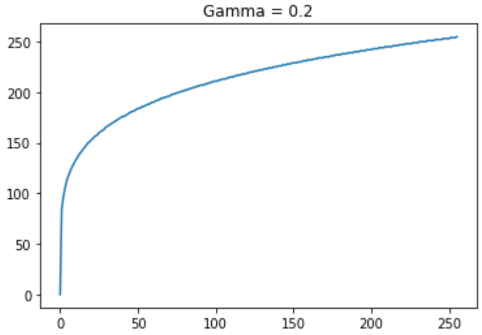
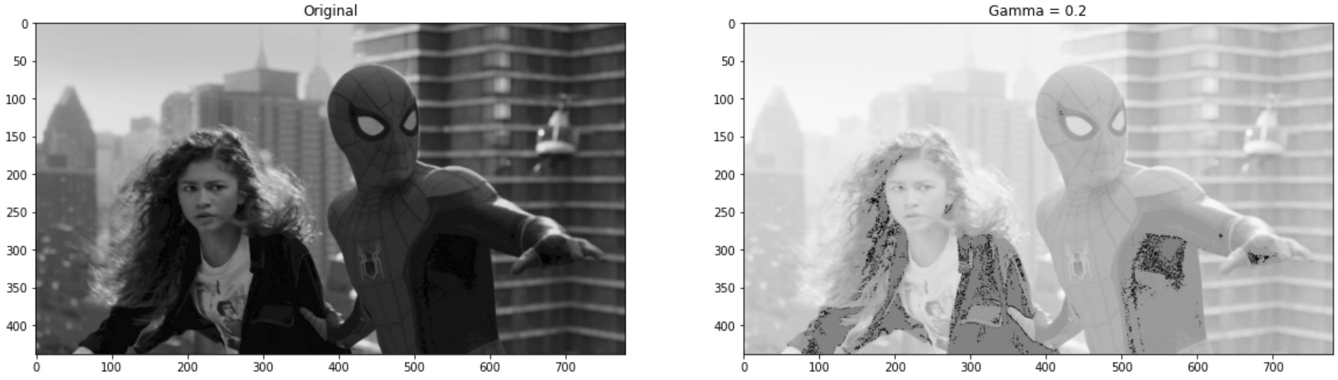
190071B

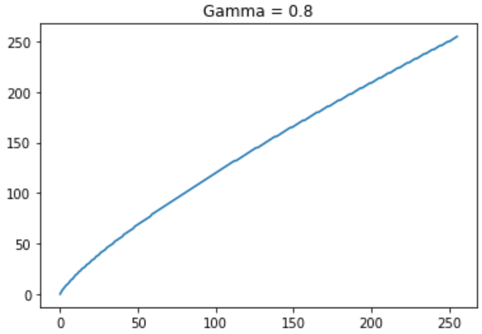
Bandara D.R.K.W.M.S.D.

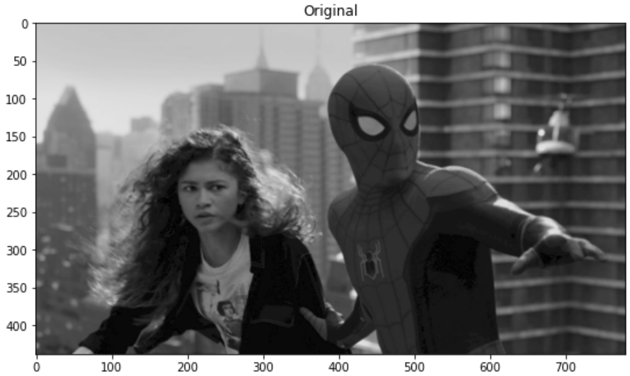
In [ ]:

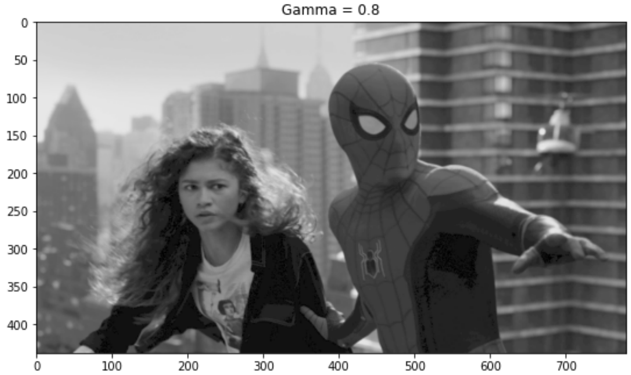
```
########################################################
#ex1
########################################################
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt

img = cv.imread('spider.png', cv.IMREAD_GRAYSCALE)
assert img is not None

gamma = [0.2,0.8,1.2,2]
for k in gamma:
    transform = np.array([(p/255)**k*255 for p in range(0,256)]).astype(np.uint8)#
    transformedImg = cv.LUT(img,transform)
    #g = t[f]#both are same this is the numpy way of doing that

    fig, ax = plt.subplots()
    ax.plot(transform)
    ax.set_title('Gamma = %s'%k)

    cv.namedWindow("Image", cv.WINDOW_AUTOSIZE)
    cv.imshow("Iamge", img)
    cv.waitKey(0)
    cv.destroyAllWindows()
    cv.imshow("Iamge", transformedImg)
    cv.waitKey(0)
    cv.destroyAllWindows()
    convertedImg1 = cv.cvtColor(img, cv.COLOR_BGR2RGB)
    convertedImg2 = cv.cvtColor(transformedImg, cv.COLOR_BGR2RGB)
    fig, ax = plt.subplots(1,2, figsize = (20,20))
    ax[0].imshow(convertedImg1)
    ax[1].imshow(convertedImg2)
    ax[0].set_title('Original')
    ax[1].set_title('Gamma = %s'%k)

plt.show()
```

