Q1

a)

T = np.array(range(256),dtype = np.uint8)\*2

T

array([ 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24,

26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50,

52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76,

78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102,

104, 106, 108, 110, 112, 114, 116, 118, 120, 122, 124, 126, 128,

130, 132, 134, 136, 138, 140, 142, 144, 146, 148, 150, 152, 154,

156, 158, 160, 162, 164, 166, 168, 170, 172, 174, 176, 178, 180,

182, 184, 186, 188, 190, 192, 194, 196, 198, 200, 202, 204, 206,

208, 210, 212, 214, 216, 218, 220, 222, 224, 226, 228, 230, 232,

234, 236, 238, 240, 242, 244, 246, 248, 250, 252, 254, 0, 2,

4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28,

30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54,

56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78, 80,

82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102, 104, 106,

108, 110, 112, 114, 116, 118, 120, 122, 124, 126, 128, 130, 132,

134, 136, 138, 140, 142, 144, 146, 148, 150, 152, 154, 156, 158,

160, 162, 164, 166, 168, 170, 172, 174, 176, 178, 180, 182, 184,

186, 188, 190, 192, 194, 196, 198, 200, 202, 204, 206, 208, 210,

212, 214, 216, 218, 220, 222, 224, 226, 228, 230, 232, 234, 236,

238, 240, 242, 244, 246, 248, 250, 252, 254], dtype=uint8)

b)

cc = np.array(range(256),dtype = np.uint8)

complements = 255 - cc

complements

array([255, 254, 253, 252, 251, 250, 249, 248, 247, 246, 245, 244, 243,

242, 241, 240, 239, 238, 237, 236, 235, 234, 233, 232, 231, 230,

229, 228, 227, 226, 225, 224, 223, 222, 221, 220, 219, 218, 217,

216, 215, 214, 213, 212, 211, 210, 209, 208, 207, 206, 205, 204,

203, 202, 201, 200, 199, 198, 197, 196, 195, 194, 193, 192, 191,

190, 189, 188, 187, 186, 185, 184, 183, 182, 181, 180, 179, 178,

177, 176, 175, 174, 173, 172, 171, 170, 169, 168, 167, 166, 165,

164, 163, 162, 161, 160, 159, 158, 157, 156, 155, 154, 153, 152,

151, 150, 149, 148, 147, 146, 145, 144, 143, 142, 141, 140, 139,

138, 137, 136, 135, 134, 133, 132, 131, 130, 129, 128, 127, 126,

125, 124, 123, 122, 121, 120, 119, 118, 117, 116, 115, 114, 113,

112, 111, 110, 109, 108, 107, 106, 105, 104, 103, 102, 101, 100,

99, 98, 97, 96, 95, 94, 93, 92, 91, 90, 89, 88, 87,

86, 85, 84, 83, 82, 81, 80, 79, 78, 77, 76, 75, 74,

73, 72, 71, 70, 69, 68, 67, 66, 65, 64, 63, 62, 61,

60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 49, 48,

47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35,

34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22,

21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9,

8, 7, 6, 5, 4, 3, 2, 1, 0], dtype=uint8)

Q4

c = io.imread("cameraman.tif")

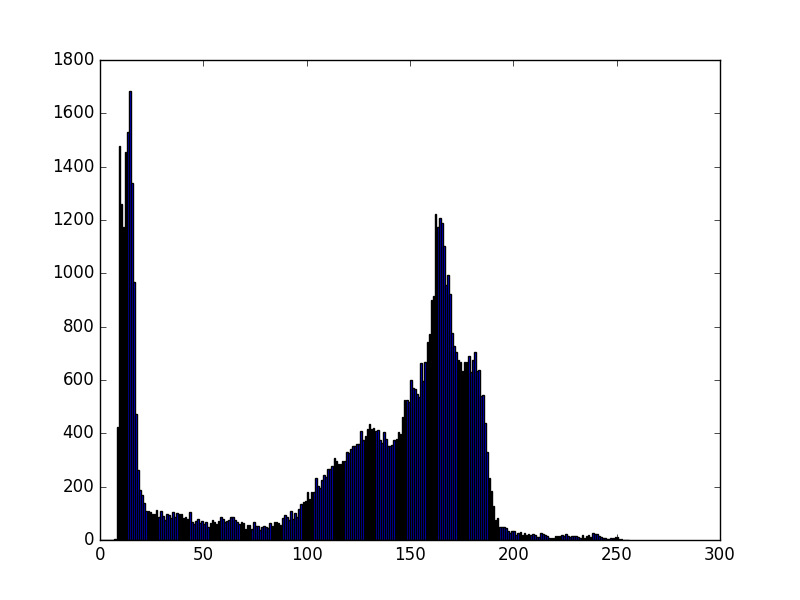
h = np.zeros(256)

for i in c:

for j in i:

h[j] += 1

plt.bar(range(256),h)



