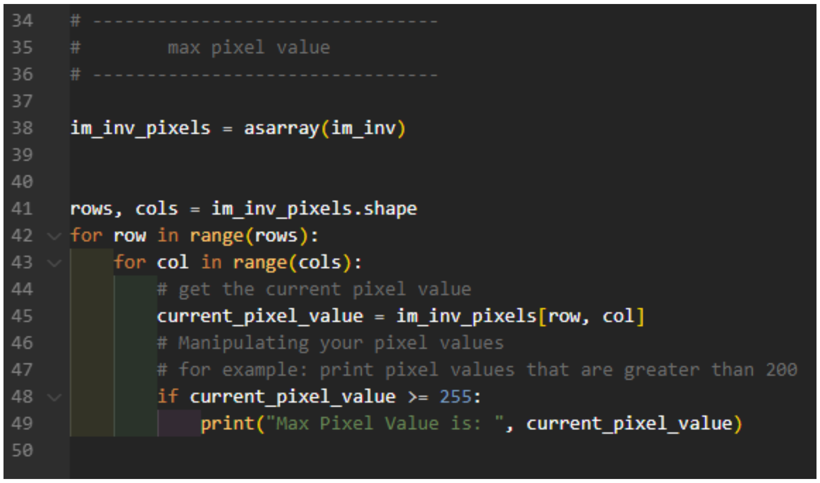
Questions for task 2:

1. What is the maximum pixel value of your inverse image?
   a. **255**
2. How is this maximum value related to the values of the original image?
   a. **255**
3. What was the most difficult part of this task?
   a. **Figuring out how to get the function to work properly**

Task2.py

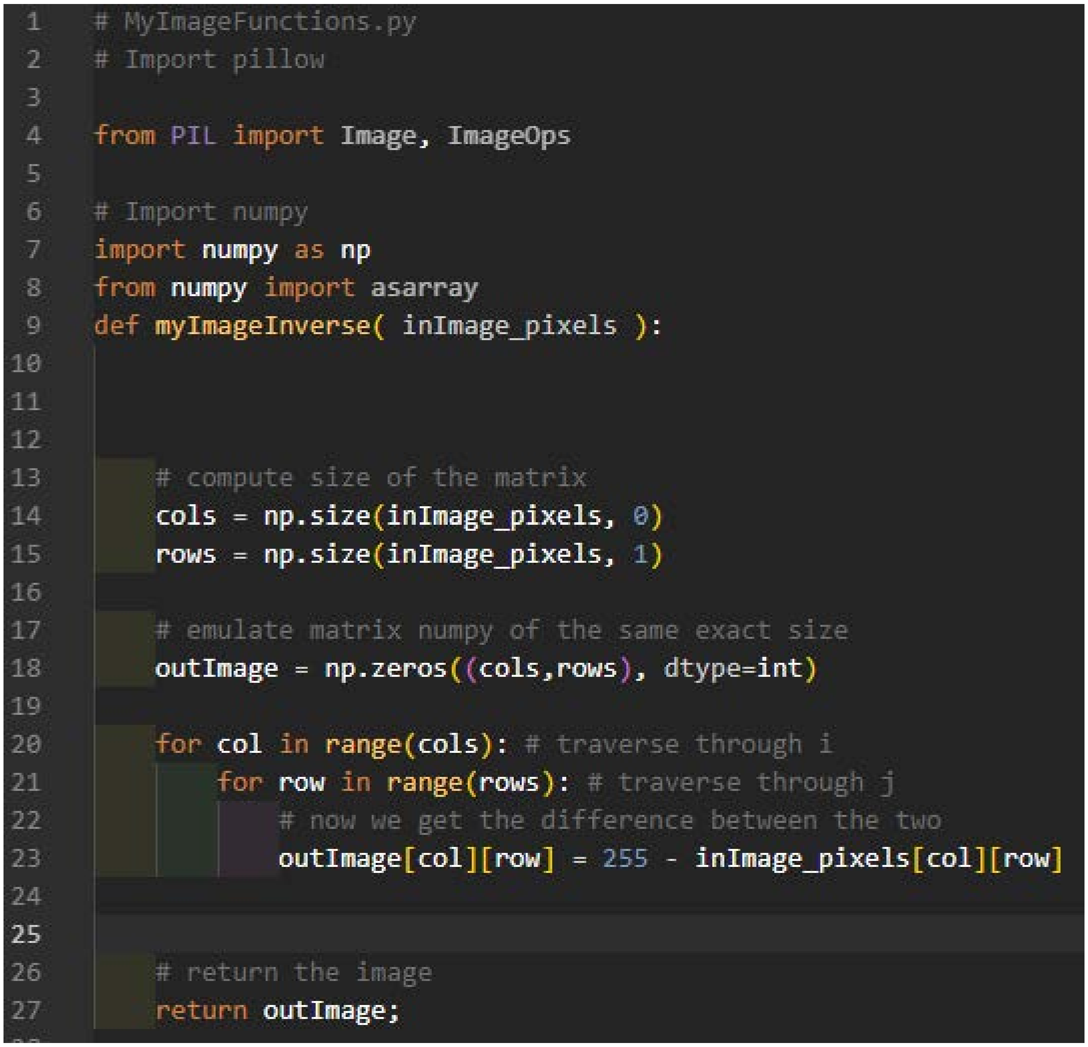


```python
# Import pillow
from PIL import Image, ImageOps
# Import numpy
import numpy as np
from numpy import asarray

# Read the image from file.
im = Image.open('Watertower.tif')

# Show the image.
im.show()

# Print the image mode.
print("image mode is:", im.mode)

# Create numpy matrix to access the pixel values.
im_pixels = asarray(im)

# Import myImageInverse from myImageInverse
from MyImageFunctions import myImageInverse
im_inv_pixels = myImageInverse(im_pixels)

# Create an image from im_inv_pixels.
im_inv = Image.fromarray(np.uint8(im_inv_pixels))

# Show the inverse image.
im_inv.show()

# Save the inverse image to a file.
im_inv.save("Watertower_inv.tif")
```



```python
# --------------------------------
#        max pixel value
# --------------------------------

im_inv_pixels = asarray(im_inv)


rows, cols = im_inv_pixels.shape
for row in range(rows):
    for col in range(cols):
        # get the current pixel value
        current_pixel_value = im_inv_pixels[row, col]
        # Manipulating your pixel values
        # for example: print pixel values that are greater than 200
        if current_pixel_value >= 255:
            print("Max Pixel Value is: ", current_pixel_value)
```

