In [2]:
```
import cv2
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Question 1

In [2]:
```
X = cv2.imread('4_1.bmp')
cv2.imshow('original', X)
cv2.waitKey(0)
cv2.destroyAllWindows()
print(X.shape)

X[:,:,1] = ((X[0::,0::,1]/255)**.67) * 255
cv2.imshow('fixed', X)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
(202, 282, 3)
```

Question 2

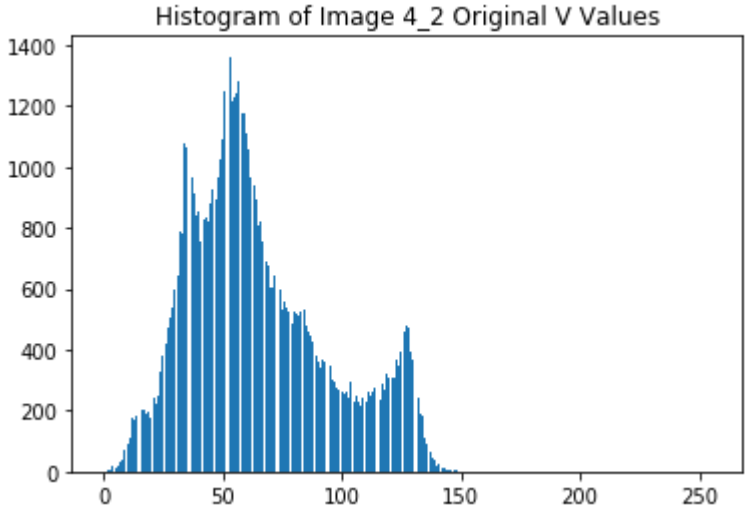
Linear Stretching

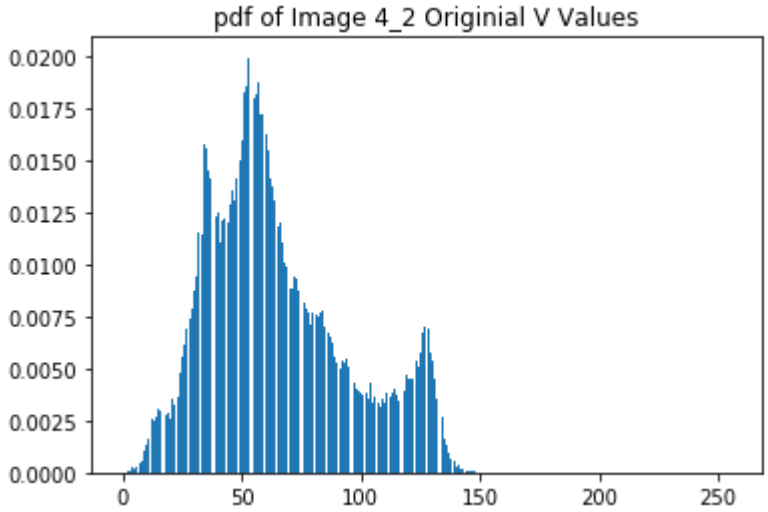
In [3]:
```
def linearStretch(x, y, array):
    array[:, :, 2] = ((array[:, :, 2] - x) / (y - x)) * 255
```

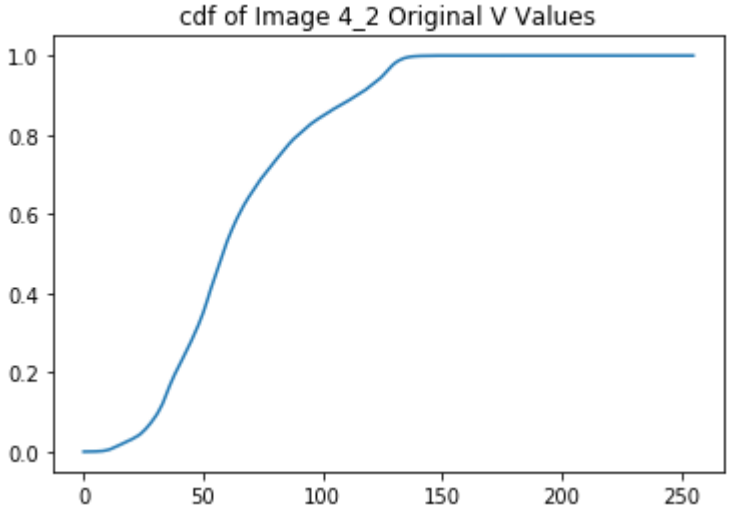
In [4]:
```
im = cv2.imread('4_2.jpg')
imConvert = cv2.cvtColor(im, cv2.COLOR_BGR2HSV)
originalCount = np.zeros(256)


print(imConvert[:, :, 2].shape)

for x in range(0, 228):
    for y in range(0, 300):
        value = imConvert[:, :, 2][x,y]
        originalCount[value] = originalCount[value] + 1

plt.title('Histogram of Image 4_2 Original V Values')
plt.bar(np.arange(0,256,1), originalCount)
plt.show()

plt.title('pdf of Image 4_2 Originial V Values')
plt.bar(np.arange(0, 256,1), originalCount / (228 * 300))
plt.show()

plt.title('cdf of Image 4_2 Original V Values')
plt.plot(np.arange(0, 256, 1), np.cumsum(originalCount / (228 * 300)))
plt.show()

minimum = np.min(imConvert[:, :, 2])
maximum = np.max(imConvert[:, :, 2])

newV = linearStretch(minimum, maximum, imConvert)
newCount = np.zeros(256)
for x in range(0, 228):
    for y in range(0, 300):
        value = imConvert[:, :, 2][x,y]
        newCount[value] = newCount[value] + 1

plt.title('Histogram of Image 4_2 V Values after Linear Stretch')
plt.bar(np.arange(0,256,1), newCount)
plt.show()

plt.title('pdf of Image 4_2 V Values after Linear Stretch')
plt.bar(np.arange(0, 256,1), newCount / (228 * 300))
plt.show()

plt.title('cdf of Image 4_2 V Values after Linear Stretch')
plt.plot(np.arange(0, 256, 1), np.cumsum(newCount / (228 * 300)))
plt.show()

newImage = cv2.cvtColor(imConvert, cv2.COLOR_HSV2BGR)

cv2.imshow('4_2 original', im)
cv2.imshow('4_2 linear stretched image', newImage)
cv2.waitKey(0)
cv2.destroyAllWindows()



im2 = cv2.imread('4_3.jpg')
```

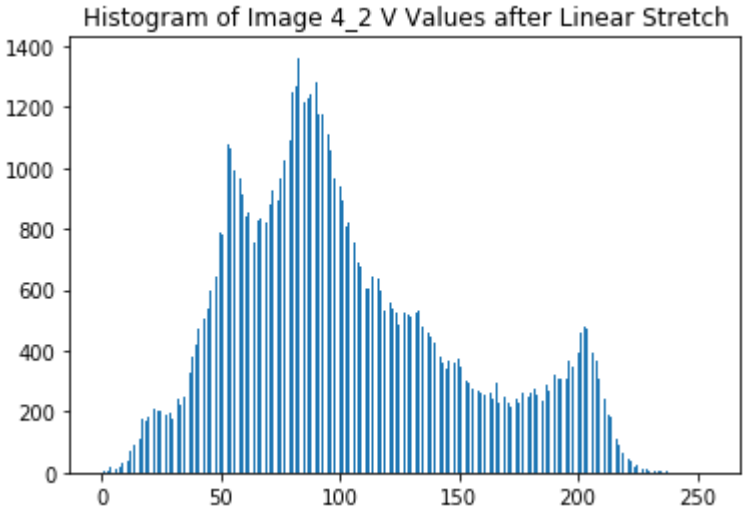
```
im2Convert = cv2.cvtColor(im2, cv2.COLOR_BGR2HSV)
originalCount2 = np.zeros(256)
print(im2.shape)
for x in range(0, 240):
    for y in range(0, 320):
        value = im2Convert[:, :, 2][x,y]
        originalCount2[value] = originalCount2[value] + 1