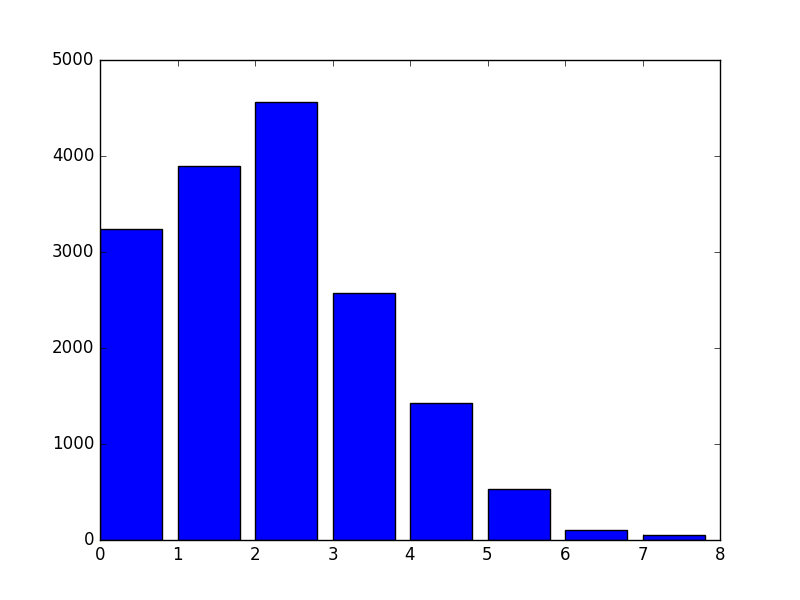
Q5

y = [3244,3899,4559,2573,1428,530,101,50]

x = range(8)

plt.bar(x,y)

image below is the histogram



# equalization

s = sum(y)

equalized\_x= []

fraction = 7./s # (L-1)/total number of pixels

ss = 0

# used CDF to do mapping

for i in y:

ss += i

v = ss\*fraction

equalized\_x.append(round(v))

equalized\_x = map(int,equalized\_x)

equalized\_x

# the original intensity levels are

[0, 1, 2, 3, 4, 5, 6, 7]

mapped to

[1, 3, 5, 6, 7, 7, 7, 7]

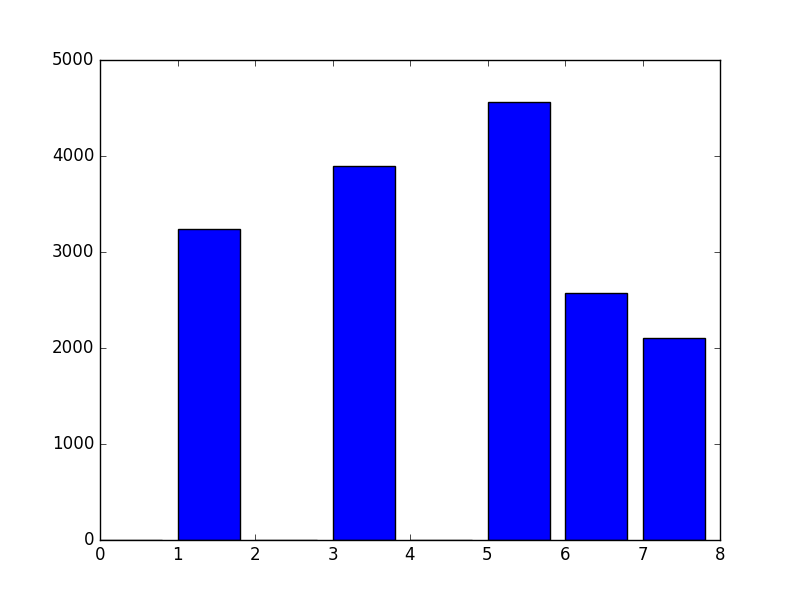
new\_y =np.zeros(8)

for i,v in enumerate(equalized\_x):

new\_y[v] += y[i]

plt.bar(x,new\_y)

