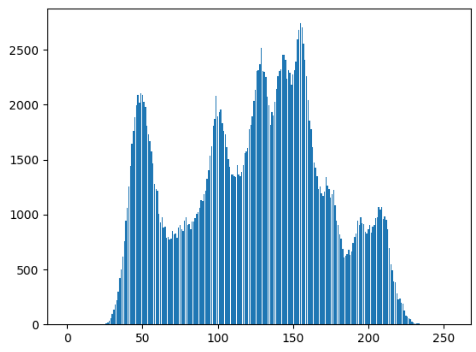
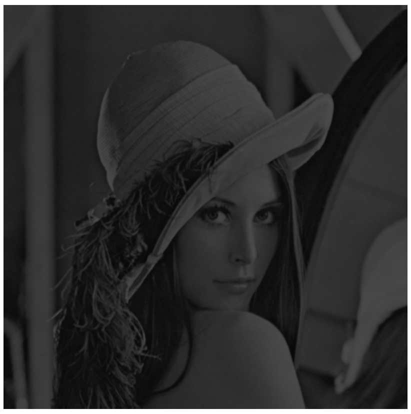
Computer Vision Homework 3   D07922015   謝銘峰

Write a program to generate (Using bmp):

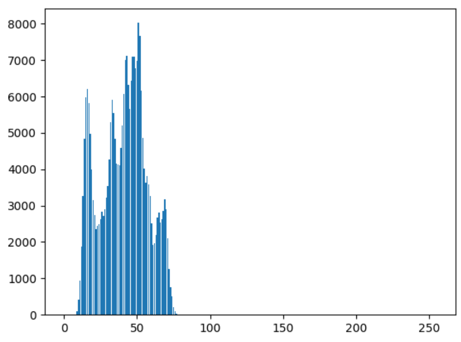
Original



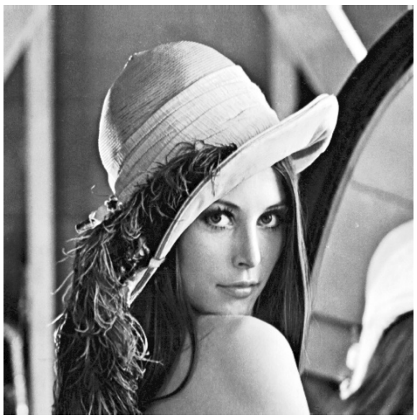
Histogram



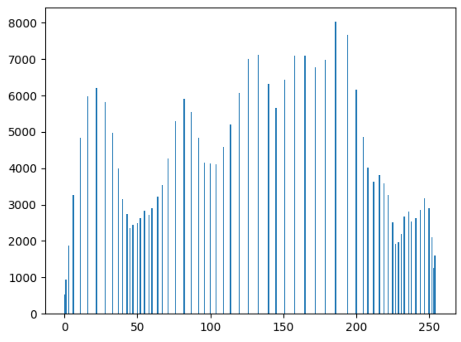
Dark 2/3



Light 1/3 histogram

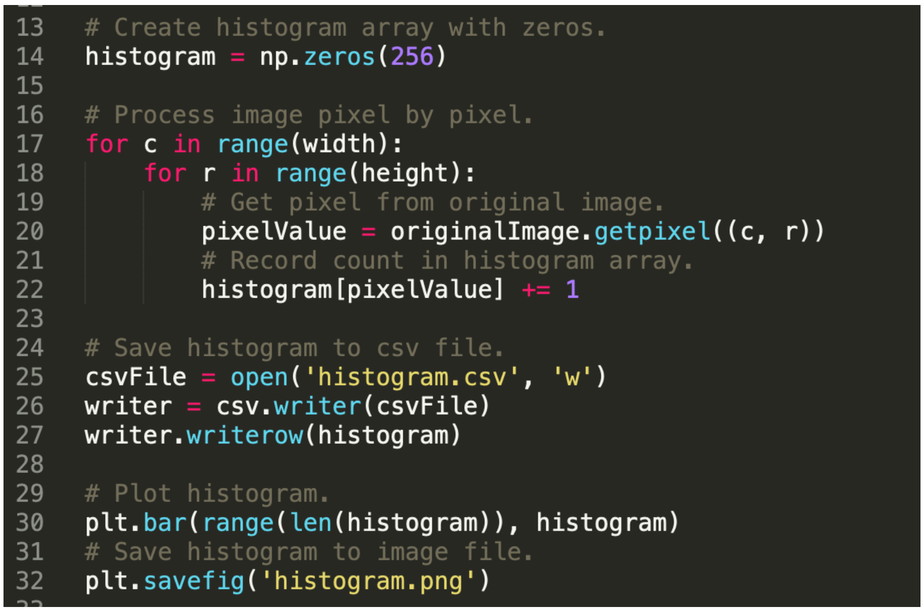


