```
    grayscale = cv2.cvtColor(img,cv2.COLOR_RGB2GRAY)

    _,binary = cv2.threshold(grayscale,127,255,cv2.THRESH_BINARY)

    mask = cv2.circle(black_img,(r//2,c//2),300,(255,255,255),-1)

    imgAnd =  cv2.bitwise_and(grayscale.copy(),mask)
    imgOr =  cv2.bitwise_or(grayscale.copy(),mask)

    imgXor =  cv2.bitwise_xor(grayscale.copy(),mask)
    imgNot =  cv2.bitwise_not(grayscale.copy())

    plt.plot(y,histogram)
    plt.ylim(0,)
    plt.hist(img.ravel(),256,[0,256])

    x,y = np.random.randint(0,(r,c))

    temp = np.rint(temp)

    process_img1 = cv2.filter2D(grayscale,-1,kernel1)

    lapacianKernel = np.array([[-1, -1, -1],
                         [-1, 8, -1],
                         [-1, -1, -1]])

    sobelKernel  = np.array([
                                     [-1, 0, 1],
                                     [-2, 0, 2],
                                     [-1, 0, 1]
                                     ])

    gaussianKernel = np.array([[1/16,1/8,1/16],
                               [1/8,1/4,1/8],
                               [1/16,1/8,1/16]])

    histogram =  cv2.calcHist([img],[0],None,[256],[0,256])

    erosionImg = cv2.erode(binaryImg,structElem1,iterations=1)

    dilationImg = cv2.dilate(binaryImg,structElem1,iterations=1)

    openingImg = cv2.morphologyEx(binaryImg, cv2.MORPH_OPEN,structElem1)

    closingImg = cv2.morphologyEx(binaryImg, cv2.MORPH_CLOSE,structElem1)

    paddedImg = np.pad(binaryImg,1,constant_values=0)

    ftImg = np.fft.fft2(grayscale)
    centeredfti_img = np.fft.fftshift(ftImg)

    magnitude_spectrum = 100 * np.log(np.abs(ftImg))
    centered_magnitude_spectrum = 100 * np.log(np.abs(centeredfti_img))

    ftiImg_gf =  centeredfti_img * filter
    filtered_img = np.abs(np.fft.ifft2(ftiImg_gf))

}
```