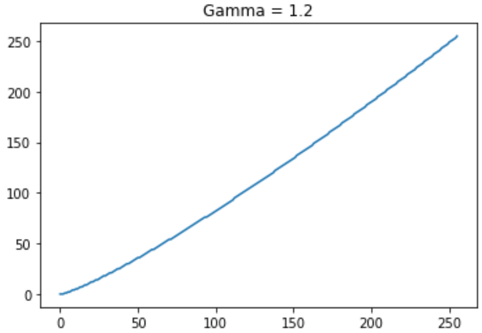


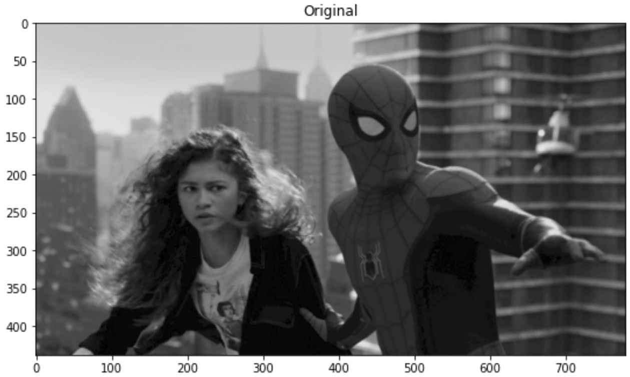


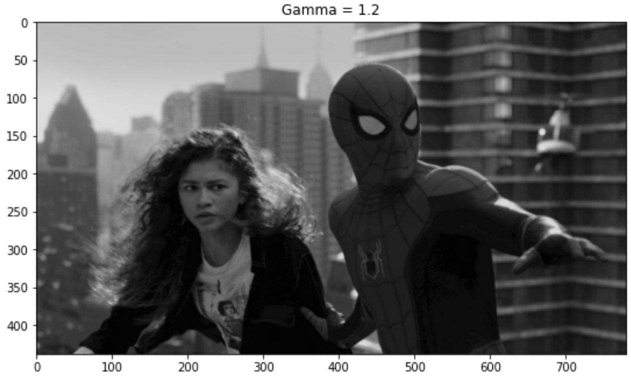

Gamma = 0.8
0
50
100
150
200
250
0
50
100
150
200
250

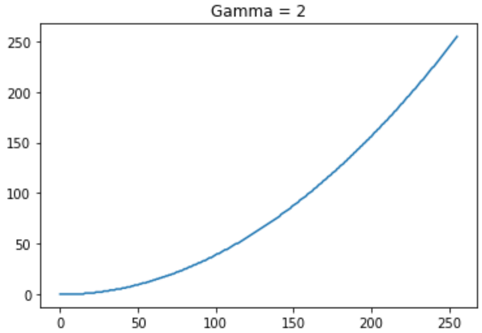

Original
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600
700

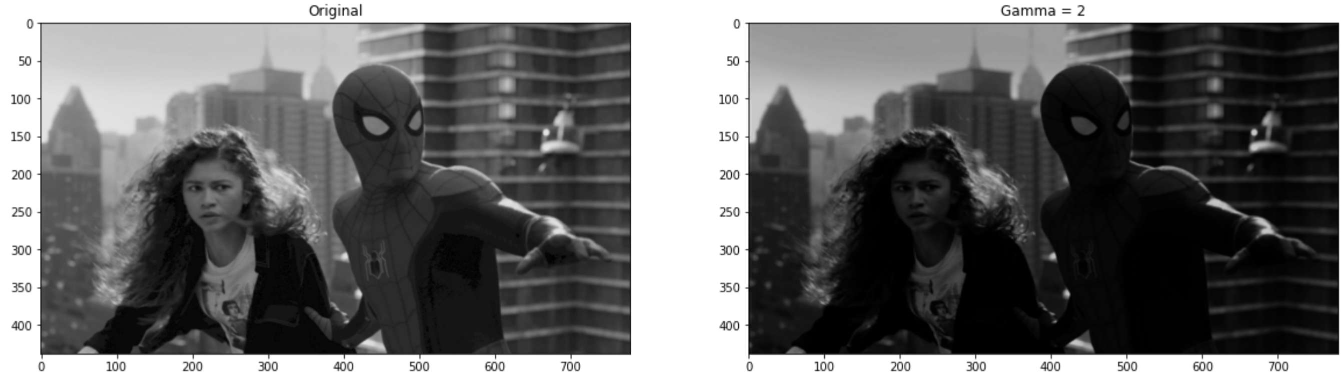

Gamma = 0.8
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600
700


Gamma = 1.2
0
50
100
150
200
250
0
50
100
150
200
250


Original
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600
700

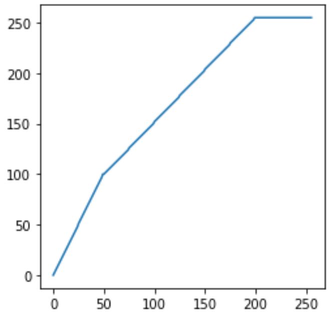

Gamma = 1.2
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600
700

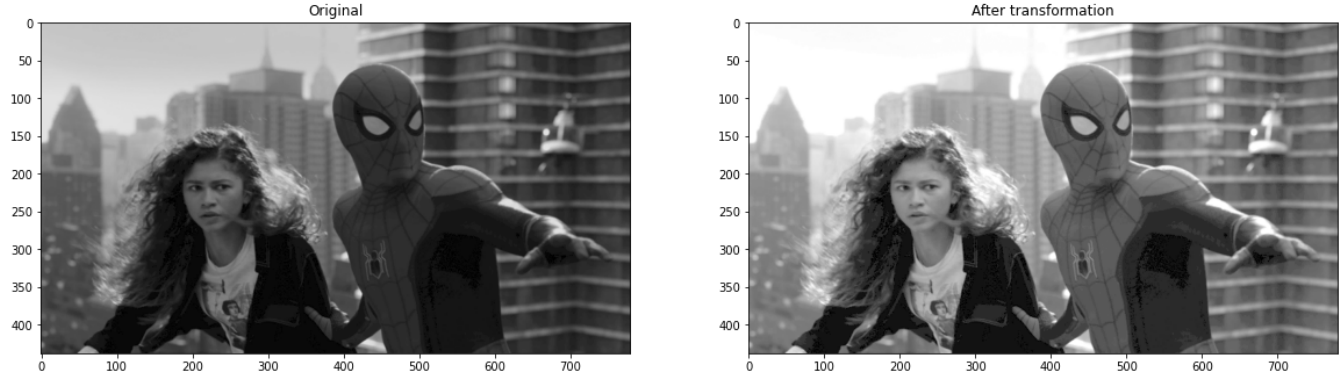

Gamma = 2
0
50
100
150
200
250
0
50
100
150
200
250



In [ ]:

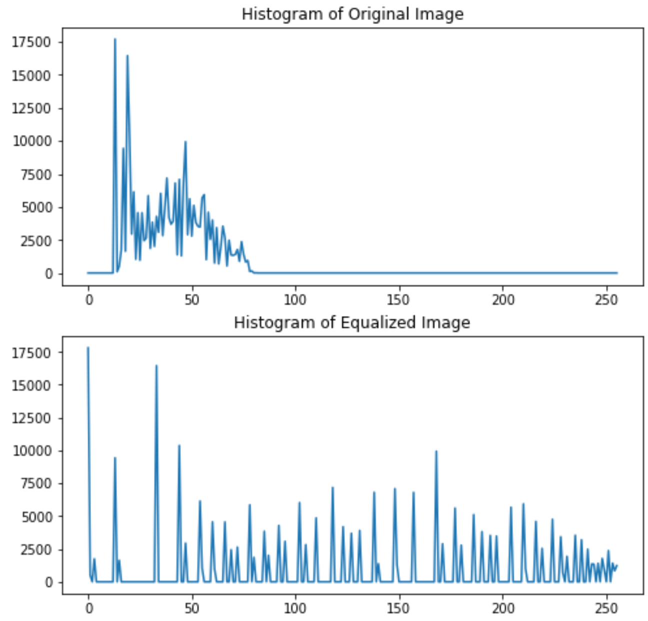
```
########################################################
#ex2
########################################################
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt

img = cv.imread('spider.png', cv.IMREAD_GRAYSCALE)
assert img is not None

transform1 = np.linspace(0,100,50)
transform2 = np.linspace(100,255,150)
transform3 = np.linspace(255,255,56)
transform = np.concatenate((transform1,transform2,transform3), axis = 0).astype(np.uint8)
transformedImg = cv.LUT(img,transform)
#g = t[f]#both are same this is the numpy way of doing that

fig, ax = plt.subplots()
ax.plot(transform)
ax.set_aspect('equal')
assert len(transform)==256
g = cv.LUT(img,transform)
cv.namedWindow("Image", cv.WINDOW_AUTOSIZE)
cv.imshow("Iamge", img)
cv.waitKey(0)
cv.destroyAllWindows()
cv.imshow("Iamge", transformedImg)
cv.waitKey(0)
cv.destroyAllWindows()
im = cv.cvtColor(img, cv.COLOR_BGR2RGB)
im2 = cv.cvtColor(transformedImg, cv.COLOR_BGR2RGB)
fig, ax = plt.subplots(1,2, figsize = (20,20))
ax[0].imshow(im)
ax[1].imshow(im2)
ax[0].set_title('Original')
ax[1].set_title('After transformation')
plt.show()
```



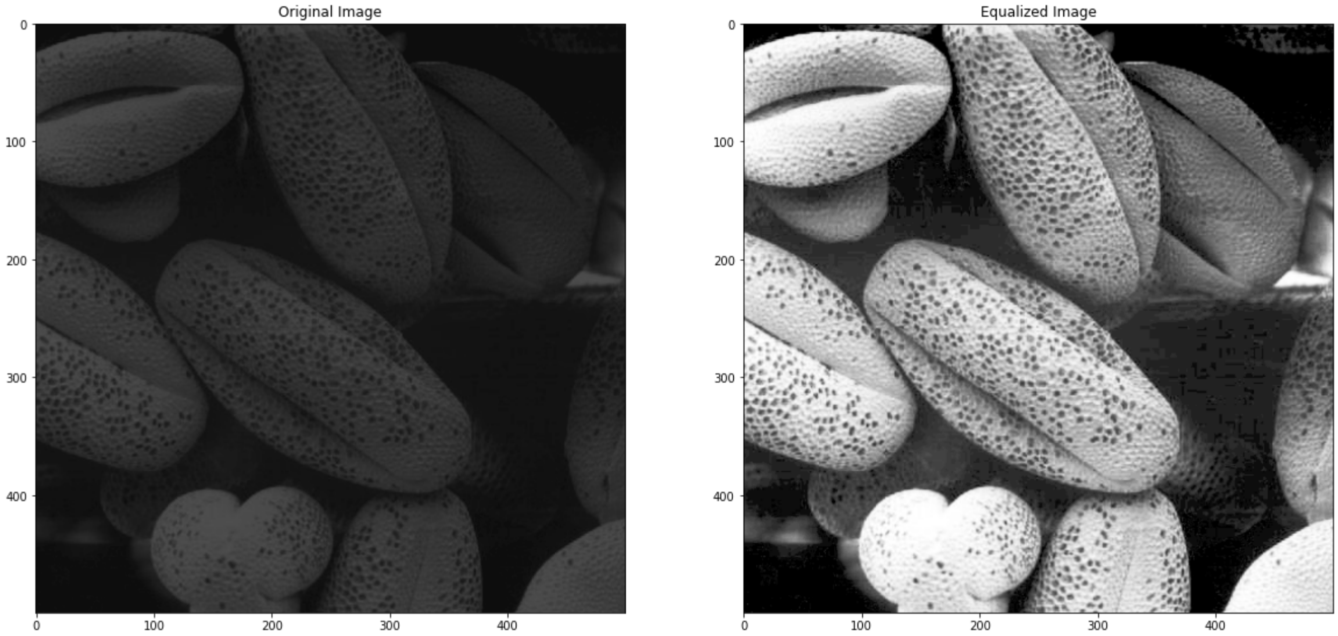


In [ ]:

```
###########################################################
#ex3
###########################################################
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt
img = cv.imread('shells.tif', cv.IMREAD_GRAYSCALE)
assert img is not None

#hist,bins = np.histogram(im,)
hist_img = cv.calcHist([img],[0],None,[256],[0,256])
equalizedImg = cv.equalizeHist(img)
hist_equalizedImg = cv.calcHist([equalizedImg],[0],None,[256],[0,256])
fig, ax = plt.subplots(2,1,figsize = (8,8))
ax[0].plot(hist_img)
ax[1].plot(hist_equalizedImg)
ax[0].set_title('Histogram of Original Image')
ax[1].set_title('Histogram of Equalized Image')
im = cv.cvtColor(img, cv.COLOR_BGR2RGB)
im2 = cv.cvtColor(equalizedImg, cv.COLOR_BGR2RGB)
fig, ax = plt.subplots(1,2, figsize = (20,20))
ax[0].imshow(im)
ax[1].imshow(im2)
ax[0].set_title('Original Image')
ax[1].set_title('Equalized Image')
plt.show()
```