Equalization



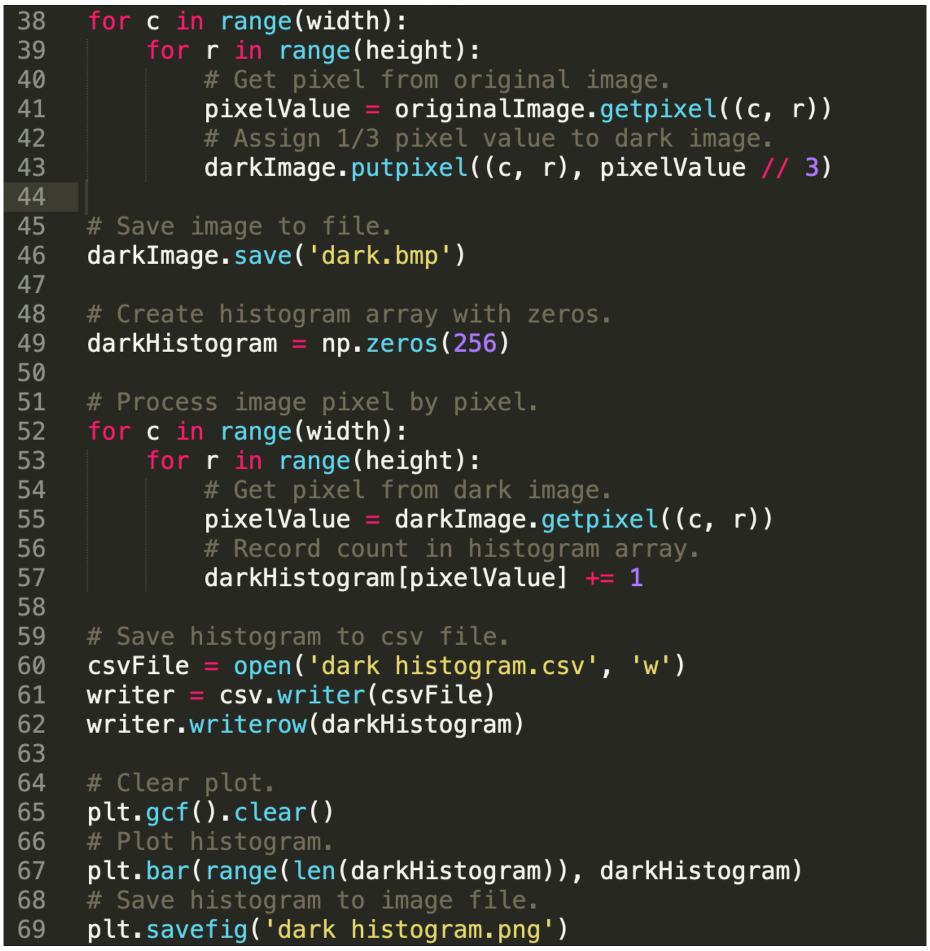
Equalization histogram

(a) original image and its histogram



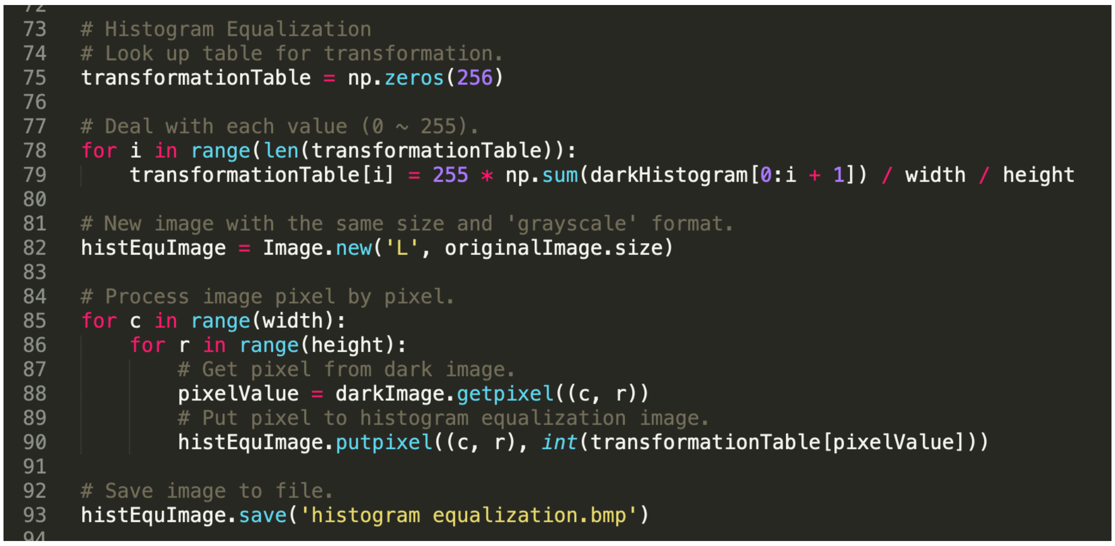
```python
# Create histogram array with zeros.
histogram = np.zeros(256)

# Process image pixel by pixel.
for c in range(width):
    for r in range(height):
        # Get pixel from original image.
        pixelValue = originalImage.getpixel((c, r))
        # Record count in histogram array.
        histogram[pixelValue] += 1

# Save histogram to csv file.
csvFile = open('histogram.csv', 'w')
writer = csv.writer(csvFile)
writer.writerow(histogram)

# Plot histogram.
plt.bar(range(len(histogram)), histogram)
# Save histogram to image file.
plt.savefig('histogram.png')
```

