**Digital Image Processing**

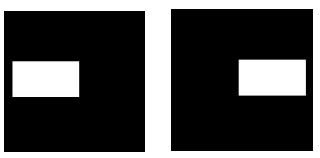
**Assignment#2**

**Mehmood Amjad**

**19I-0472**

**Question # 1:**

After getting the path of both the images, I used & (and) operator to find the overlapping points between the two images:



img = img1 & img2

After this, I counted the pixels on the overlapping area using a loop and displayed the result on the final image.

# getting area

area = 0

# loop to count pixels of the white region

for i in range(0,img.shape[0]):

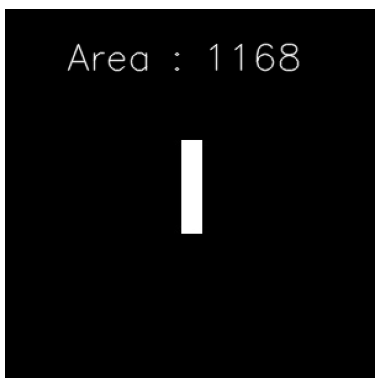
    check2 = True

    for j in range(0,img.shape[1]):

        if (img[i][j]==255):

            area += 1

**Output:**



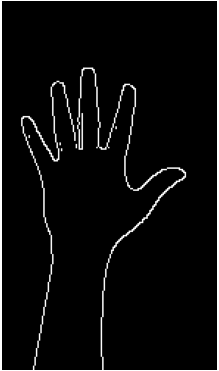
**Question # 2:**

First I read the image.



Then, I used the canny function to detect the edges of the image by canny edge detector method. The value of lower threshold is 100 and higher threshold is 200.

edge = cv2.Canny(img,100,200)



After this, I used a loop and converted all the white pixels in the resulting image after canny to green in the original image.

# loop to convert the edges into green in original image

for i in range(0,img.shape[0]):

    for j in range(0,img.shape[1]):

        if (edge[i][j]==255):

            img[i][j] = (0,255,0)

The final output is shown below.

**Output:**

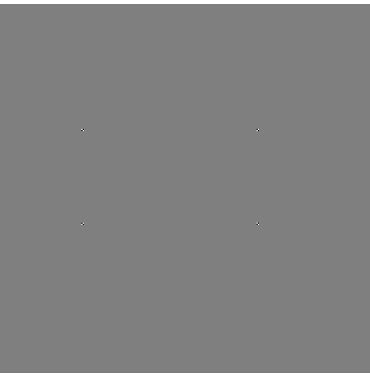


**Question # 3:**

First, I read the image. Then, I used the sobel operator in xy direction with kernel size 1 to find the corner points of the box.

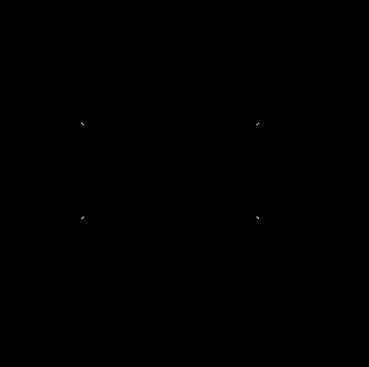
# using sobel operator of x and y direction with kernel size 1

img2 = cv2.Sobel(img,cv2.CV\_64F,2,2,ksize=1)



After this, I used threshold to get a clear image of the corner points.

ret, thresh1 = cv2.threshold(img2, 100, 255, cv2.THRESH\_BINARY)



Then, I calculated the x and y of each corner point and stored them in n array.

arrx = [0,0,0]

arry = [0,0,0]

count = 0

index = 0

# loop to get the corner points

for i in range(thresh1.shape[0]):

    for j in range(thresh1.shape[1]):

        if (thresh1[i][j] == 255):

            if(count == 0 or count == 1 or count == 6):

                arrx[index] = i

                arry[index] = j

                index += 1

            count += 1

In the end I calculated the area of the image using this value.

# calculating area using the corner points

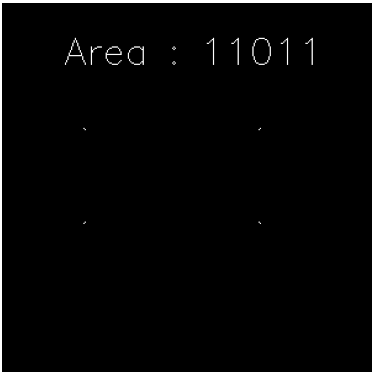
width = arry[1] - arry[0]

length = arrx[2] - arrx[0]

area = length \* width

The final image is shown below:

**Output:**

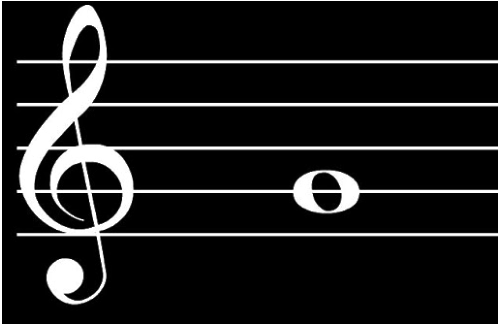


**Question # 4:**

I used morphological operations erosion and dilation to get the horizontal lines. First, I got the negative of the image after reading the image.

# getting the negative of the image

img2 = 255 - img



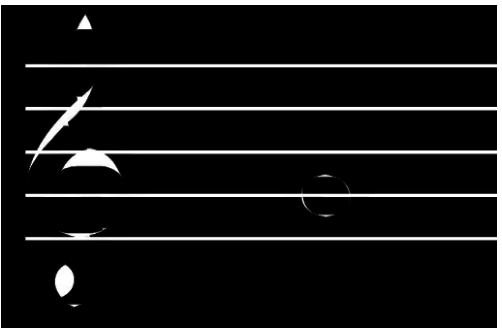
Then, I set a kernel on hit and trial basis for morphological operators. After this, I erode the image 4 times.

# setting the kernel of the image

kernel = np.ones((1, 20), np.uint8)

# Eroding the image 4 times

img2 = cv2.erode(img2,kernel,iterations=4)



After first erosion



After second erosion



After third erosion



After fourth erosion

Then, I dilated the image 1 time.

# DIlating the image 1 time

img2 = cv2.dilate(img2,kernel,iterations=1)



In the end, I took the negative of the image and got the final output.

# Getting the negative of the final image

img2 = 255 - img2

**Output:**

