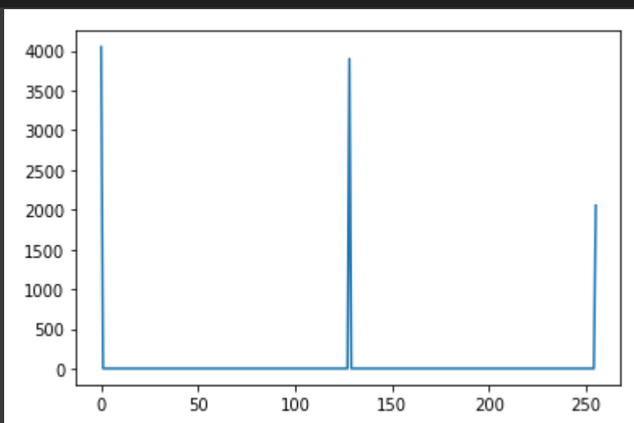
Rohan Anil More TE-EXTC 118A2065

Q1

```
import cv2
import numpy as np
from PIL import Image, ImageDraw
```

```
im = Image.new('RGB', (100, 100), (128, 128, 128))
draw = ImageDraw.Draw(im)
draw.rectangle((0, 0, 100, 50), fill=(0, 0, 0))
draw.ellipse((25, 25, 75, 75), fill=(255, 255, 255))
im1=np.array(im)
im
```

```
import cv2
from matplotlib import pyplot as plt
gs= cv2.cvtColor(im1, cv2.COLOR_BGR2GRAY)
hist = cv2.calcHist([gs],[0],None,[256],[0,256])
plt.plot(hist)
plt.show()
```



Q2

```
from google.colab import files
uploaded=files.upload()
```

Choose Files low.png

- **low.png**(image/png) - 4027 bytes, last modified: 6/25/2021 - 100% done

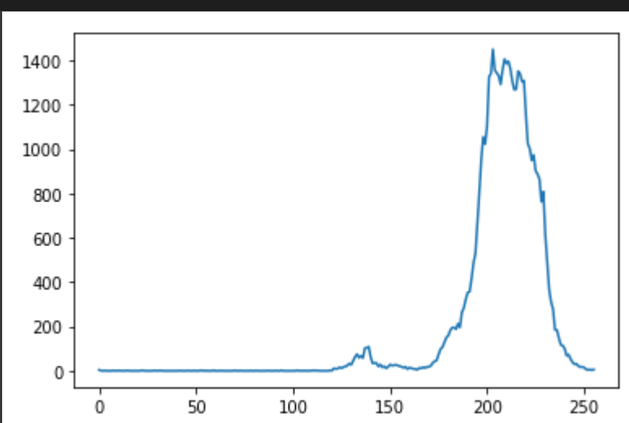
```
Saving low.png to low.png
```

```
image=cv2.imread("low.png")
```

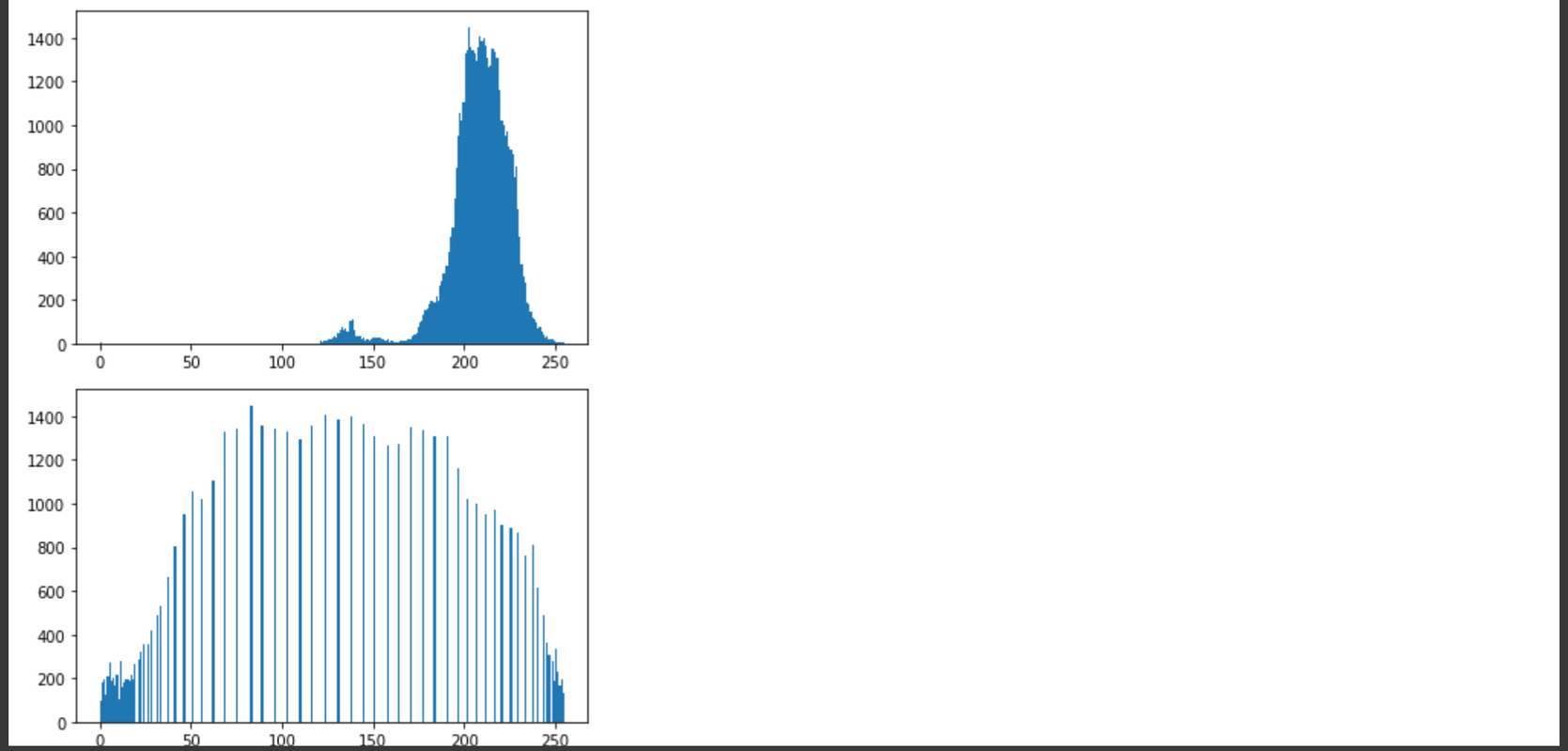
```
from google.colab.patches import cv2_imshow
cv2_imshow(image)
```

```
from matplotlib import pyplot as plt
height  = image.shape[0]
width = image.shape[1]
max_intensity = 255
img_gs= cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
cv2_imshow(img_gs)
```

```
histg = cv2.calcHist([img_gs],[0],None,[256],[0,256])
cum_hist = histg.copy()
plt.plot(histg)
plt.show()
```




```
equ = cv2.equalizeHist(img_gs)
plt.hist(img_gs.ravel(), 256, [0, 255])
plt.show()
plt.hist(equ.ravel(), 256, [0, 255])
plt.show()
```

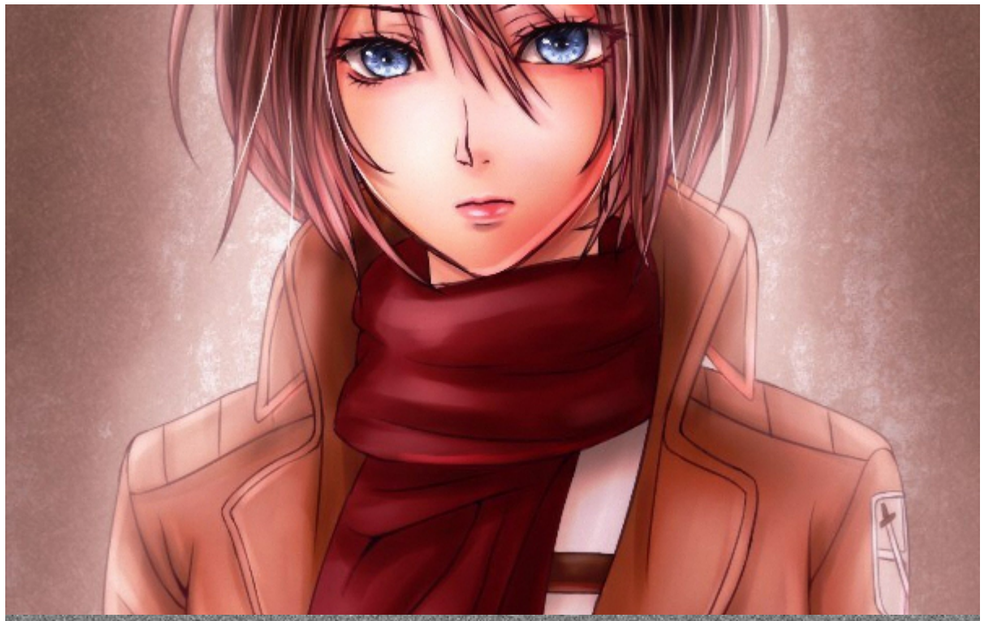


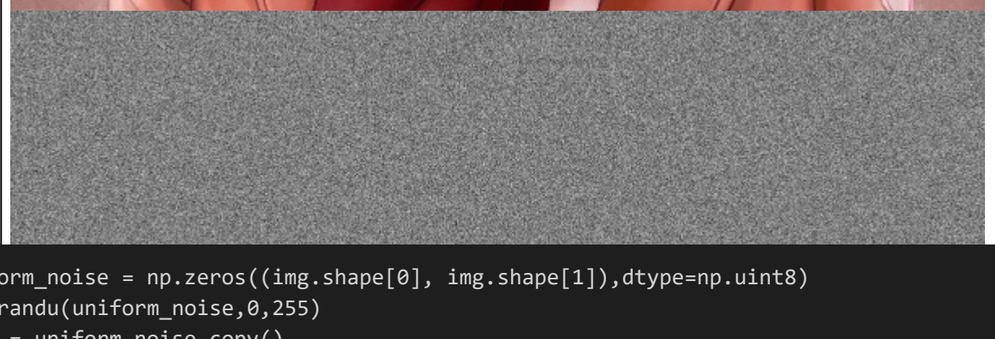
Q3

```
uploaded=files.upload()
```

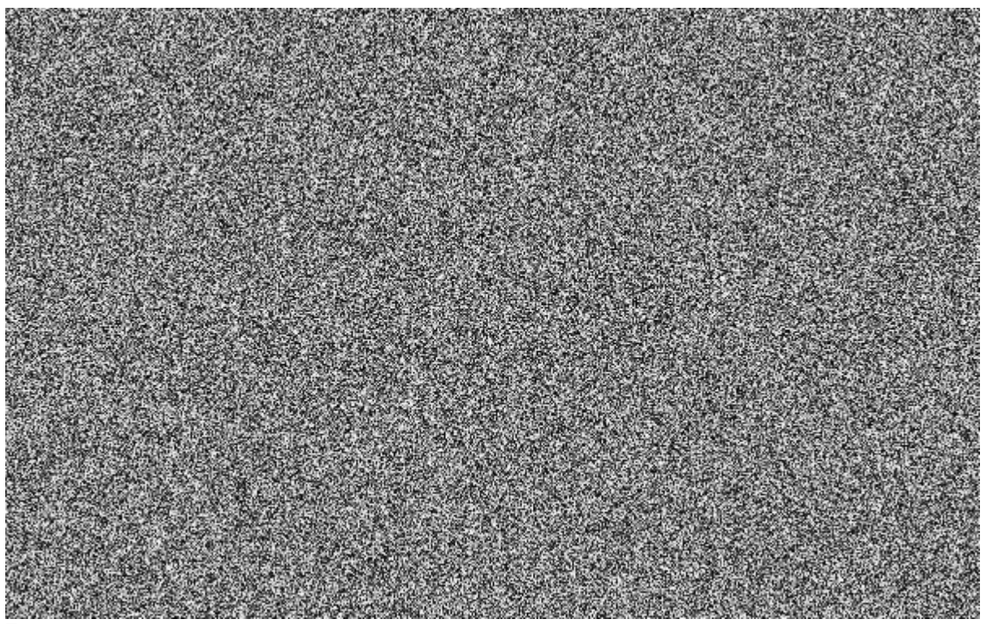
Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving 434487.jpg to 434487.jpg

```
img=cv2.imread("434487.jpg")
img = cv2.resize(img,(600,376))
cv2_imshow(img)
img=cv2.imread("434487.jpg",cv2.IMREAD_GRAYSCALE)
img = cv2.resize(img,(600,376))
img1 = np.zeros((img.shape[0], img.shape[1]),dtype=np.uint8)
cv2.randn(img1, 128, 20)
cv2_imshow(img1)
cv2.imwrite("Gaussian random noise.jpg",img1)
```





```
uniform_noise = np.zeros((img.shape[0], img.shape[1]),dtype=np.uint8)
cv2.randu(uniform_noise,0,255)
img2 = uniform_noise.copy()
ret,imgn2= cv2.threshold(uniform_noise,250,255,cv2.THRESH_BINARY)
cv2_imshow(img2)
cv2.imwrite("Impuls noise.jpg",img2)
```



```
True
```

Average filtering on an image with gaussian noise

```
img1 = (img1*0.1).astype(np.uint8)
imgn1 = cv2.add(img,img1)
processed_image1 = cv2.blur(imgn1,(5,5))
st1=np.hstack((imgn1,processed_image1))
cv2_imshow(st1)
```

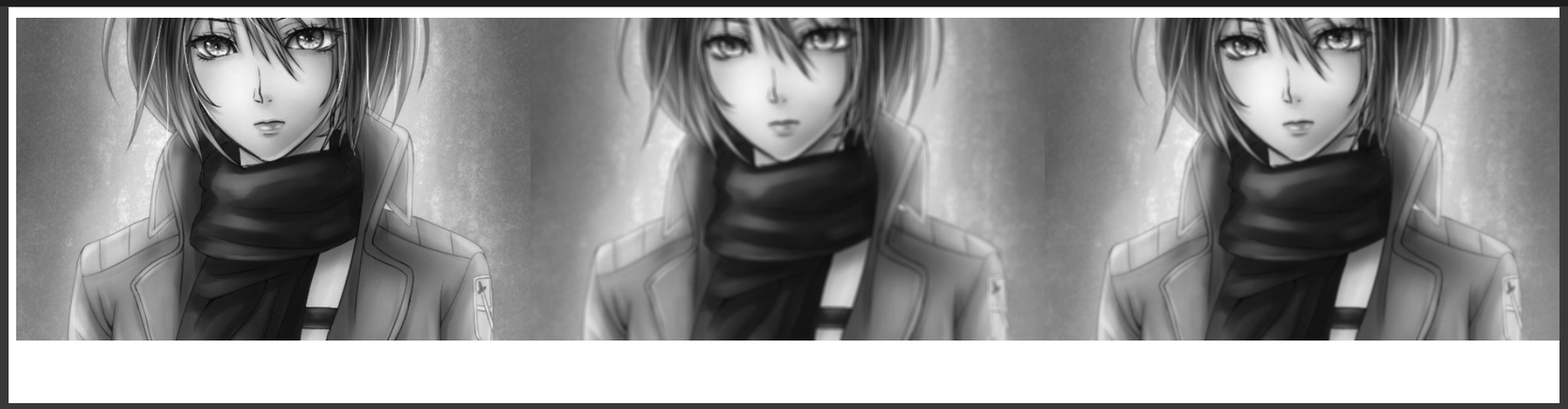
Average filtering on an image with impulse noise

```
img2 = (img2*0.5).astype(np.uint8)
imgn2 = cv2.add(img,img2)

processed_image2 = cv2.blur(imgn2,(10,5))
st2=np.hstack((imgn2,processed_image2))
cv2_imshow(st2)
```

Gaussian filtering on an image with gaussian noise

```
image_gaussian_processed = cv2.GaussianBlur(imgn1,(5,5),1)
gst1=np.hstack((imgn1,processed_image1,image_gaussian_processed))
cv2_imshow(gst1)
```

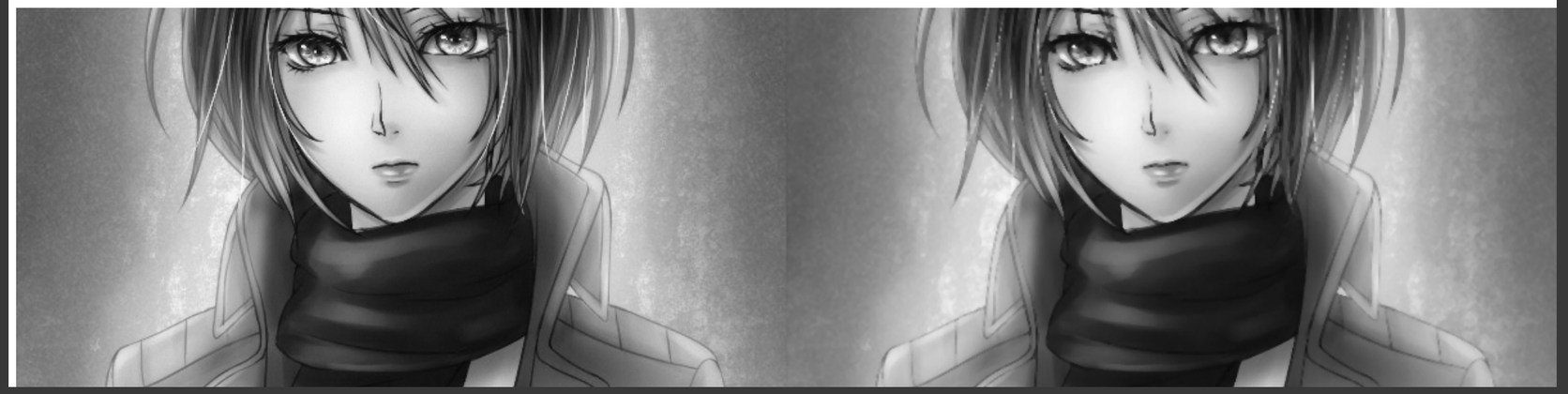


Gaussian filtering on an image with impulse noise

```
image_impulse_processed = cv2.GaussianBlur(imgn2,(5,5),1)
gst2=np.hstack((imgn1,processed_image2,image_impulse_processed))
cv2_imshow(gst2)
```

Median filtering on an image with gaussian noise

```
blurred1 = cv2.medianBlur(imgn1, 3)
mst1=np.hstack((imgn1,blurred1))
cv2_imshow(mst1)
cv2.imwrite("Median filter - Gaussian noise.jpg",blurred1)
```



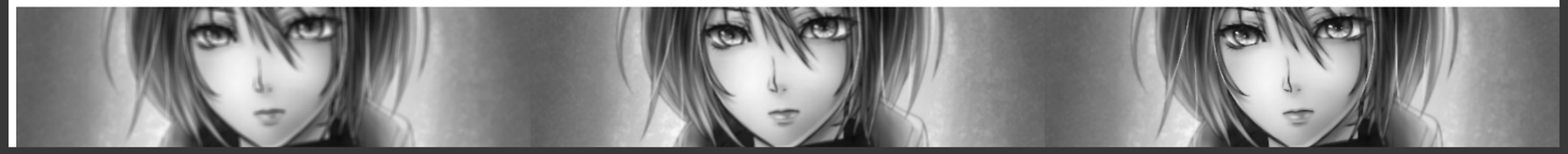
Median filtering on an image with impulse noise

```
blurred2 = cv2.medianBlur(imgn2, 3)
mst2=np.hstack((imgn2,blurred2))
cv2_imshow(mst2)
cv2.imwrite("Median filter - Impulse noise.jpg",blurred2)
```

True

Bilateral Filtering on an image with gaussian noise

```
bilateral1 = cv2.bilateralFilter(imgn1, 5, 75, 75)
bst1=np.hstack((processed_image1,image_gaussian_processed,bilateral1))
cv2_imshow(bst1)
```



Bilateral Filtering on an image with impulse noise

```
bilateral2 = cv2.bilateralFilter(imgn2, 5, 75, 75)
bst2=np.hstack((processed_image2,image_impulse_processed,bilateral2))
cv2_imshow(bst2)
```

Q4

Sobel

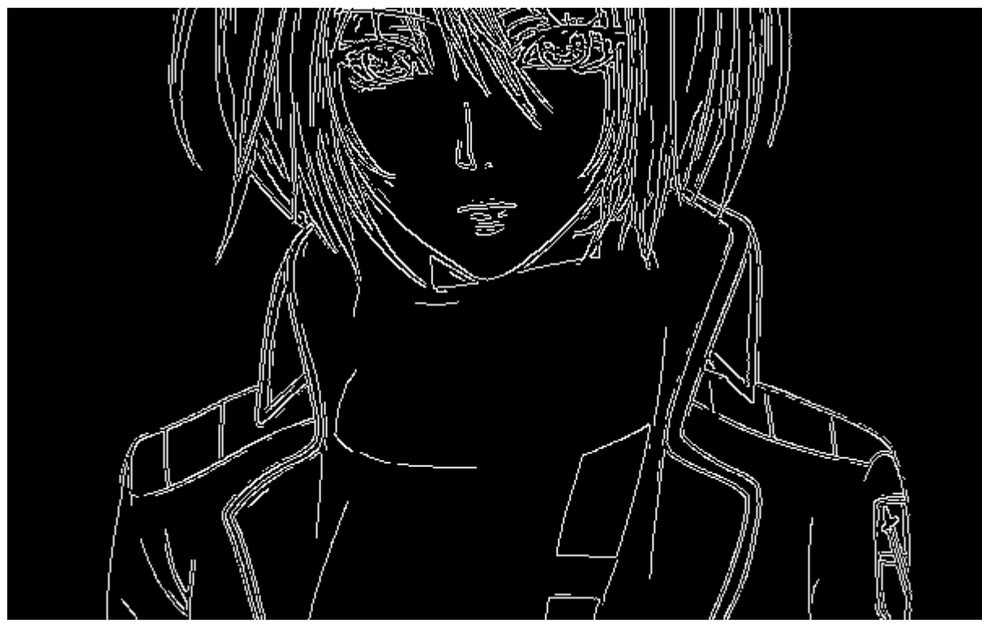
```
cv2_imshow(img)
sobelx = cv2.Sobel(img, cv2.CV_64F, 1, 0)
cv2_imshow(sobelx)
sobely = cv2.Sobel(img, cv2.CV_64F, 0, 1)
cv2_imshow(sobely)
```

Laplacian

```
laplacian = cv2.Laplacian(img, cv2.CV_64F, ksize=5)
cv2_imshow(laplacian)
```

Canny edge detector

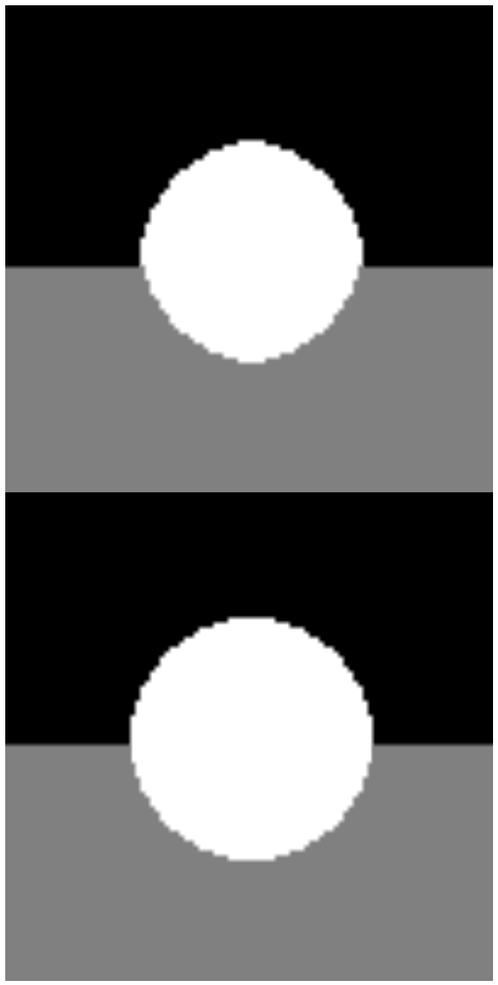
```
canny = cv2.Canny(img, 120, 170)
cv2_imshow(canny)
```



Q5

```
imgbw=cv2.resize(im1,(300,300))
cv2_imshow(imgbw)
```

```
kernel = np.ones((5,5), np.uint8)
imgE = cv2.erode(imgbw, kernel, iterations=4)
imgD = cv2.dilate(imgE, kernel, iterations=3)
cv2_imshow(imgE)
cv2_imshow(imgD)
```



Q6

```
e=imgbw-imgE
cv2_imshow(e)
```