plt.title('Histogram of Image 4_3 Original V Values')
plt.bar(np.arange(0,256,1), originalCount2)
plt.show()

plt.title('pdf of Image 4_3 Originial V Values')
plt.bar(np.arange(0, 256,1), originalCount2 / (240 * 320))
plt.show()

plt.title('cdf of Image 4_3 Original V Values')
plt.plot(np.arange(0, 256, 1), np.cumsum(originalCount2 / (240 * 320)))
plt.show()

minimum2 = np.min(im2Convert[:, :, 2])
maximum2 = np.max(im2Convert[:, :, 2])

new2V = linearStretch(minimum2, maximum2, im2Convert)
newCount2 = np.zeros(256)
for x in range(0, 240):
    for y in range(0, 320):
        value = im2Convert[:, :, 2][x,y]
        newCount2[value] = newCount2[value] + 1
plt.title('Histogram of Image 4_3 V Values after Linear Stretch')
plt.bar(np.arange(0,256,1), newCount2)
plt.show()

plt.title('pdf of Image 4_3 V Values after Linear Stretch')
plt.bar(np.arange(0, 256,1), newCount2 / (240 * 320))
plt.show()

plt.title('cdf of Image 4_3 V Values after Linear Stretch')
plt.plot(np.arange(0, 256, 1), np.cumsum(newCount2 / (240 * 320)))
plt.show()

newImage2 = cv2.cvtColor(im2Convert, cv2.COLOR_HSV2BGR)

cv2.imshow('4_3 original', im2)
cv2.imshow('4_3 linear stretched image', newImage2)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

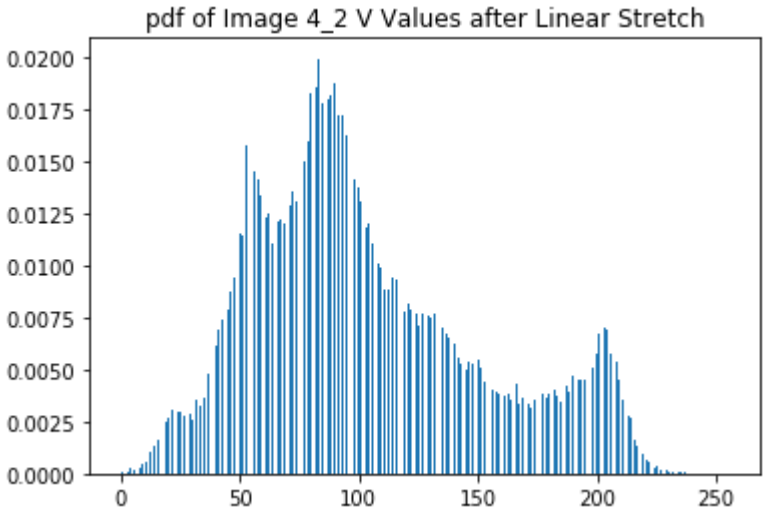
```
(228, 300)
```

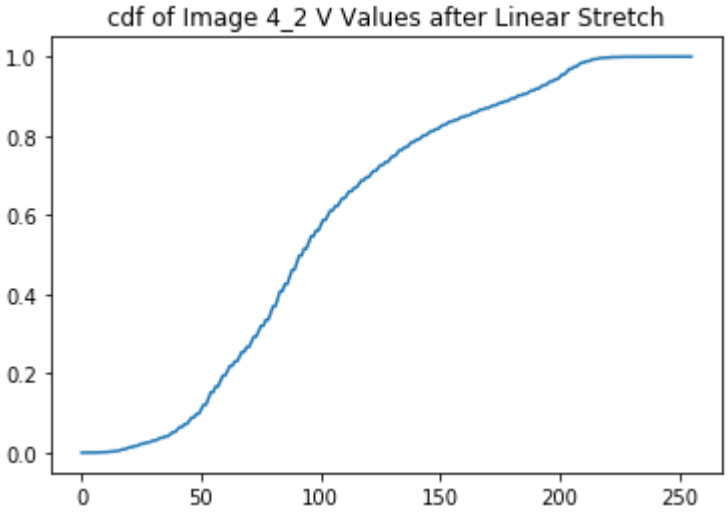

Histogram of Image 4_2 Original V Values
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250


pdf of Image 4_2 Originial V Values
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250

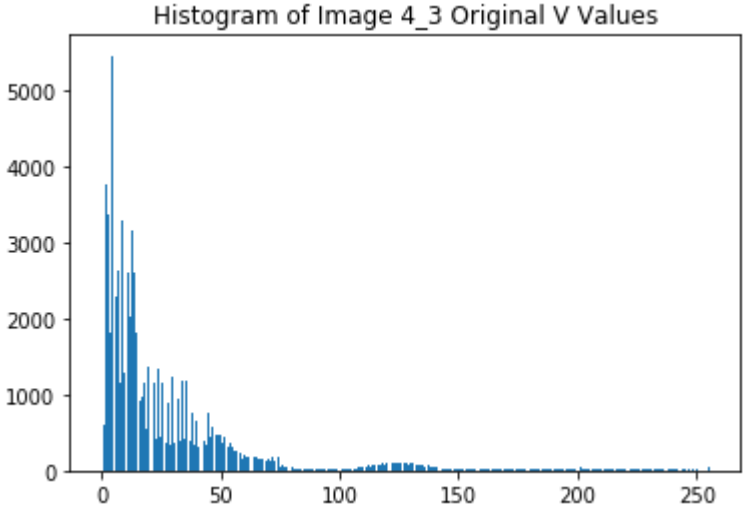

cdf of Image 4_2 Original V Values
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250

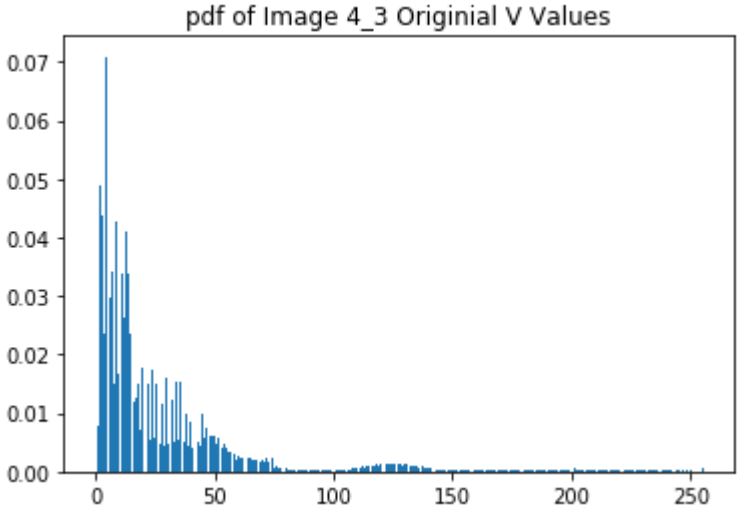

Histogram of Image 4_2 V Values after Linear Stretch
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250