(b) image with intensity divided by 3 and its histogram

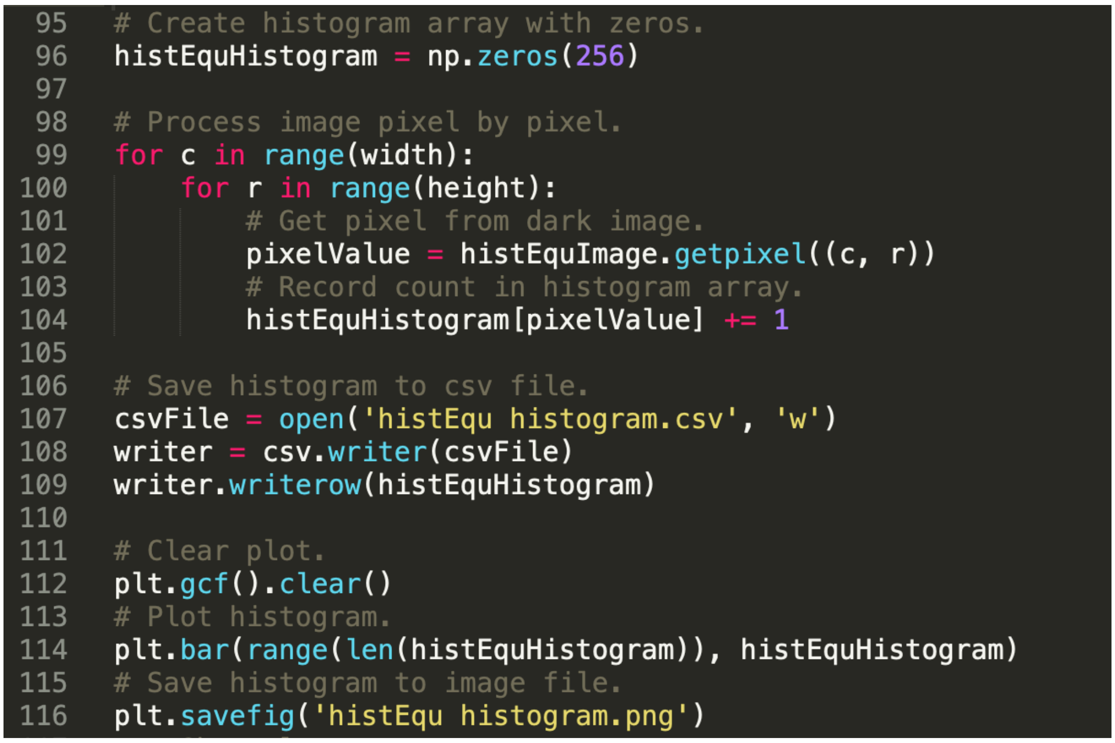


```python
for c in range(width):
    for r in range(height):
        # Get pixel from original image.
        pixelValue = originalImage.getpixel((c, r))
        # Assign 1/3 pixel value to dark image.
        darkImage.putpixel((c, r), pixelValue // 3)

# Save image to file.
darkImage.save('dark.bmp')

# Create histogram array with zeros.
darkHistogram = np.zeros(256)

# Process image pixel by pixel.
for c in range(width):
    for r in range(height):
        # Get pixel from dark image.
        pixelValue = darkImage.getpixel((c, r))
        # Record count in histogram array.
        darkHistogram[pixelValue] += 1

# Save histogram to csv file.
csvFile = open('dark histogram.csv', 'w')
writer = csv.writer(csvFile)
writer.writerow(darkHistogram)

# Clear plot.
plt.gcf().clear()
# Plot histogram.
plt.bar(range(len(darkHistogram)), darkHistogram)
# Save histogram to image file.
plt.savefig('dark histogram.png')
```

(c) image after applying histogram equalization to (b) and its histogram



```python
# Histogram Equalization
# Look up table for transformation.
transformationTable = np.zeros(256)

# Deal with each value (0 ~ 255).
for i in range(len(transformationTable)):
    transformationTable[i] = 255 * np.sum(darkHistogram[0:i + 1]) / width / height

# New image with the same size and 'grayscale' format.
histEquImage = Image.new('L', originalImage.size)

# Process image pixel by pixel.
for c in range(width):
    for r in range(height):
        # Get pixel from dark image.
        pixelValue = darkImage.getpixel((c, r))
        # Put pixel to histogram equalization image.
        histEquImage.putpixel((c, r), int(transformationTable[pixelValue]))

# Save image to file.
histEquImage.save('histogram equalization.bmp')
```



```python
# Create histogram array with zeros.
histEquHistogram = np.zeros(256)

# Process image pixel by pixel.
for c in range(width):
    for r in range(height):
        # Get pixel from dark image.
        pixelValue = histEquImage.getpixel((c, r))
        # Record count in histogram array.
        histEquHistogram[pixelValue] += 1

# Save histogram to csv file.
csvFile = open('histEqu histogram.csv', 'w')
writer = csv.writer(csvFile)
writer.writerow(histEquHistogram)

# Clear plot.
plt.gcf().clear()
# Plot histogram.
plt.bar(range(len(histEquHistogram)), histEquHistogram)
# Save histogram to image file.
plt.savefig('histEqu histogram.png')
```