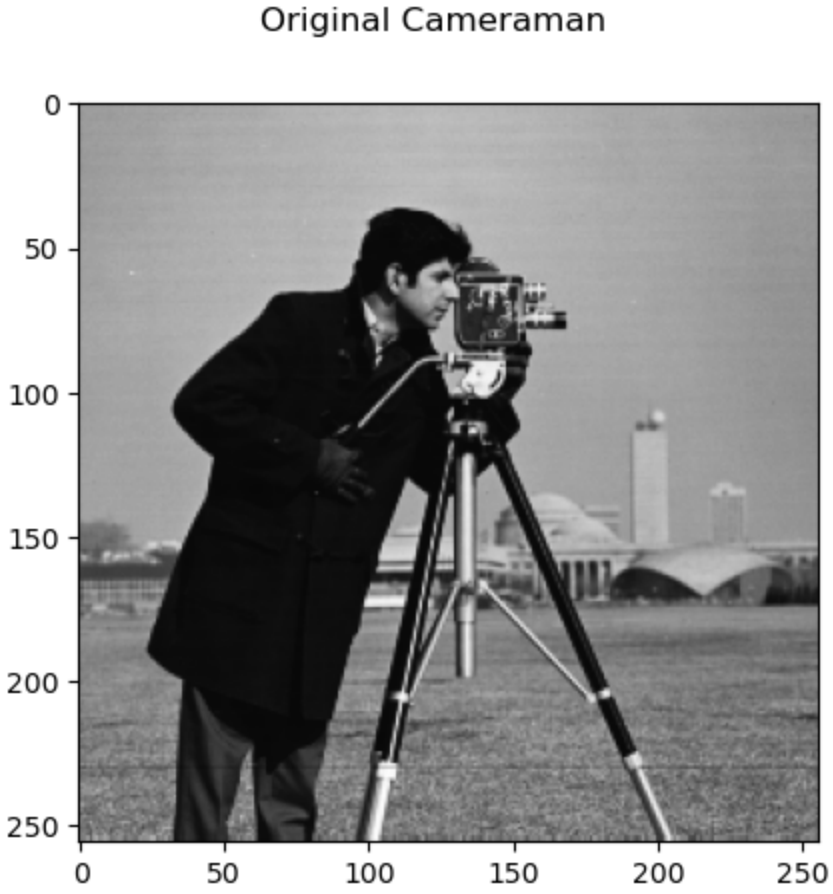
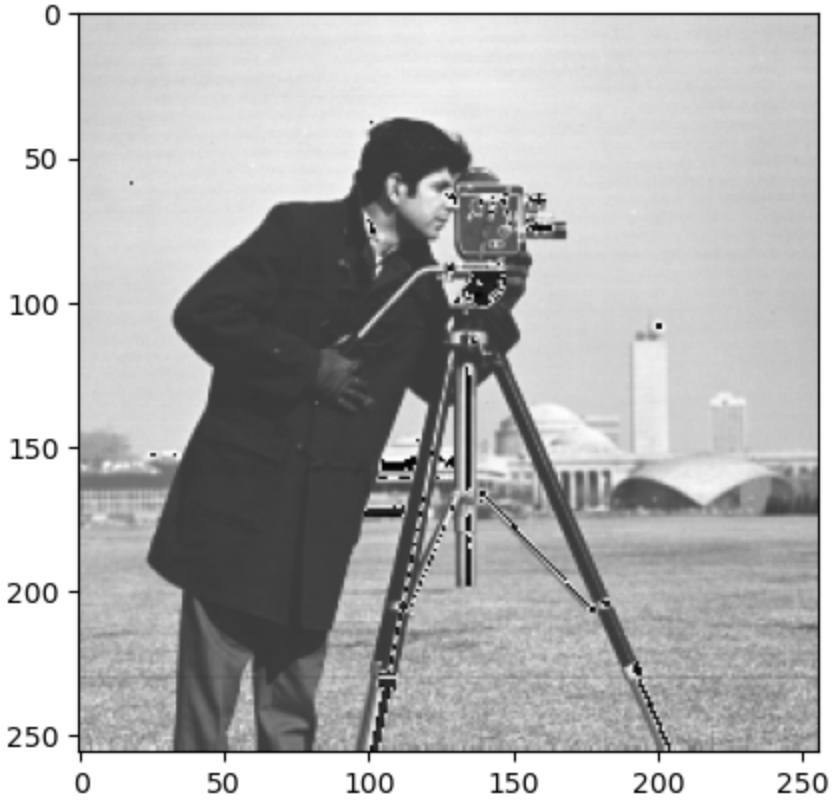
In [23]:
```
import numpy as np
from scipy import misc
import matplotlib.pyplot as plt
import copy
from imageio import imread
%matplotlib inline

# Implement this function
def imadd(pic,brightness):
     #Add brightness to each pixel
    for i in range(len(pic)):
        for j in range(len(pic[i])):
            pic[i][j] = brightness + pic[i][j]
    return pic

    #return pic+brightness

# Read the image
cameraman_origin = imread('cameraman.tif')
eco_origin = imread('eco.tif')
# Create a copy of the origina image for us to manipulate
cameraman_bright_50  = copy.deepcopy(cameraman_origin)
cameraman_bright_300 = copy.deepcopy(cameraman_origin)
eco_bright_50        = copy.deepcopy(eco_origin)
eco_bright_300       = copy.deepcopy(eco_origin)

# Call imadd to perform enhancement
cameraman_bright_50  = imadd(cameraman_bright_50,50)
cameraman_bright_300 = imadd(cameraman_bright_300,300)

print ("the dynamic range for cameraman_origin is", len(np.unique(cameraman_or
print ("the dynamic range for cameraman enhanced by 50 is", len(np.unique(came
print ("the dynamic range for cameraman enhanced by 300 is", len(np.unique(cam

print ("the dynamic range for eco origin is", len(np.unique(eco_origin)))
print ("the dynamic range for eco enhanced by 30 is", len(np.unique(eco_bright
print ("the dynamic range for eco enhanced by 500 is", len(np.unique(eco_brigh


eco_bright_50 = imadd(eco_bright_50,50)
eco_bright_300= imadd(eco_bright_300,300)




print (type(cameraman_origin))
# Show the results
fig_cam_origin = plt.figure(1)
fig_cam_origin.suptitle('Original Cameraman')
plt.imshow(cameraman_origin,cmap='gray',vmin = 0, vmax = 255)
fig_cam_bright = plt.figure(2)
fig_cam_bright.suptitle('Brightened Cameraman by 50')
plt.imshow(cameraman_bright_50,cmap='gray',vmin = 0, vmax = 255)
plt.show()
fig_cam_bright = plt.figure(3)
fig_cam_bright.suptitle('Brightened Cameraman by 300 (iterating through indivi
plt.imshow(cameraman_bright_300,cmap='gray',vmin = 0, vmax = 255)
```