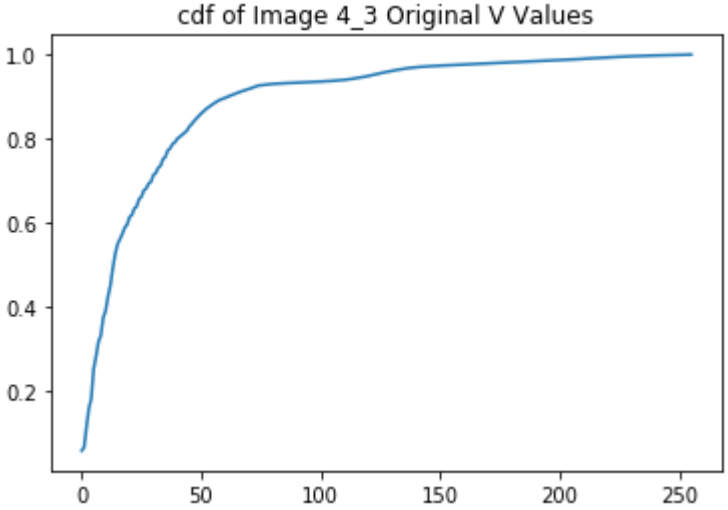

pdf of Image 4_2 V Values after Linear Stretch
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250

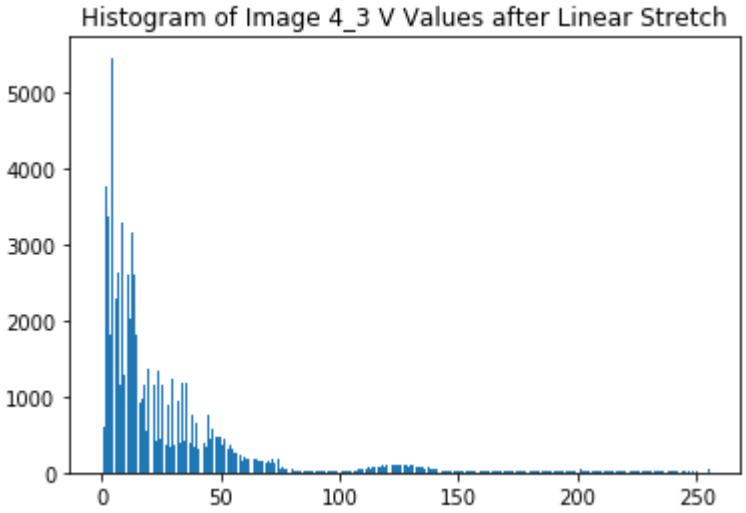

cdf of Image 4_2 V Values after Linear Stretch
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250

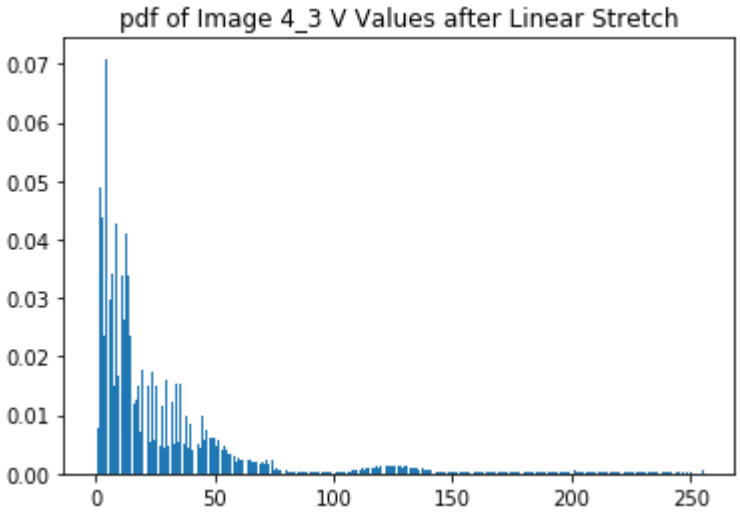
```
(240, 320, 3)
```

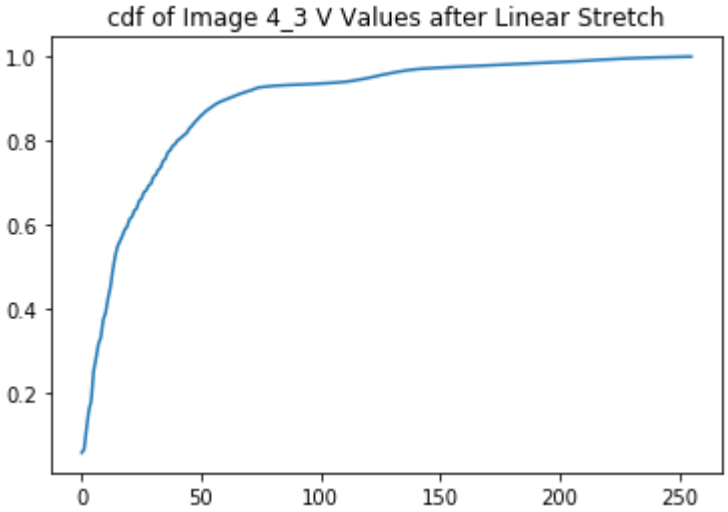

Histogram of Image 4_3 Original V Values
5000
4000
3000
2000
1000
0
0
50
100
150
200
250


pdf of Image 4_3 Originial V Values
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
50
100
150
200
250