The histogram below is the one after equalization



Q7

q7 = np.ndarray(shape = (8,8),dtype = 'int')

q7[0] = [12,6,5,13,14,14,16,15]

q7[1] = [11,10,8,5,8,11,14,14]

q7[2]= [9,8,3,4,7,12,18,19]

q7[3]= [10,7,4,2,10,12,13,17]

q7[4]= [16,9,13,13,16,19,19,17]

q7[5]= [12,10,14,15,18,18,16,14]

q7[6]= [11,8,10,12,14,13,14,15]

q7[7]= [8,6,3,7,9,11,12,12]

q7

# this is the input data

array([[12, 6, 5, 13, 14, 14, 16, 15],

[11, 10, 8, 5, 8, 11, 14, 14],

[ 9, 8, 3, 4, 7, 12, 18, 19],

[10, 7, 4, 2, 10, 12, 13, 17],

[16, 9, 13, 13, 16, 19, 19, 17],

[12, 10, 14, 15, 18, 18, 16, 14],

[11, 8, 10, 12, 14, 13, 14, 15],

[ 8, 6, 3, 7, 9, 11, 12, 12]])

# plot histogram

xQ7 = range(20)

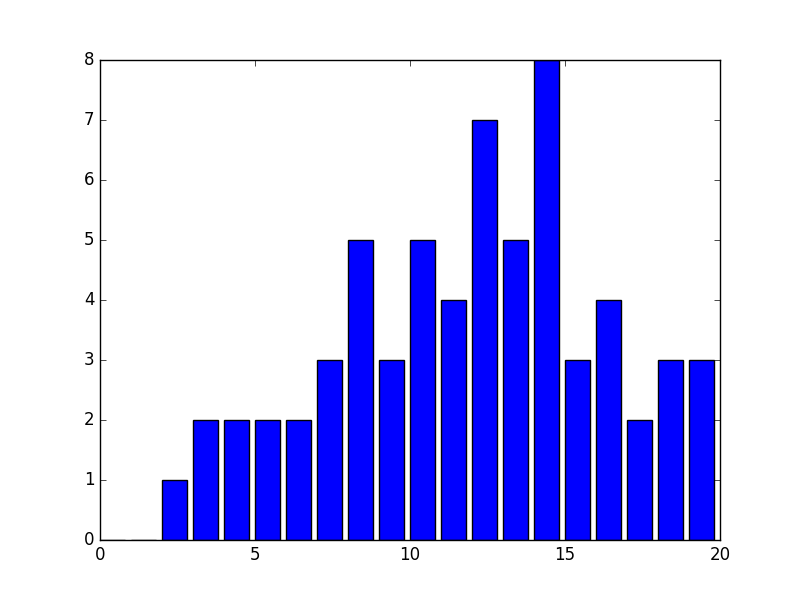
yQ7 = np.zeros(20)

for i in q7:

for j in i:

yQ7[j] += 1

plt.bar(xQ7,yQ7)



# conduct equalization

sQ7 = sum(yQ7)

equalized\_xQ7 = []

fraction = 19./sQ7

ss = 0

for i in yQ7:

ss += i

v = ss\*fraction

equalized\_xQ7.append(round(v))

equalized\_xQ7 = map(int,equalized\_xQ7)

equalized\_xQ7

# the original intensity levels are

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

mapped to

[0, 0, 0, 1, 1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 15, 15, 17, 17, 18, 19]