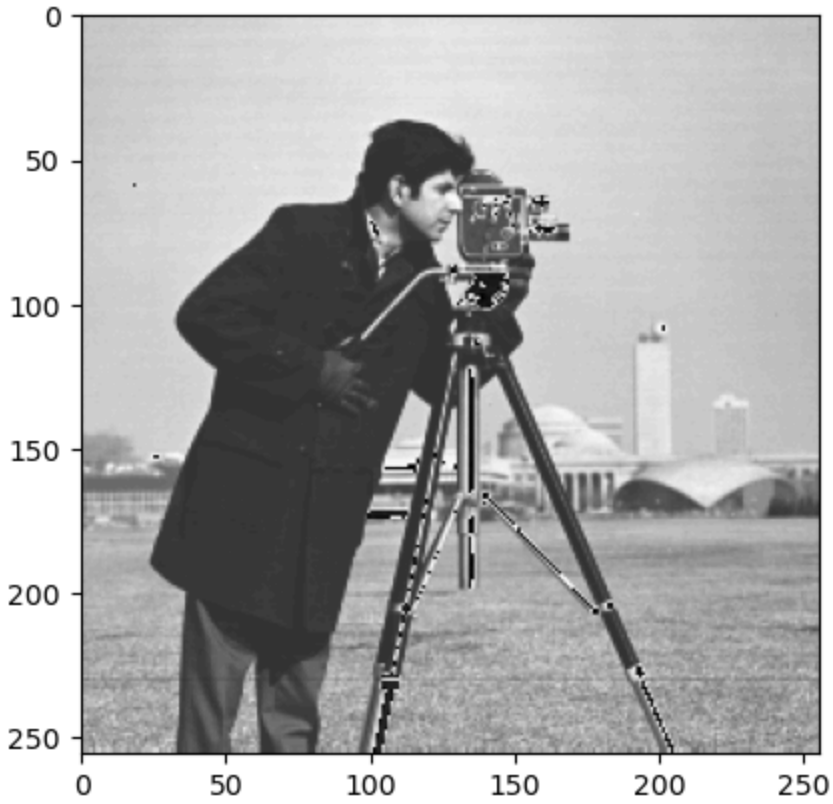
```
plt.show()
fig_cam_bright = plt.figure(4)
fig_cam_bright.suptitle('Brightened Cameraman by 300 (direct adding)')
plt.imshow(cameraman_origin+300,cmap='gray',vmin = 0, vmax = 255)
plt.show()
fig_cam_bright = plt.figure(4)
fig_cam_bright.suptitle('Original Eco')
plt.imshow(eco_origin,cmap='gray',vmin = 0, vmax = 65535)
plt.show()
fig_cam_bright = plt.figure(5)
fig_cam_bright.suptitle('Brightened Eco by 50')
plt.imshow(eco_bright_50,cmap='gray',vmin = 0, vmax = 65535)
plt.show()
fig_cam_bright = plt.figure(6)
fig_cam_bright.suptitle('Brightened Eco by 300')
plt.imshow(eco_bright_300,cmap='gray',vmin = 0, vmax = 65535)
plt.show()
```

```
C:\Users\Yicheng Zhou\AppData\Local\Temp\ipykernel_84832\1253908021.py:19: De
precationWarning: Starting with ImageIO v3 the behavior of this function will
switch to that of iio.v3.imread. To keep the current behavior (and make this
warning disappear) use `import imageio.v2 as imageio` or call `imageio.v2.imr
ead` directly.
  cameraman_origin = imread('cameraman.tif')
C:\Users\Yicheng Zhou\AppData\Local\Temp\ipykernel_84832\1253908021.py:20: De
precationWarning: Starting with ImageIO v3 the behavior of this function will
switch to that of iio.v3.imread. To keep the current behavior (and make this
warning disappear) use `import imageio.v2 as imageio` or call `imageio.v2.imr
ead` directly.
  eco_origin = imread('eco.tif')

the dynamic range for cameraman_origin is 247
the dynamic range for cameraman enhanced by 50 is 247
the dynamic range for cameraman enhanced by 300 is 247
the dynamic range for eco origin is 1748
the dynamic range for eco enhanced by 30 is 1748
the dynamic range for eco enhanced by 500 is 1748
<class 'imageio.core.util.Array'>
```