MyImageFucntion.py



```python
# MyImageFunctions.py
# Import pillow

from PIL import Image, ImageOps

# Import numpy
import numpy as np
from numpy import asarray
def myImageInverse( inImage_pixels ):



    # compute size of the matrix
    cols = np.size(inImage_pixels, 0)
    rows = np.size(inImage_pixels, 1)

    # emulate matrix numpy of the same exact size
    outImage = np.zeros((cols,rows), dtype=int)

    for col in range(cols): # traverse through i
        for row in range(rows): # traverse through j
            # now we get the difference between the two
            outImage[col][row] = 255 - inImage_pixels[col][row]


    # return the image
    return outImage;
```

Task 3: Creating a gradient grayscale image. Computing the image average.

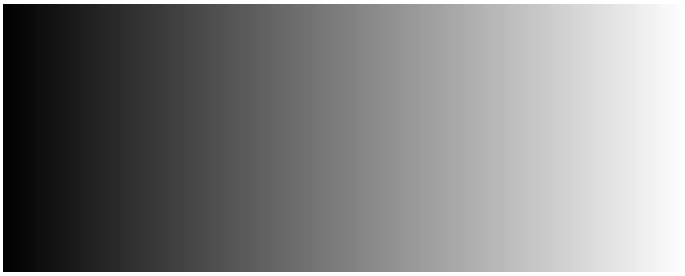


Figure 3. The Gradient Image

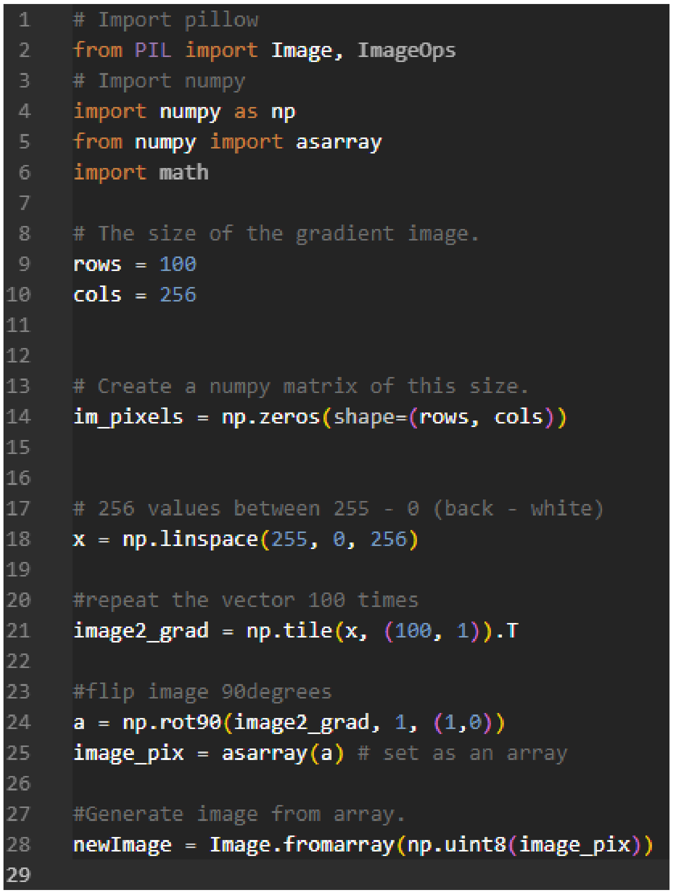
Questions for task 3:

1. What is the average pixel value in your gradient image?
   a. **127.5**

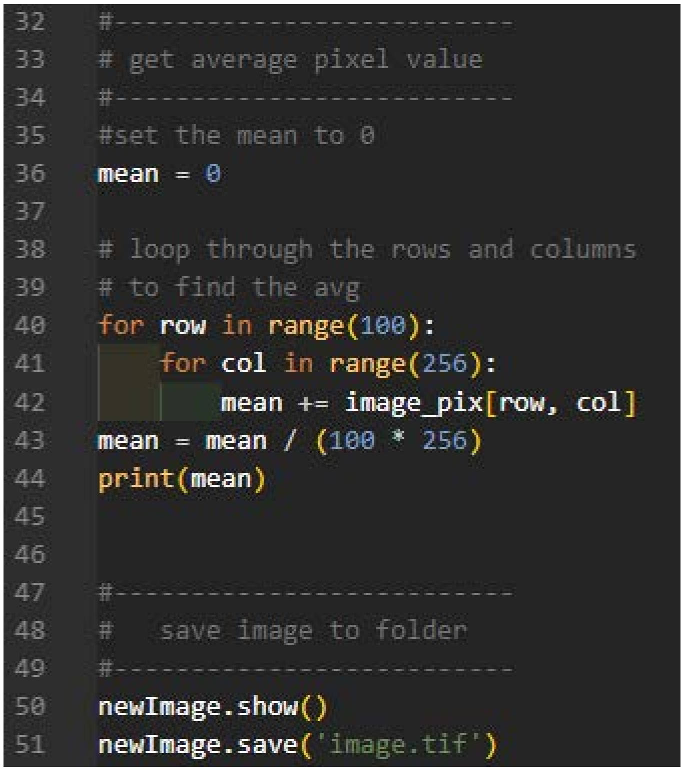
2. Why did you expect to get this value from the gradient image?
   a. **Because the average between 0 & 255 is 127.5**

3. What was the most difficult part of this task?
   a. **Figuring out how to code the average**

Task3.py



```python
# Import pillow
from PIL import Image, ImageOps
# Import numpy
import numpy as np
from numpy import asarray
import math

# The size of the gradient image.
rows = 100
cols = 256


# Create a numpy matrix of this size.
im_pixels = np.zeros(shape=(rows, cols))


# 256 values between 255 - 0 (back - white)
x = np.linspace(255, 0, 256)

#repeat the vector 100 times
image2_grad = np.tile(x, (100, 1)).T

#flip image 90degrees
a = np.rot90(image2_grad, 1, (1,0))
image_pix = asarray(a) # set as an array

#Generate image from array.
newImage = Image.fromarray(np.uint8(image_pix))

```



```python
#--------------------------
# get average pixel value
#--------------------------
#set the mean to 0
mean = 0

# loop through the rows and columns
# to find the avg
for row in range(100):
    for col in range(256):
        mean += image_pix[row, col]
mean = mean / (100 * 256)
print(mean)


#--------------------------
#   save image to folder
#--------------------------
newImage.show()
newImage.save('image.tif')
```