cdf of Image 4_3 Original V Values
1.0
0.8
0.6
0.4
0.2
0
50
100
150
200
250







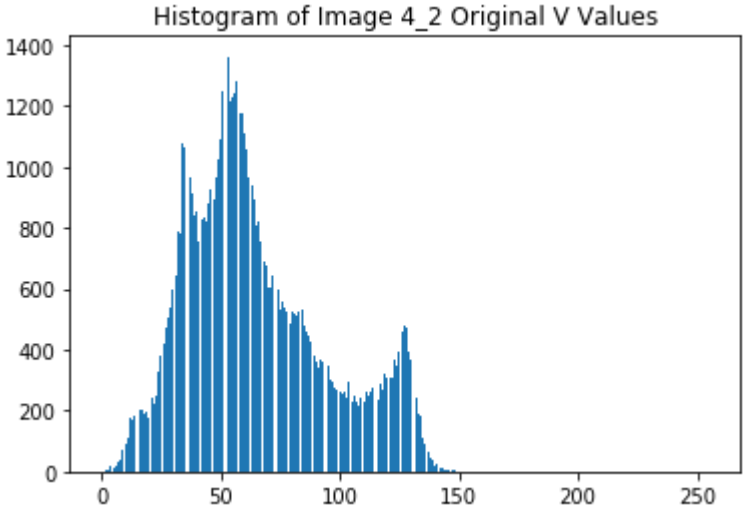
Histogram Equalization

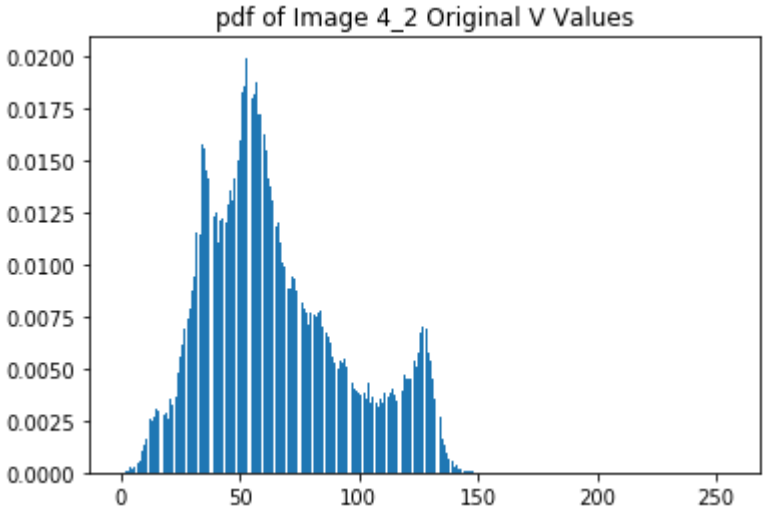
In [5]:
```
def histogramEqualization(minimum, maximum, xrange, yrange, array):
    count = np.zeros(maximum + 1)
    for y in range(0, yrange):
        for x in range(0, xrange):
            value = array[x, y]
            count[value] = count[value] + 1
    return count
```

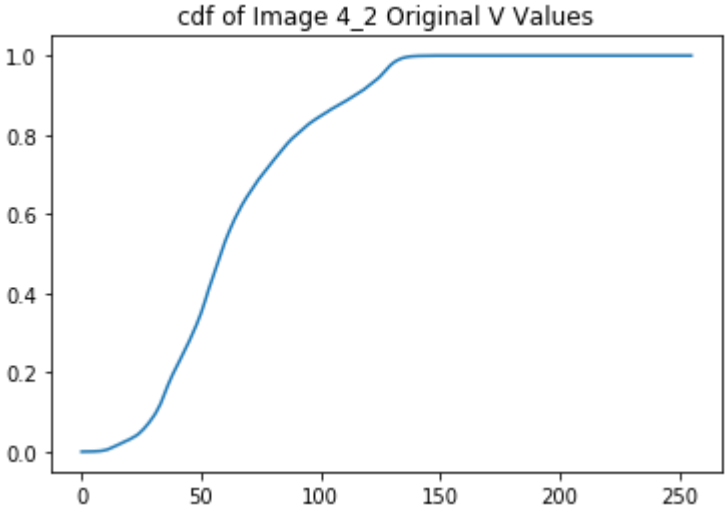
In [6]:
```
im = cv2.imread('4_2.jpg')
imConvert = cv2.cvtColor(im, cv2.COLOR_BGR2HSV)
originalCount = np.zeros(256)
counter = 0

print(imConvert[:, :, 2].shape)

for x in range(0, 228):
    for y in range(0, 300):
        counter = counter + 1
        value = imConvert[:, :, 2][x,y]
        originalCount[value] = originalCount[value] + 1

plt.title('Histogram of Image 4_2 Original V Values')
plt.bar(np.arange(0,256,1), originalCount)
plt.show()

plt.title('pdf of Image 4_2 Original V Values')
pdf = originalCount / (228 * 300)
plt.bar(np.arange(0,256,1), pdf)
plt.show()

cdf = np.cumsum(pdf)
plt.title('cdf of Image 4_2 Original V Values')
plt.plot(np.arange(0, 256, 1), cdf)
plt.show()

newHistogram = 255 * cdf
plt.title('Histogram of Image 4_2 V Values after Equalization')
plt.bar(np.arange(0, 256, 1), newHistogram)
plt.show()

newPdf = newHistogram / (228 * 300)
plt.title('pdf of Image 4_2 V Values after Equalization')
plt.bar(np.arange(0, 256, 1), newPdf)
plt.show()

newCdf = np.cumsum(newPdf)
plt.title('cdf of Image 4_2 V Values after Equalization')
plt.plot(np.arange(0, 256, 1), newCdf)
plt.show()

for x in range(0, 228):
    for y in range(0, 300):
        value = imConvert[:, :, 2][x,y]
        newValue = newHistogram[value]
        imConvert[:, :, 2][x,y] = newValue

equalizedImage1 = cv2.cvtColor(imConvert, cv2.COLOR_HSV2BGR)
cv2.imshow('4_2 original', im)
cv2.imshow('4_2 equalized image', equalizedImage1)
cv2.waitKey(0)
cv2.destroyAllWindows()

im2 = cv2.imread('4_3.jpg')
im2Convert = cv2.cvtColor(im2, cv2.COLOR_BGR2HSV)
```