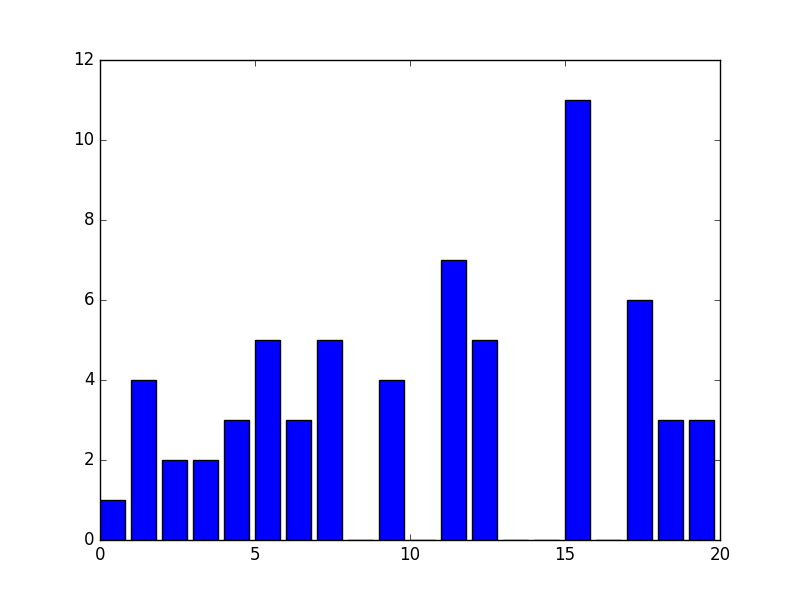
new\_yQ7 =np.zeros(20)

for i,v in enumerate(equalized\_xQ7):

new\_yQ7[v] += yQ7[i]

plt.bar(xQ7,new\_yQ7)

# The histogram below is the one after equlization



#computer the 8X8 grid after equalization

equalized\_q7 = np.ndarray(shape = (8,8),dtype = int)

for i in range(8):

for j in range(8):

equalized\_q7[i][j] = equalized\_xQ7[q7[i][j]]

equalized\_q7

# this is the data after equalization

array([[11, 3, 2, 12, 15, 15, 17, 15],

[ 9, 7, 5, 2, 5, 9, 15, 15],

[ 6, 5, 1, 1, 4, 11, 18, 19],

[ 7, 4, 1, 0, 7, 11, 12, 17],

[17, 6, 12, 12, 17, 19, 19, 17],

[11, 7, 15, 15, 18, 18, 17, 15],

[ 9, 5, 7, 11, 15, 12, 15, 15],

[ 5, 3, 1, 4, 6, 9, 11, 11]])

q7

# this is the original data

array([[12, 6, 5, 13, 14, 14, 16, 15],

[11, 10, 8, 5, 8, 11, 14, 14],

[ 9, 8, 3, 4, 7, 12, 18, 19],

[10, 7, 4, 2, 10, 12, 13, 17],

[16, 9, 13, 13, 16, 19, 19, 17],

[12, 10, 14, 15, 18, 18, 16, 14],

[11, 8, 10, 12, 14, 13, 14, 15],

[ 8, 6, 3, 7, 9, 11, 12, 12]])

Q13

# the idea is to apply piece-wise function to part of the data

TT = np.array(range(256),dtype = np.uint8)

TT[0:100]\*(3./5)+40

TT[100:150]\*(12./5)-200

TT[150:]\*(1./3)+170

array([ 220. , 220.33333333, 220.66666667, 221. ,

221.33333333, 221.66666667, 222. , 222.33333333,

222.66666667, 223. , 223.33333333, 223.66666667,

224. , 224.33333333, 224.66666667, 225. ,

225.33333333, 225.66666667, 226. , 226.33333333,

226.66666667, 227. , 227.33333333, 227.66666667,

228. , 228.33333333, 228.66666667, 229. ,

229.33333333, 229.66666667, 230. , 230.33333333,

230.66666667, 231. , 231.33333333, 231.66666667,

232. , 232.33333333, 232.66666667, 233. ,

233.33333333, 233.66666667, 234. , 234.33333333,

234.66666667, 235. , 235.33333333, 235.66666667,

236. , 236.33333333, 236.66666667, 237. ,

237.33333333, 237.66666667, 238. , 238.33333333,

238.66666667, 239. , 239.33333333, 239.66666667,

240. , 240.33333333, 240.66666667, 241. ,

241.33333333, 241.66666667, 242. , 242.33333333,

242.66666667, 243. , 243.33333333, 243.66666667,

244. , 244.33333333, 244.66666667, 245. ,

245.33333333, 245.66666667, 246. , 246.33333333,

246.66666667, 247. , 247.33333333, 247.66666667,

248. , 248.33333333, 248.66666667, 249. ,

249.33333333, 249.66666667, 250. , 250.33333333,

250.66666667, 251. , 251.33333333, 251.66666667,

252. , 252.33333333, 252.66666667, 253. ,

253.33333333, 253.66666667, 254. , 254.33333333,

254.66666667, 255. ])