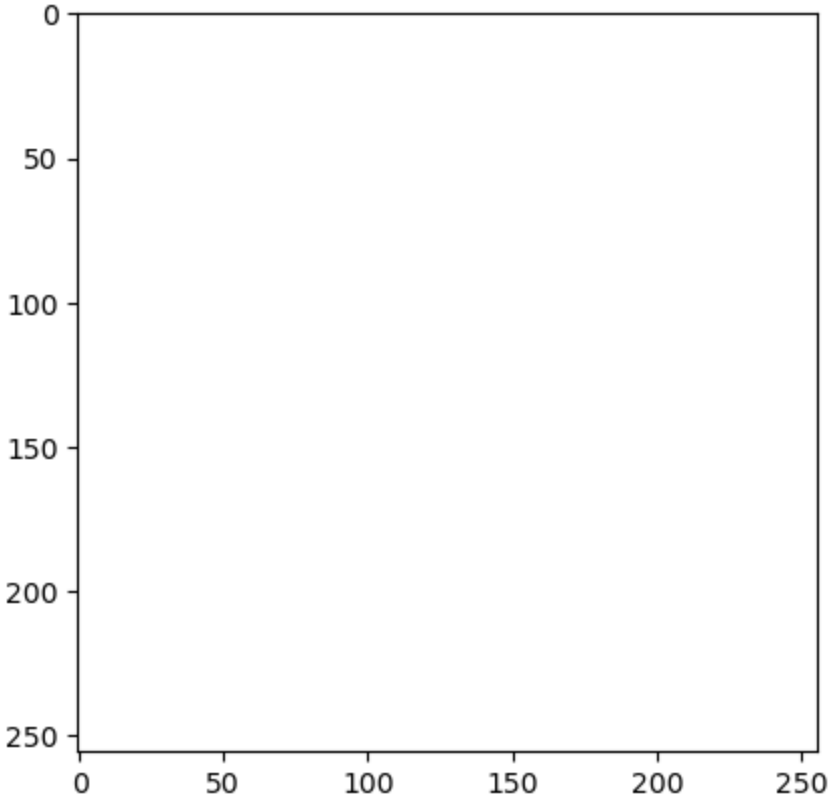

Original Cameraman
0
50
100
150
200
250
0
50
100
150
200
250

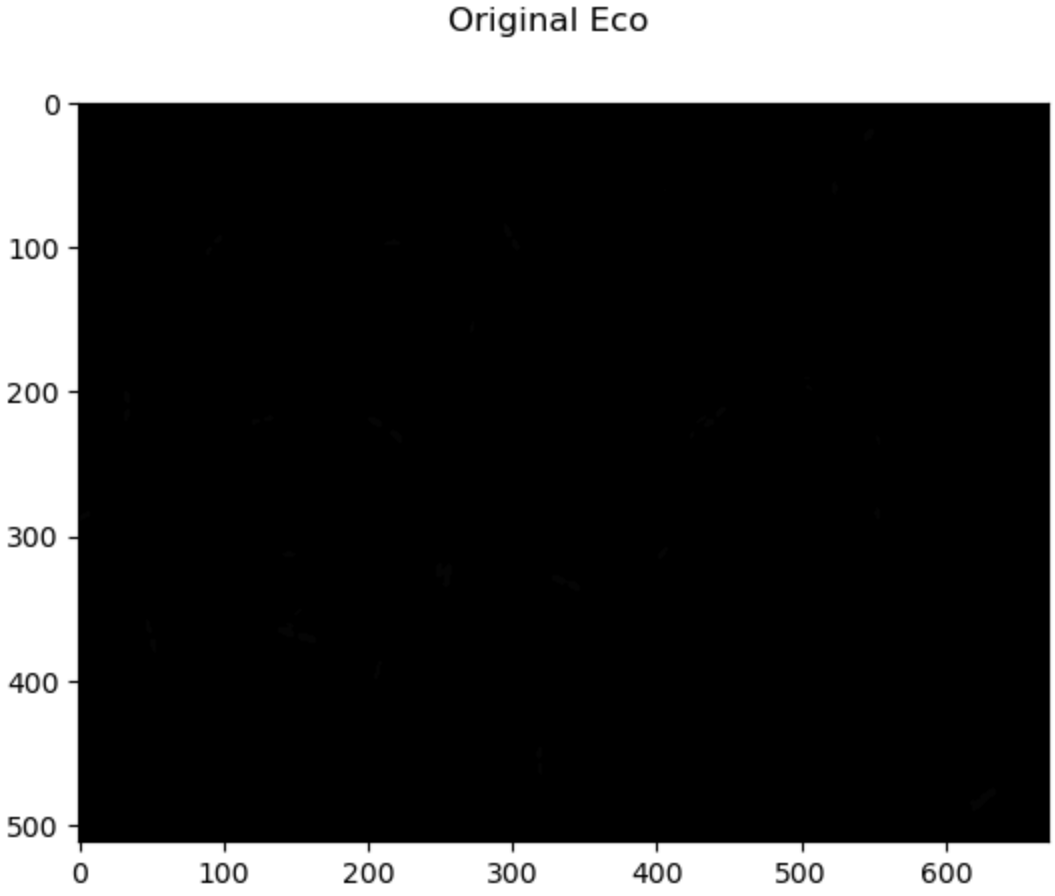

Brightened Cameraman by 50