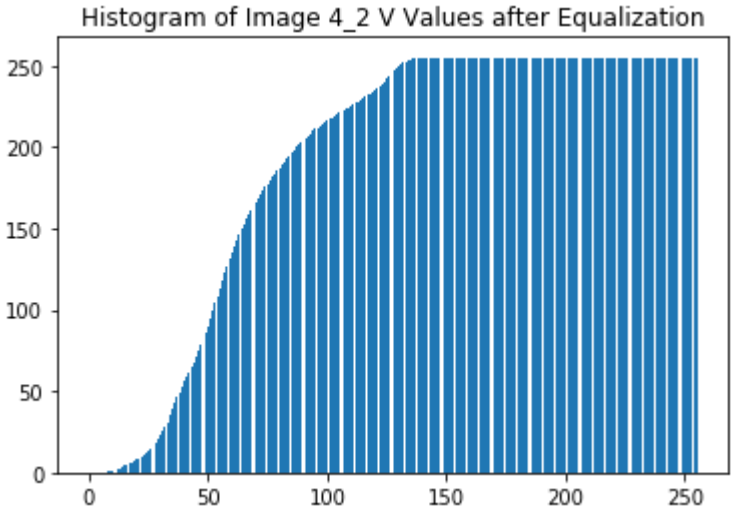
```
originalCount2 = np.zeros(256)
counter2 = 0
print(im2Convert[:, :, 2].shape)

for x in range(0, 240):
    for y in range(0, 320):
        counter2 = counter2 + 1
        value = im2Convert[:, :, 2][x,y]
        originalCount2[value] = originalCount2[value] + 1

plt.title('Histogram of Image 4_3 Original V Values')
plt.bar(np.arange(0,256,1), originalCount2)
plt.show()

plt.title('pdf of Image 4_3 Original V Values')
pdf2 = originalCount / (240 * 320)
plt.bar(np.arange(0,256,1), pdf2)
plt.show()

cdf2 = np.cumsum(pdf2)
plt.title('cdf of Image 4_3 Original V Values')
plt.plot(np.arange(0, 256, 1), cdf2)
plt.show()

newHistogram2 = 255 * cdf2
plt.title('Histogram of Image 4_3 V Values after Equalization')
plt.bar(np.arange(0, 256, 1), newHistogram2)
plt.show()

newPdf2 = newHistogram2 / (240 * 320)
plt.title('pdf of Image 4_3 V Values after Equalization')
plt.bar(np.arange(0, 256, 1), newPdf2)
plt.show()

newCdf2 = np.cumsum(newPdf2)
plt.title('cdf of Image 4_3 V Values after Equalization')
plt.plot(np.arange(0, 256, 1), newCdf2)
plt.show()

for x in range(0, 240):
    for y in range(0, 320):
        value = im2Convert[:, :, 2][x,y]
        newValue = newHistogram2[value]
        im2Convert[:, :, 2][x,y] = newValue

equalizedImage2 = cv2.cvtColor(im2Convert, cv2.COLOR_HSV2BGR)
cv2.imshow('4_3 original', im2)
cv2.imshow('4_3 equalized image', equalizedImage2)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

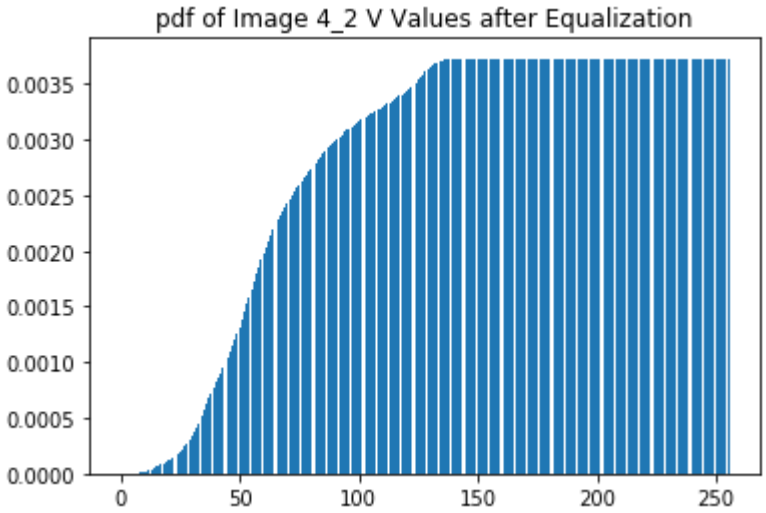
```
(228, 300)
```

