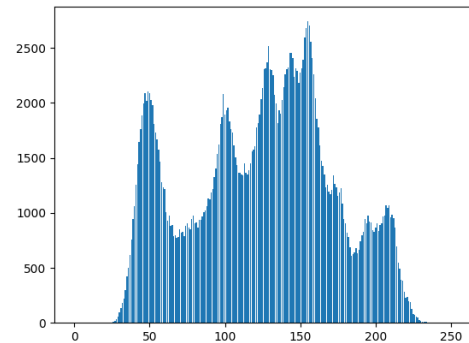


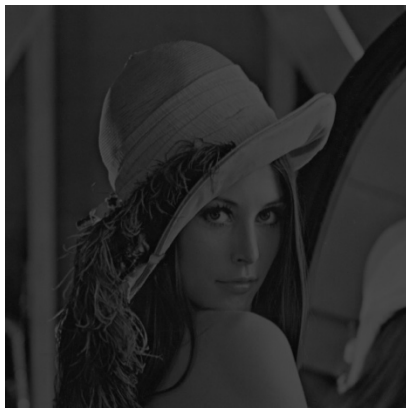
Write a program to generate (Using bmp):



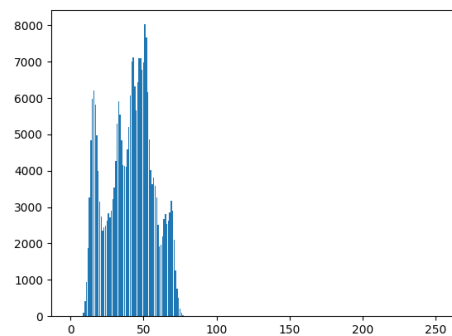
Original



Histogram



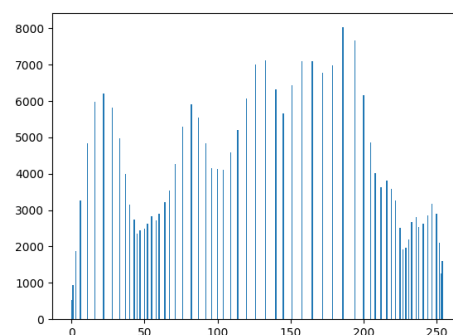
Dark 2/3



Light 1/3 histogram



Equalization



Equalization histogram

(a) original image and its histogram

```
13 # Create histogram array with zeros.
14 histogram = np.zeros(256)
15
16 # Process image pixel by pixel.
17 for c in range(width):
18     for r in range(height):
19         # Get pixel from original image.
20         pixelValue = originalImage.getpixel((c, r))
21         # Record count in histogram array.
22         histogram[pixelValue] += 1
23
24 # Save histogram to csv file.
25 csvFile = open('histogram.csv', 'w')
26 writer = csv.writer(csvFile)
27 writer.writerow(histogram)
28
29 # Plot histogram.
30 plt.bar(range(len(histogram)), histogram)
31 # Save histogram to image file.
32 plt.savefig('histogram.png')
```

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

```
38 for c in range(width):
39     for r in range(height):
40         # Get pixel from original image.
41         pixelValue = originalImage.getpixel((c, r))
42         # Assign 1/3 pixel value to dark image.
43         darkImage.putpixel((c, r), pixelValue // 3)
44
45 # Save image to file.
46 darkImage.save('dark.bmp')
47
48 # Create histogram array with zeros.
49 darkHistogram = np.zeros(256)
50
51 # Process image pixel by pixel.
52 for c in range(width):
53     for r in range(height):
54         # Get pixel from dark image.
55         pixelValue = darkImage.getpixel((c, r))
56         # Record count in histogram array.
57         darkHistogram[pixelValue] += 1
58
59 # Save histogram to csv file.
60 csvFile = open('dark histogram.csv', 'w')
61 writer = csv.writer(csvFile)
62 writer.writerow(darkHistogram)
63
64 # Clear plot.
65 plt.gcf().clear()
66 # Plot histogram.
67 plt.bar(range(len(darkHistogram)), darkHistogram)
68 # Save histogram to image file.
69 plt.savefig('dark histogram.png')
```

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

```
72
73 # Histogram Equalization
74 # Look up table for transformation.
75 transformationTable = np.zeros(256)
76
77 # Deal with each value (0 ~ 255).
78 for i in range(len(transformationTable)):
79     transformationTable[i] = 255 * np.sum(darkHistogram[0:i + 1]) / width / height
80
81 # New image with the same size and 'grayscale' format.
82 histEquImage = Image.new('L', originalImage.size)
83
84 # Process image pixel by pixel.
85 for c in range(width):
86     for r in range(height):
87         # Get pixel from dark image.
88         pixelValue = darkImage.getpixel((c, r))
89         # Put pixel to histogram equalization image.
90         histEquImage.putpixel((c, r), int(transformationTable[pixelValue]))
91
92 # Save image to file.
93 histEquImage.save('histogram equalization.bmp')
94
```

```
95 # Create histogram array with zeros.
96 histEquHistogram = np.zeros(256)
97
98 # Process image pixel by pixel.
99 for c in range(width):
100     for r in range(height):
101         # Get pixel from dark image.
102         pixelValue = histEquImage.getpixel((c, r))
103         # Record count in histogram array.
104         histEquHistogram[pixelValue] += 1
105
106 # Save histogram to csv file.
107 csvFile = open('histEqu histogram.csv', 'w')
108 writer = csv.writer(csvFile)
109 writer.writerow(histEquHistogram)
110
111 # Clear plot.
112 plt.gcf().clear()
113 # Plot histogram.
114 plt.bar(range(len(histEquHistogram)), histEquHistogram)
115 # Save histogram to image file.
116 plt.savefig('histEqu histogram.png')
```