0
50
100
150
200
250
0
50
100
150
200
250

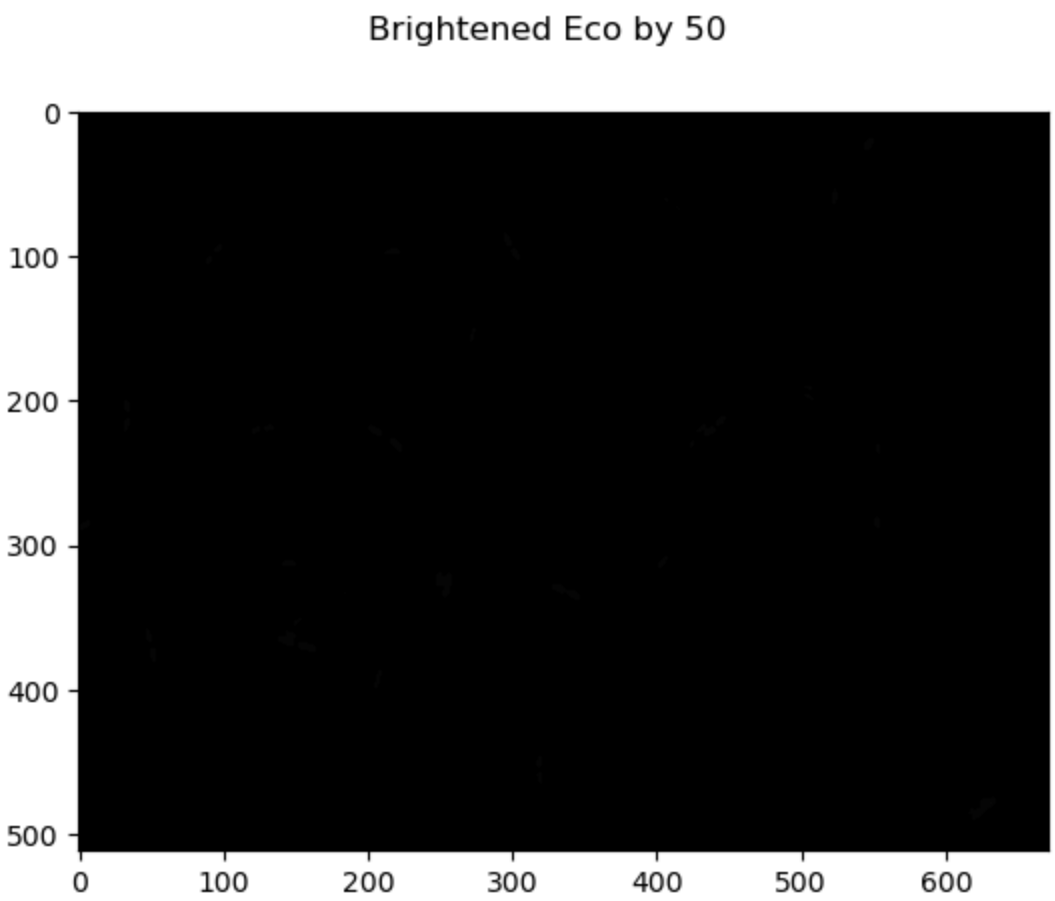
Brightened Cameraman by 300 (iterating through individual pixels)