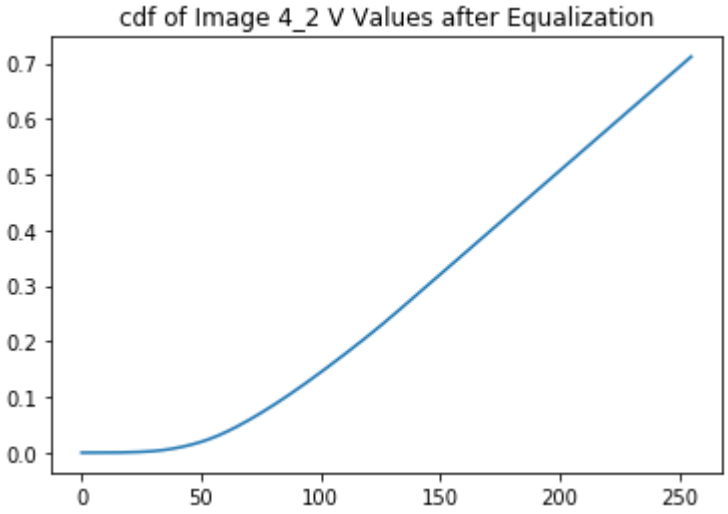

Histogram of Image 4_2 Original V Values
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250


pdf of Image 4_2 Original V Values
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250

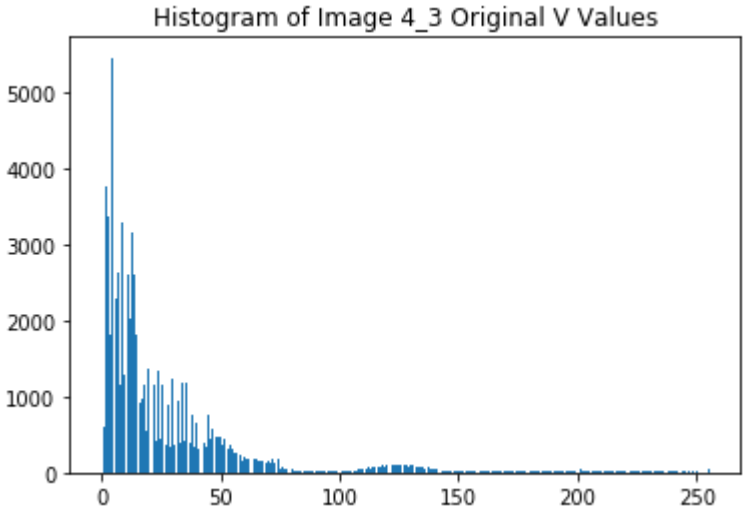

cdf of Image 4_2 Original V Values
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250

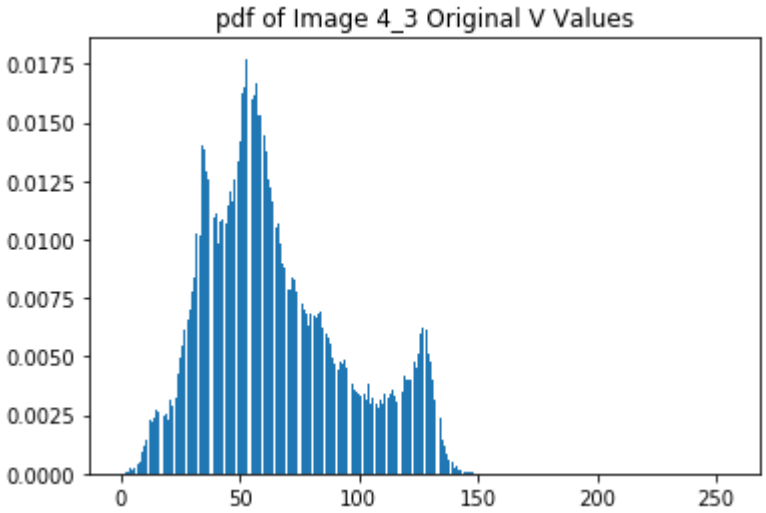

Histogram of Image 4_2 V Values after Equalization
250
200
150
100
50
0
0
50
100
150
200
250

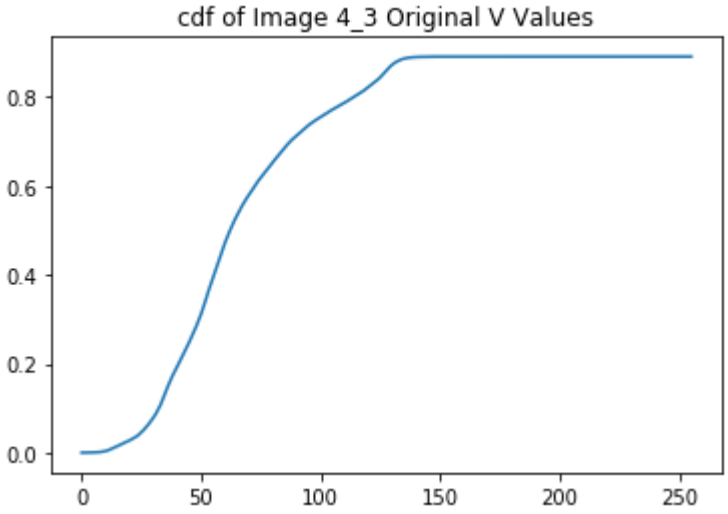

pdf of Image 4_2 V Values after Equalization
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
50
100
150
200
250

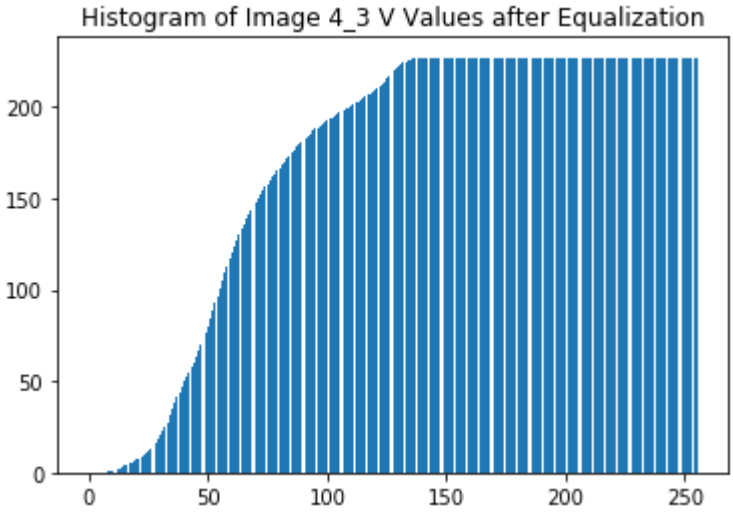

cdf of Image 4_2 V Values after Equalization
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
50
100
150
200
250

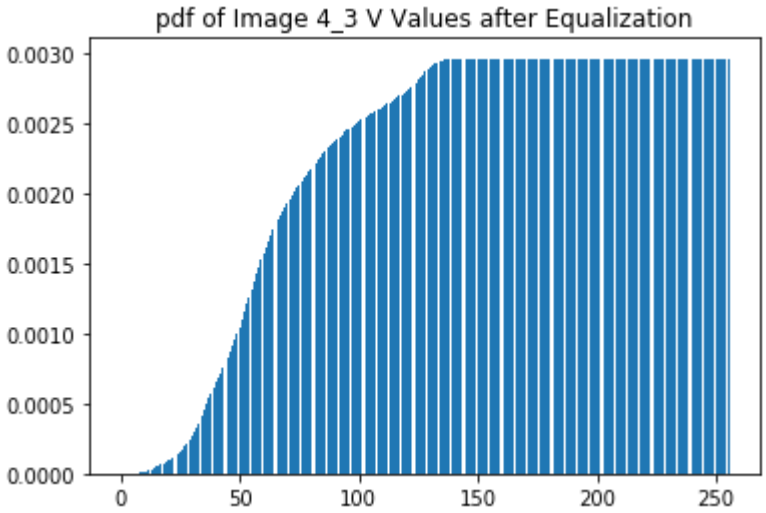
```
(240, 320)
```

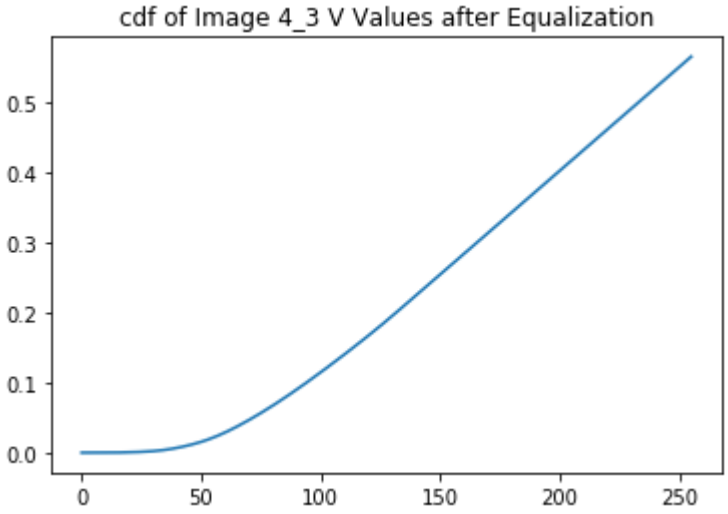

Histogram of Image 4_3 Original V Values
5000
4000
3000
2000
1000
0
0
50
100
150
200
250


pdf of Image 4_3 Original V Values
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250


cdf of Image 4_3 Original V Values
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250



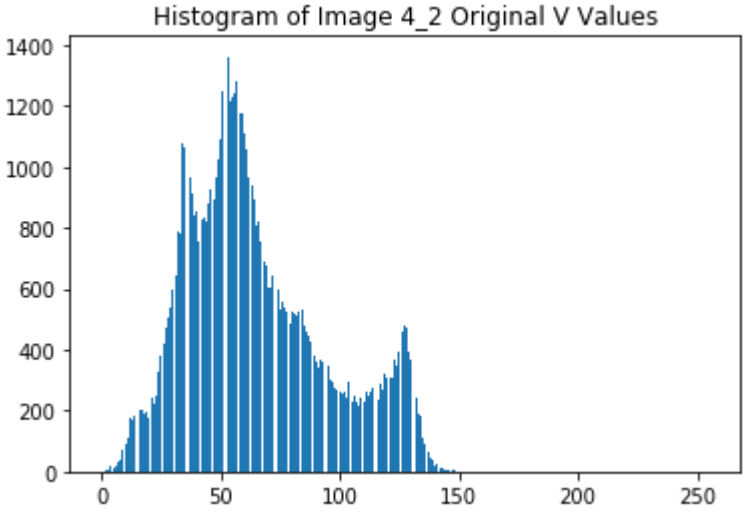


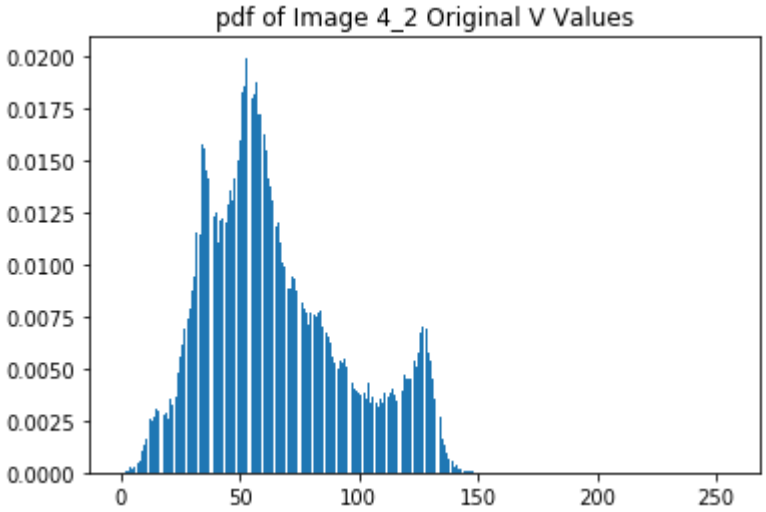


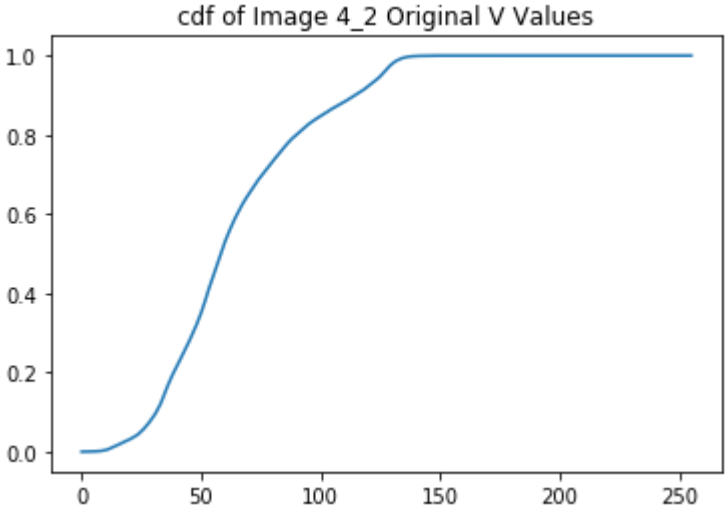
Histogram Specification

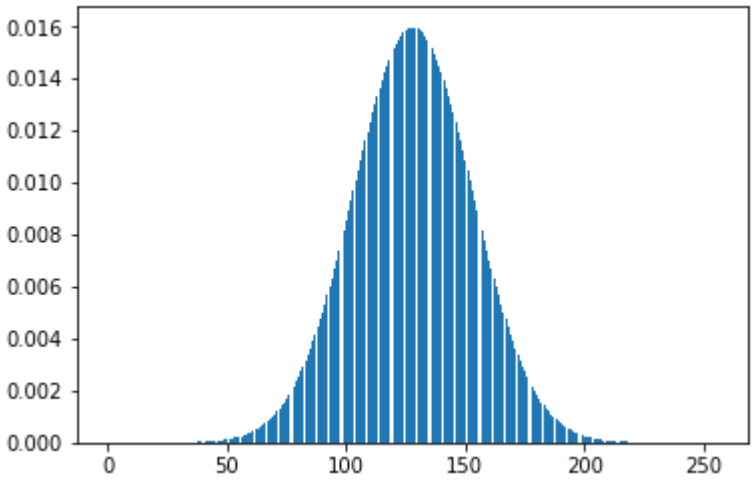
In [ ]:
```
im = cv2.imread('4_2.jpg')
imConvert = cv2.cvtColor(im, cv2.COLOR_BGR2HSV)
originalCount = np.zeros(256)
counter = 0