Histogram of Original Image

Histogram of Equalized Image

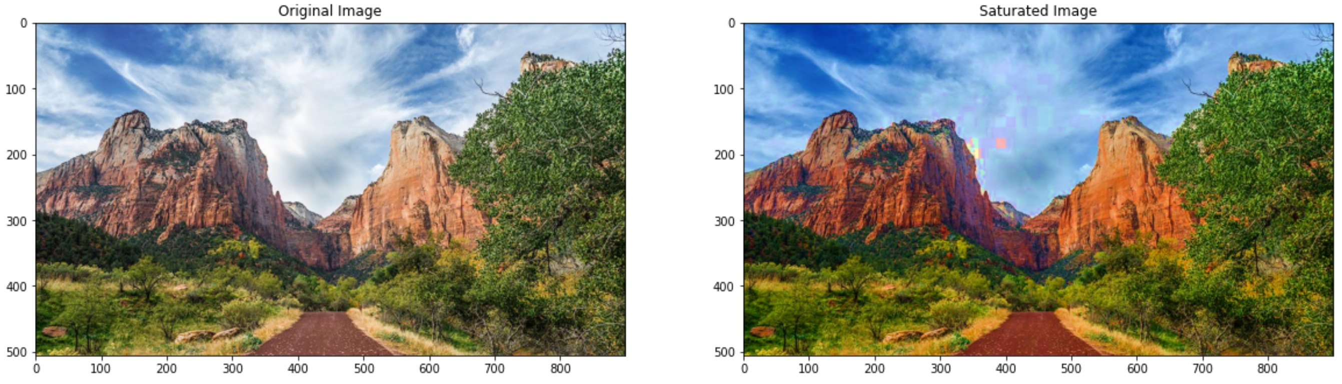

Original Image

Equalized Image

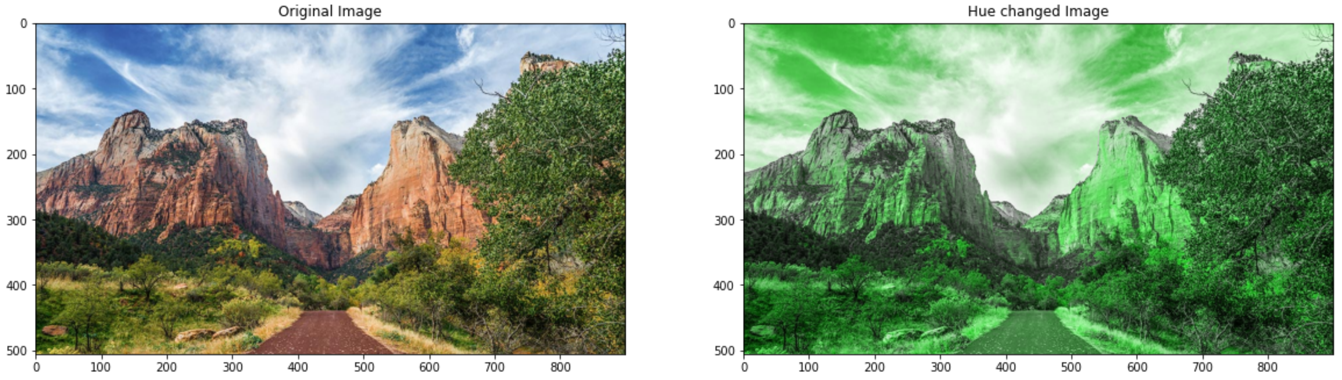
In [ ]:

```
###########################################################
#ex4
```

```python
########################################################
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt

img = cv.imread('zion_pass.jpg')
assert img is not None
########################################################
#Saturation increasing
########################################################
img_hsv = cv.cvtColor(img,cv.COLOR_BGR2HSV)
img_addedsat = cv.add(img_hsv[:,:,1],70)
img_hsv[:,:,1] = img_addedsat
img_sat = cv.cvtColor(img_hsv,cv.COLOR_HSV2BGR)
cv.namedWindow("Image", cv.WINDOW_AUTOSIZE)
cv.imshow("Iamge", img)
cv.waitKey(0)
cv.destroyAllWindows()
cv.imshow("Iamge", img_sat)
cv.waitKey(0)
cv.destroyAllWindows()
im = cv.cvtColor(img, cv.COLOR_BGR2RGB)
im2 = cv.cvtColor(img_sat, cv.COLOR_BGR2RGB)
fig, ax = plt.subplots(1,2, figsize = (20,20))
ax[0].imshow(im)
ax[1].imshow(im2)
ax[0].set_title('Original Image')
ax[1].set_title('Saturated Image')

########################################################
#hue increasing
########################################################
img_hsv = cv.cvtColor(img,cv.COLOR_BGR2HSV)
img_hsv[:,:,0] = 60
img_hue = cv.cvtColor(img_hsv,cv.COLOR_HSV2BGR)
cv.namedWindow("Image", cv.WINDOW_AUTOSIZE)
cv.imshow("Iamge", img)
cv.waitKey(0)
cv.destroyAllWindows()
cv.imshow("Iamge", img_hue)
cv.waitKey(0)
cv.destroyAllWindows()
im = cv.cvtColor(img, cv.COLOR_BGR2RGB)
im2 = cv.cvtColor(img_hue, cv.COLOR_BGR2RGB)
fig, ax = plt.subplots(1,2, figsize = (20,20))
ax[0].imshow(im)
ax[1].imshow(im2)
ax[0].set_title('Original Image')
ax[1].set_title('Hue changed Image')



plt.show()
```


Original Image

Saturated Image


Original Image

Hue changed Image

```
In [ ]:
```