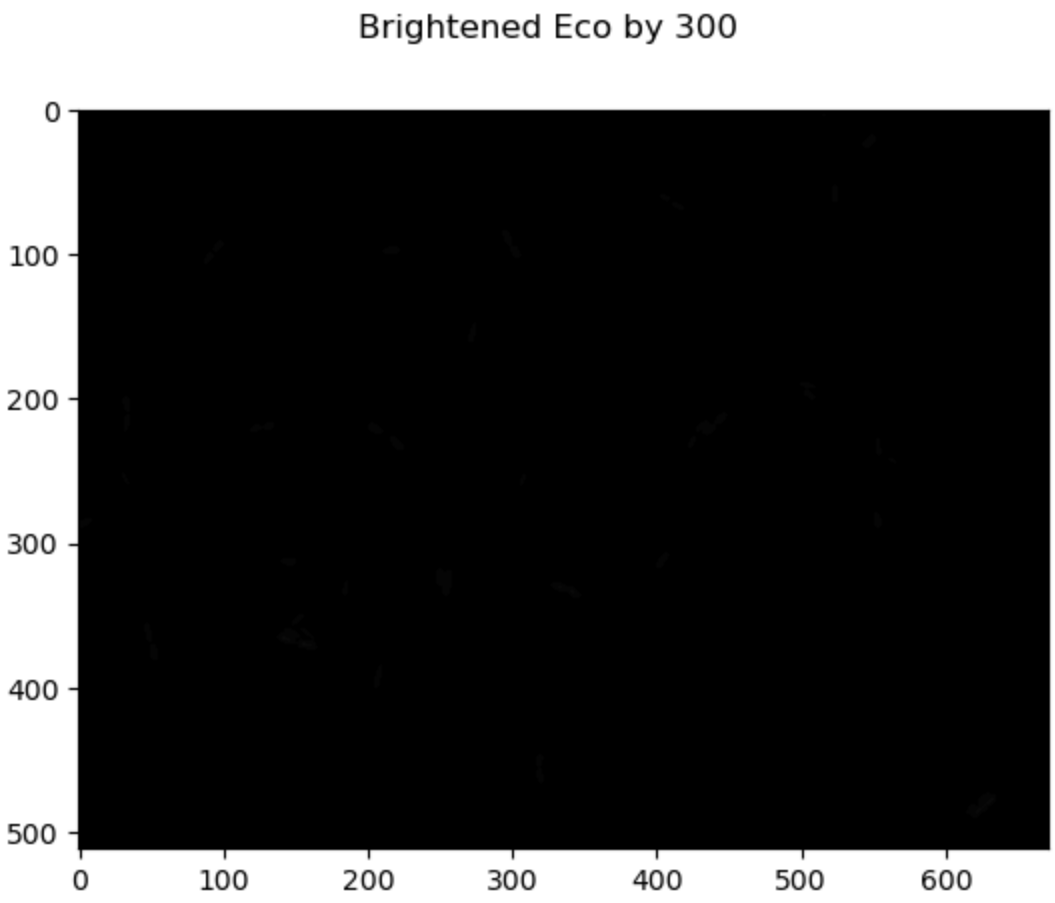


Brightened Cameraman by 300 (direct adding)




Original Eco
0
100
200
300
400
500
0
100
200
300
400
500
600


Brightened Eco by 50
0
100
200
300
400
500
0
100
200
300
400
500
600



Question:

1. the dynamic range for cameraman original and enhanced are 247, the dynamic range for eco original and enhanced are 1748

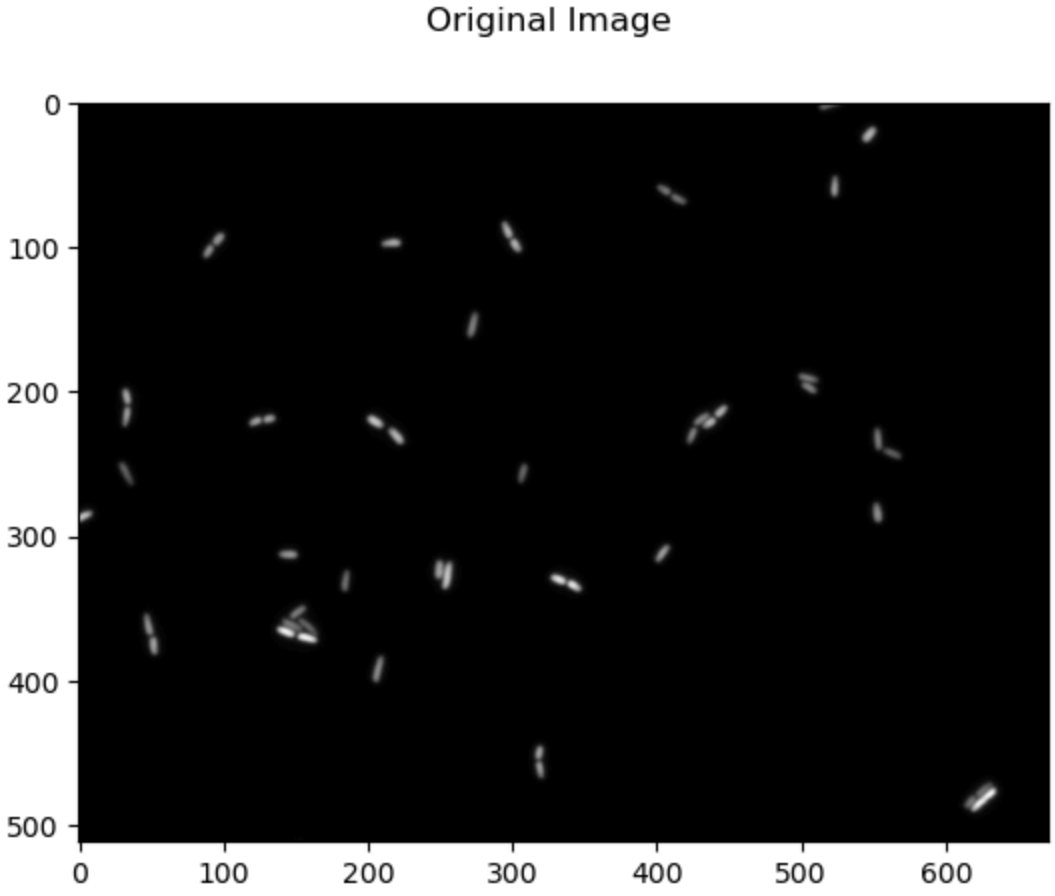
2. in the case of cameraman image, what would happen depends on how t he imadd is implemented. This is due to how python handels data type for array addition. If the algorithm is implemented by iterating thro ugh each individual pixels and performing the addition, the data type will remain uint8 and the value would overflow, thus the output would look like the same as if we increased brightness by 44. However, if w e cast the addition directly onto the entire array, python will chang e data type to allow values above 255 (persumably to uint16), than th e output image will be all white

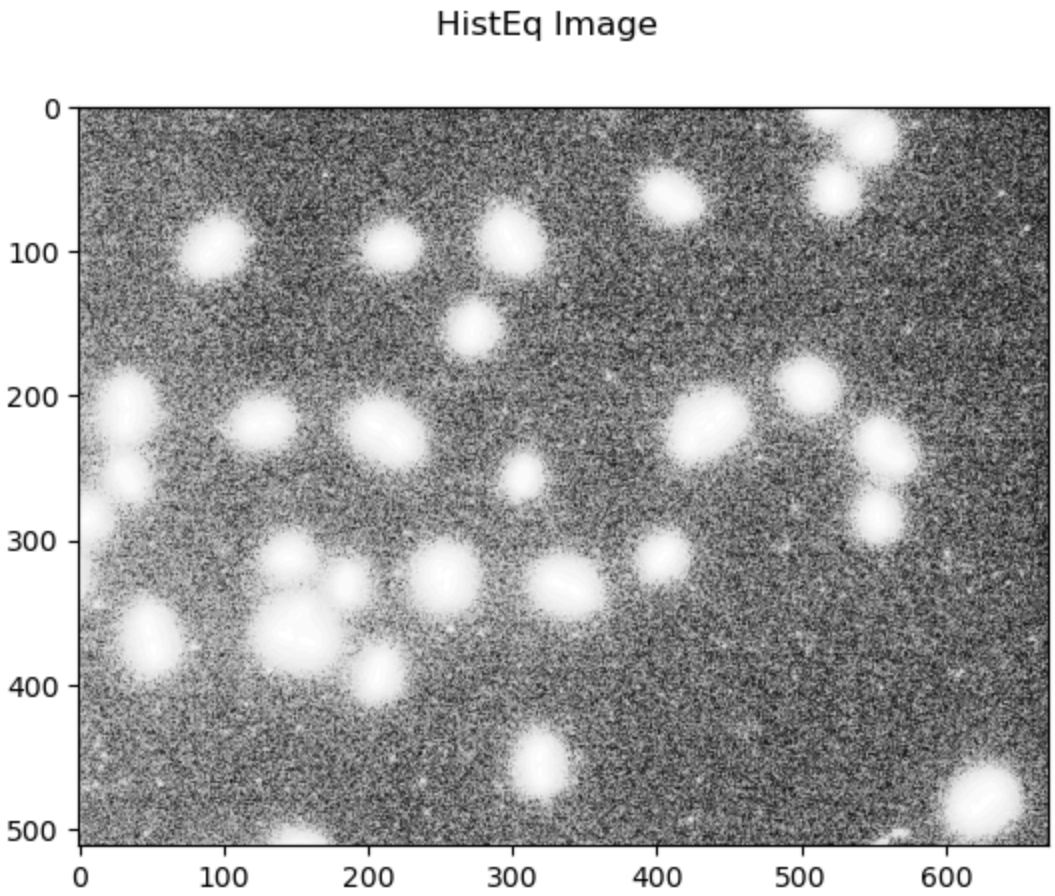
for eco, since the max is 65535, enhancing by 300 is not noticable

3. No, the problem with eco is its dynamic range (contrast) is too lo w, increasing brightness won't solve that

In [25]:
```
import matplotlib.pyplot as plt
import numpy as np
from scipy import misc
from skimage import exposure
import copy
%matplotlib inline


# Read the image
eco_origin = imread('eco.tif')

# Apply Histogram Equalization here!
eco_histeq = exposure.equalize_hist(eco_origin)

# Show the results
fig_cam_origin = plt.figure(1)
fig_cam_origin.suptitle('Original Image')
plt.imshow(eco_origin,cmap='gray')
fig_cam_bright = plt.figure(2)
fig_cam_bright.suptitle('HistEq Image')
plt.imshow(eco_histeq,cmap='gray')
plt.show()
```

```
C:\Users\Yicheng Zhou\AppData\Local\Temp\ipykernel_84832\2423867746.py:10: De
precationWarning: Starting with ImageIO v3 the behavior of this function will
switch to that of iio.v3.imread. To keep the current behavior (and make this
warning disappear) use `import imageio.v2 as imageio` or call `imageio.v2.imr
ead` directly.
  eco_origin = imread('eco.tif')
```


Original Image
0
100
200
300
400
500
0
100
200
300
400
500
600


HistEq Image
0
100
200
300
400
500
0
100
200
300
400
500
600

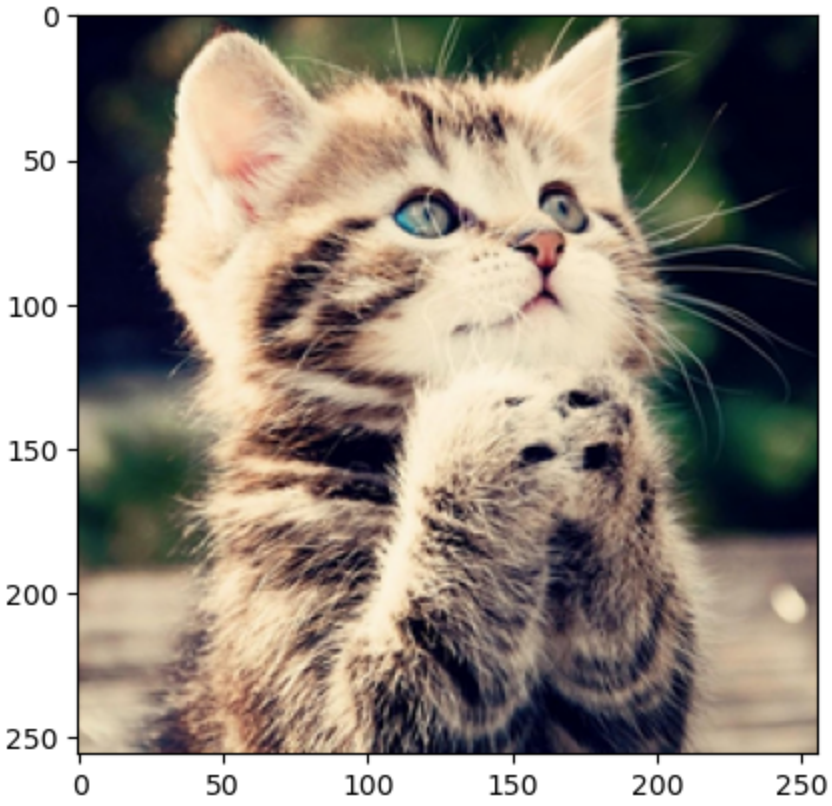
Question:

No, after the first histogram equalization, the histogram is already close to uniform, thus further attempts will only yield negligible ga in, so we can't improve result by repeating the equalization

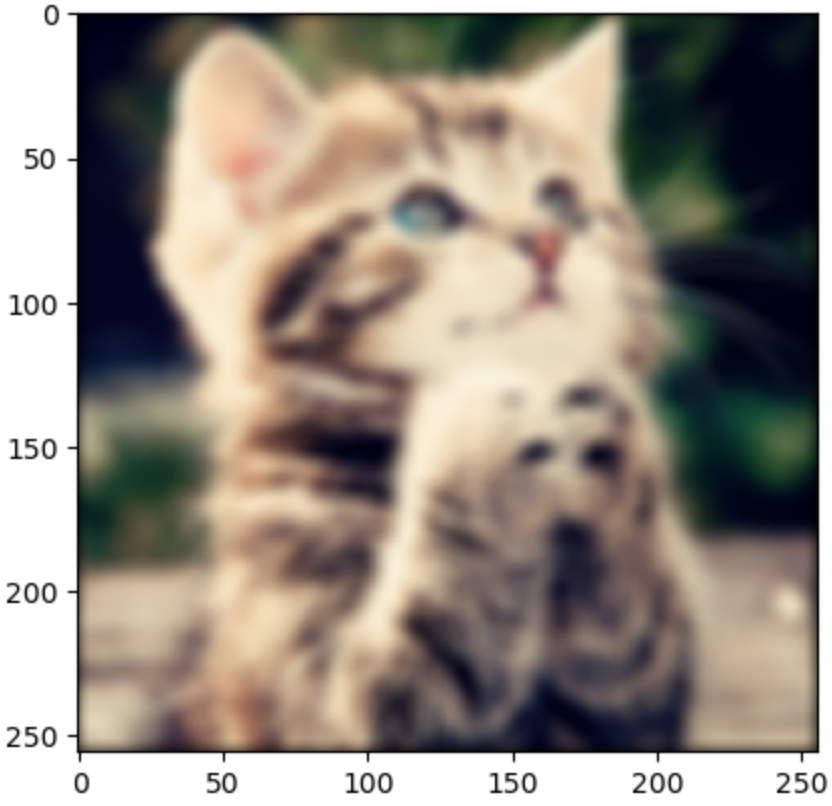
In [43]:
```
import numpy
from scipy import misc
from scipy import signal
import matplotlib.pyplot as plt

# Gaussian Kernel Following the Descriptiong:
# http://www.mathworks.com/help/images/ref/fspecial.html
def gengaussian(size=5,sigma=3.0):
    if size%2==0 or size<2:
        print('Size Not Valid')
        return None
    kernel = numpy.zeros((size,size))
    for x in range(size):
        for y in range(size):
            kernel[x][y] = numpy.exp(-((x-(size-1)/2)**2 \
                           +(y-(size-1)/2)**2)/(2*sigma**2))
    kernel = kernel / numpy.sum(kernel)
    return kernel

# Read Image and Display
kitten_origin = imread('kitten.png')
# Create a copy of the origina image for us to manipulate
kitten_blur = copy.deepcopy(kitten_origin)
# Generate Kernel
kernel = gengaussian(9)
# Apply Convolution Here!
#kitten_blur = np.zeros((len(kitten_origin),len(kitten_origin[0]),3))

for i in range(3):
    temp = signal.convolve2d(kitten_blur[:,:,i],kernel,mode='same')
    kitten_blur[:,:,i] = temp


# Display Results
fig_kitten_origin = plt.figure(1)
fig_kitten_origin.suptitle('Original Kitten.png', fontsize=14, fontweight='bol
plt.imshow(kitten_origin,vmin = 0, vmax = 255)
fig_kitten_blur = plt.figure(2)
fig_kitten_blur.suptitle('Blurred Kitten.png', fontsize=14, fontweight='bold')
plt.imshow(kitten_blur,vmin = 0, vmax = 255)
plt.show()
```

```
C:\Users\Yicheng Zhou\AppData\Local\Temp\ipykernel_84832\2422740831.py:21: De
precationWarning: Starting with ImageIO v3 the behavior of this function will
switch to that of iio.v3.imread. To keep the current behavior (and make this
warning disappear) use `import imageio.v2 as imageio` or call `imageio.v2.imr
ead` directly.
  kitten_origin = imread('kitten.png')
```

Original Kitten.png



Blurred Kitten.png



The sigma value is capped between 3 to 9, and in this range, the higher the sigma value the more blurry the image is