print(imConvert[:, :, 2].shape)

for x in range(0, 228):
    for y in range(0, 300):
        counter = counter + 1
        value = imConvert[:, :, 2][x,y]
        originalCount[value] = originalCount[value] + 1

plt.title('Histogram of Image 4_2 Original V Values')
plt.bar(np.arange(0,256,1), originalCount)
plt.show()

plt.title('pdf of Image 4_2 Original V Values')
pdf = originalCount / (228 * 300)
plt.bar(np.arange(0,256,1), pdf)
plt.show()

cdf = np.cumsum(pdf)
plt.title('cdf of Image 4_2 Original V Values')
plt.plot(np.arange(0, 256, 1), cdf)
plt.show()

#create normal distribution
x = np.arange(0, 256, 1)
y = (1 / (np.sqrt(np.pi * 2 * 25**2))) * np.exp(-((x - 128)**2) / (2 * 25**2))
plt.bar(x, y)
plt.show()
normalCdf = np.cumsum(y)
plt.plot(normalCdf)
plt.show()

targetCdf = np.uint8(normalCdf * 255)
plt.bar(np.arange(0, 256, 1), targetCdf)

for xx in range (0, 228):
    for yy in range(0, 300):
        oldValue = imConvert[xx, yy, 2]
        newValue = targetCdf[oldValue]
        imConvert[xx, yy, 2] = newValue

specifiedImage = cv2.cvtColor(imConvert, cv2.COLOR_HSV2BGR)
cv2.imshow('4_2 original', im)
cv2.imshow('4_2 specified image', specifiedImage)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

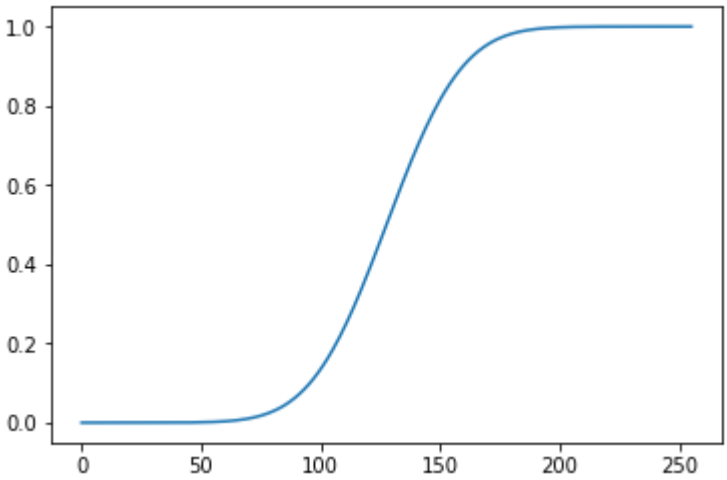
```
(228, 300)
```


Histogram of Image 4_2 Original V Values
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250


pdf of Image 4_2 Original V Values
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250

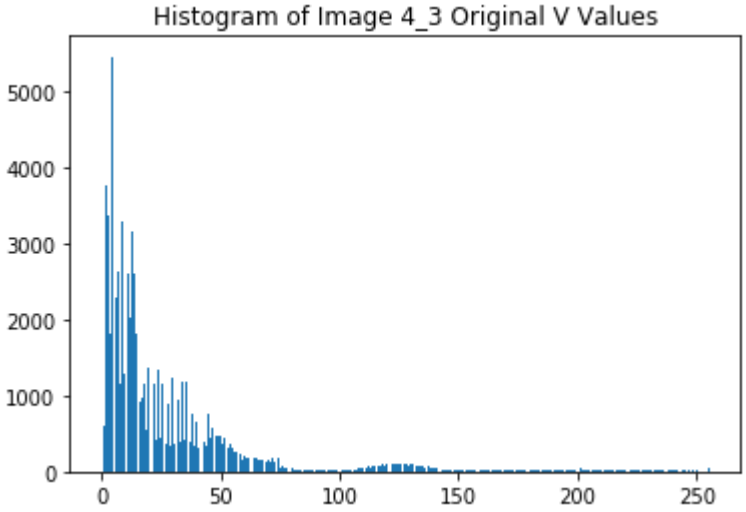

cdf of Image 4_2 Original V Values
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250

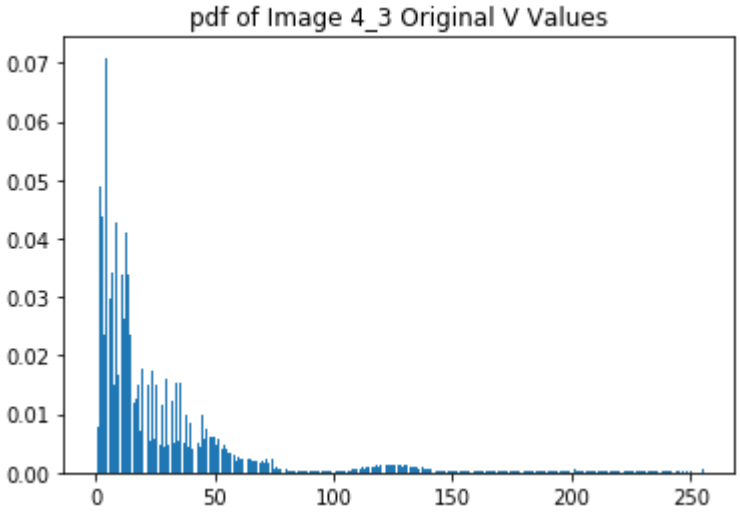

0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
50
100
150
200
250

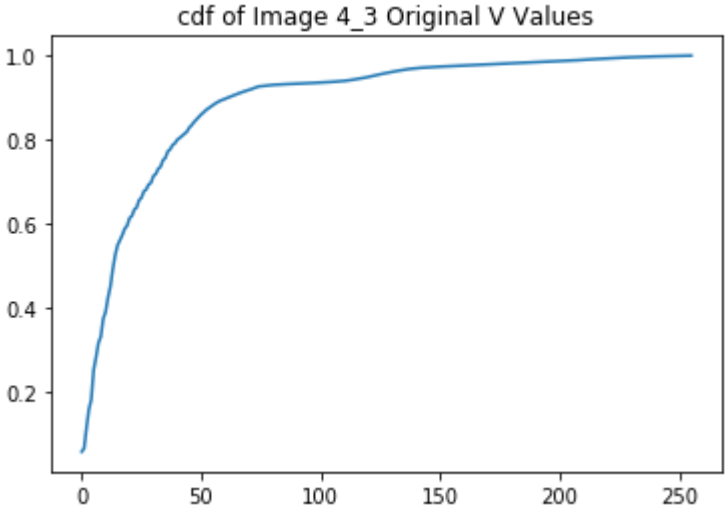

1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250

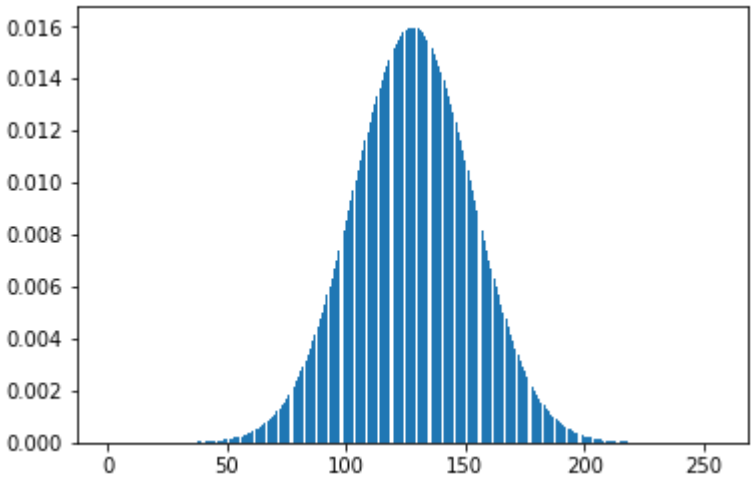
In [5]:
```
im = cv2.imread('4_3.jpg')
imConvert = cv2.cvtColor(im, cv2.COLOR_BGR2HSV)
originalCount = np.zeros(256)
counter = 0

print(imConvert[:, :, 2].shape)

for x in range(0, 240):
    for y in range(0, 320):
        counter = counter + 1
        value = imConvert[:, :, 2][x,y]
        originalCount[value] = originalCount[value] + 1

plt.title('Histogram of Image 4_3 Original V Values')
plt.bar(np.arange(0,256,1), originalCount)
plt.show()

plt.title('pdf of Image 4_3 Original V Values')
pdf = originalCount / (240 * 320)
plt.bar(np.arange(0,256,1), pdf)
plt.show()

cdf = np.cumsum(pdf)
plt.title('cdf of Image 4_3 Original V Values')
plt.plot(np.arange(0, 256, 1), cdf)
plt.show()

#create normal distribution
x = np.arange(0, 256, 1)
y = (1 / (np.sqrt(np.pi * 2 * 25**2))) * np.exp(-((x - 128)**2) / (2 * 25**2))
plt.bar(x, y)
plt.show()
normalCdf = np.cumsum(y)
plt.plot(normalCdf)
plt.show()

targetCdf = np.uint8(normalCdf * 255)
plt.bar(np.arange(0, 256, 1), targetCdf)

for xx in range (0, 240):
    for yy in range(0, 320):
        oldValue = imConvert[xx, yy, 2]
        newValue = targetCdf[oldValue]
        imConvert[xx, yy, 2] = newValue

specifiedImage = cv2.cvtColor(imConvert, cv2.COLOR_HSV2BGR)
cv2.imshow('4_3 original', im)
cv2.imshow('4_3 specified image', specifiedImage)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

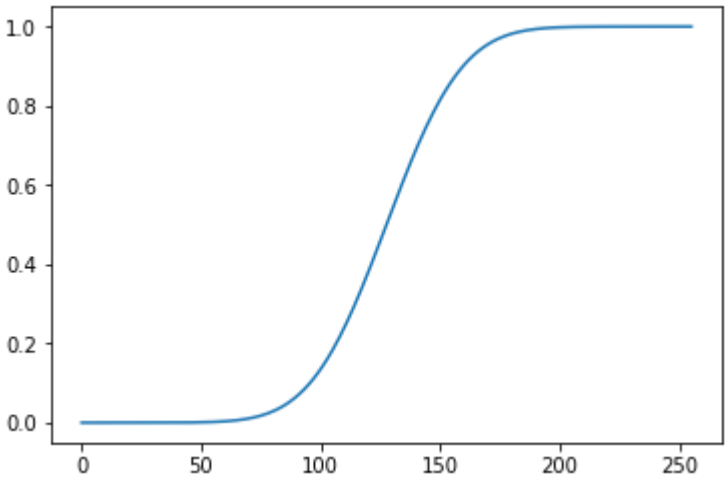
```
(240, 320)
```

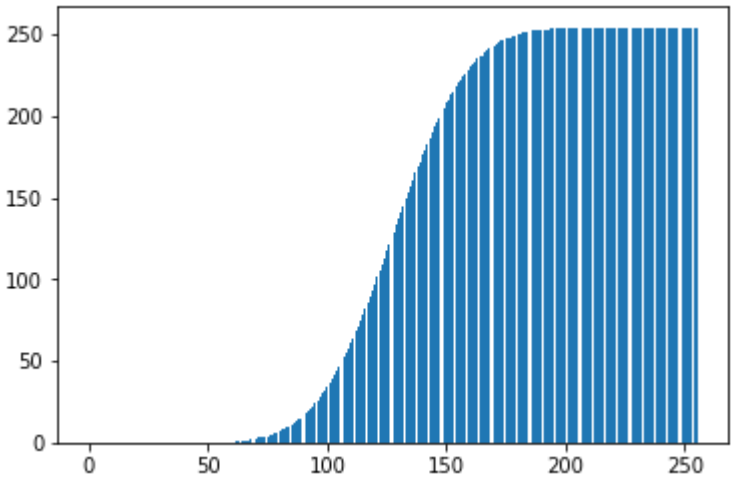

Histogram of Image 4_3 Original V Values
5000
4000
3000
2000
1000
0
0
50
100
150
200
250


pdf of Image 4_3 Original V Values
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
50
100
150
200
250


cdf of Image 4_3 Original V Values
1.0
0.8
0.6
0.4
0.2
0
50
100
150
200
250


0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
50
100
150
200
250


1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250


250
200
150
100
50
0
0
50
100
150
200
250

In [ ]: