ASSIGNMENT

Image Segmentation and Analysis

Muhammmad Sunaam – msunaam.bscs21seecs@seecs.edu.pk - 393223

Contents

Problem Statement

The problem targeted in this assignment is to detect the count of objects and resize, reshape, and recolor an image.

Code Structure

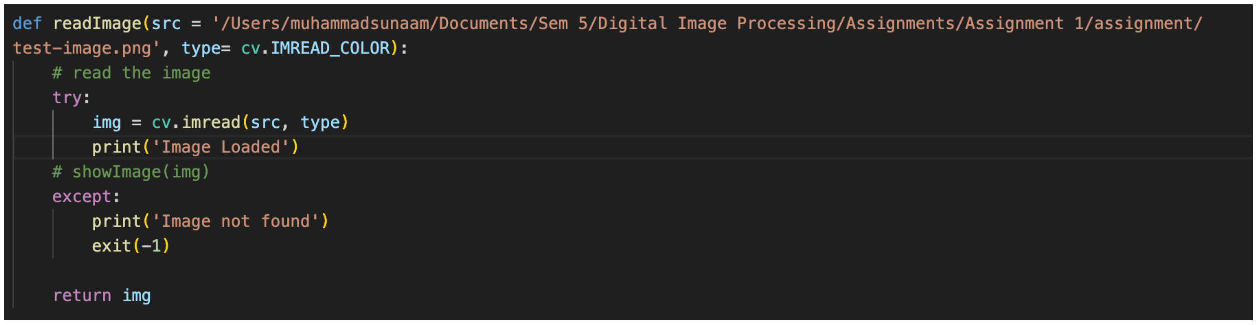
To keep the code design modular, the code is split into task files and a utility file. The utility file keeps all the functions that are repeatedly required such as reading the file, getting image shape, etc. This approach reduces code redundancy and improves readability. As there are four tasks, each task has its own separate file which imports the utility file and performs some function on the image. All the code is kept in a Github Repository, which can be accessed here.

Utils File

The utils file or the utility file keeps all the functions that are repeatedly used in each task. This keeps the code clean, concise and less redundant. Following are the functions available in the utils file.

Read Image

ReadImage function essentially reads the image using cv2.imread function. It incorporates error handling using try and except blocks. It takes two arguments, src, and type. Both arguments have default parameters.

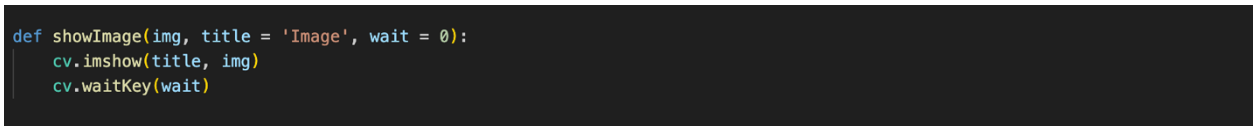


```python
def readImage(src = '/Users/muhammadsunaam/Documents/Sem 5/Digital Image Processing/Assignments/Assignment 1/assignment/
test-image.png', type= cv.IMREAD_COLOR):
    # read the image
    try:
        img = cv.imread(src, type)
        print('Image Loaded')
    # showImage(img)
    except:
        print('Image not found')
        exit(-1)

    return img
```

Show Image

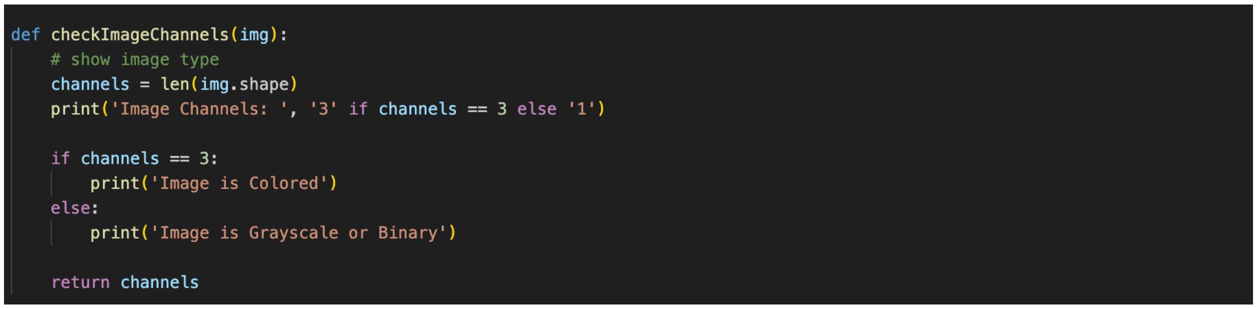
ShowImage function basically displays the image in a window using cv2.imshow function. It takes three arguments, the first one being the image object, while the other two have default parameters and are the window title and wait time (before the window closes) arguments.



```python
def showImage(img, title = 'Image', wait = 0):
    cv.imshow(title, img)
    cv.waitKey(wait)
```

Check Image Channels

CheckImageChannels is a function that uses the shape property of the image object to determine how many channels the image has. This helps in determining if the image is colored or grayscale. It takes only one argument, which is the image object.



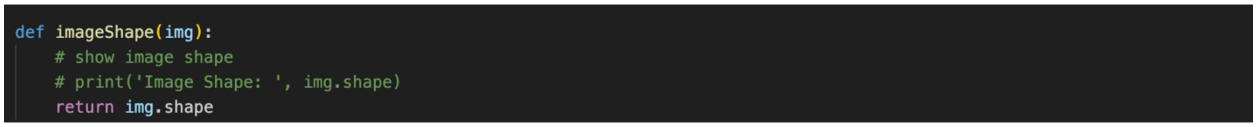
```python
def checkImageChannels(img):
    # show image type
    channels = len(img.shape)
    print('Image Channels: ', '3' if channels == 3 else '1')

    if channels == 3:
        print('Image is Colored')
    else:
        print('Image is Grayscale or Binary')

    return channels
```

Image Shape

The Image shape function only returns the shape of the image and takes one argument which is the image object.



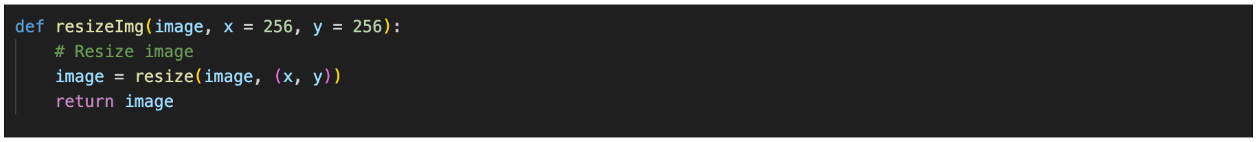
```python
def imageShape(img):
    # show image shape
    # print('Image Shape: ', img.shape)
    return img.shape
```

Task 1

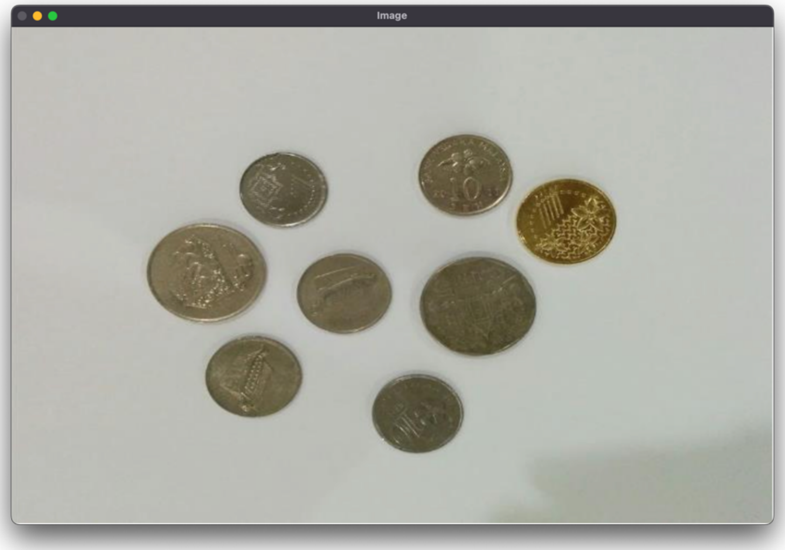
The first task was to read the image and resize it to 256 by 256 pixels.

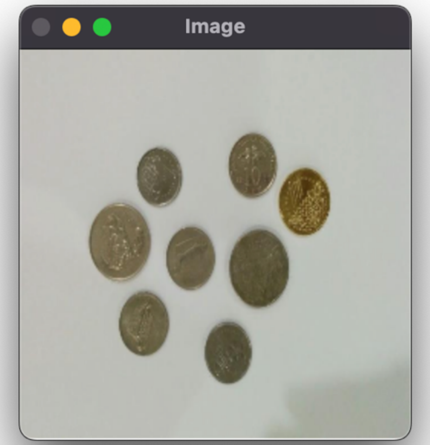
Approach

Simply put the approach was to use the cv2 function resize which takes three arguments. The first argument is the image object and the second and third arguments are the number of pixels wanted in x and y axis.



```python
def resizeImg(image, x = 256, y = 256):
    # Resize image
    image = resize(image, (x, y))
    return image
```



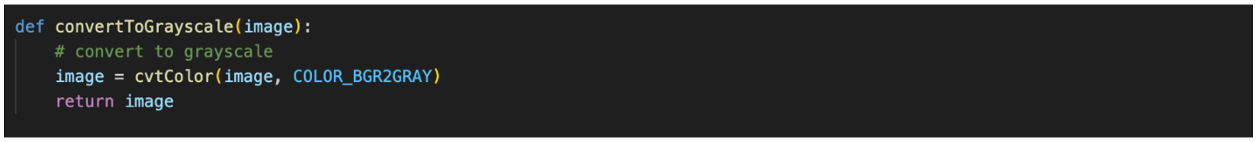


Task 2

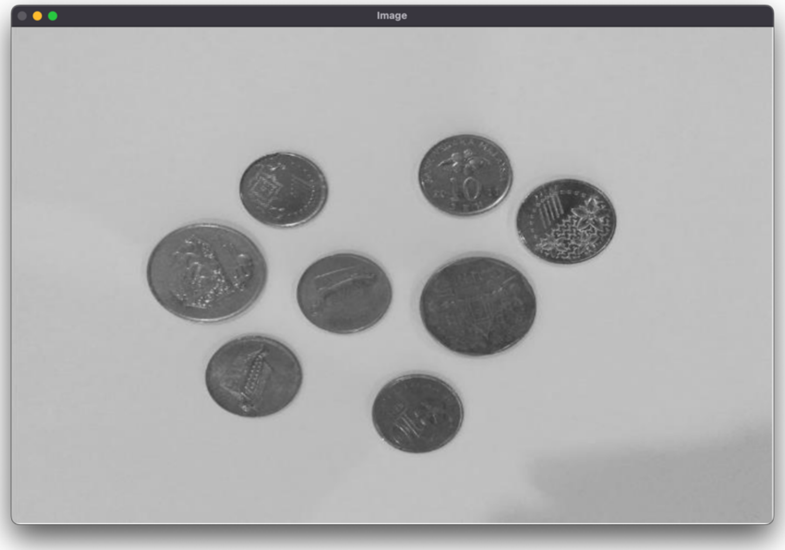
The second task was to read the image and convert it to a grayscale image.

Approach

The cv2 function cvtColor was used which takes two arguments, the first one being the image object and the second one being an Enum which defines which color conversion is required.



```python
def convertToGrayscale(image):
    # convert to grayscale
    image = cvtColor(image, COLOR_BGR2GRAY)
    return image
```

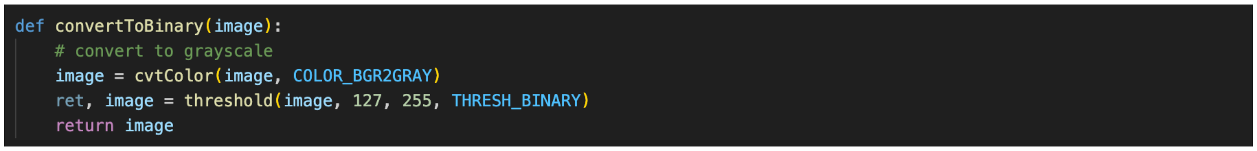


Task 3

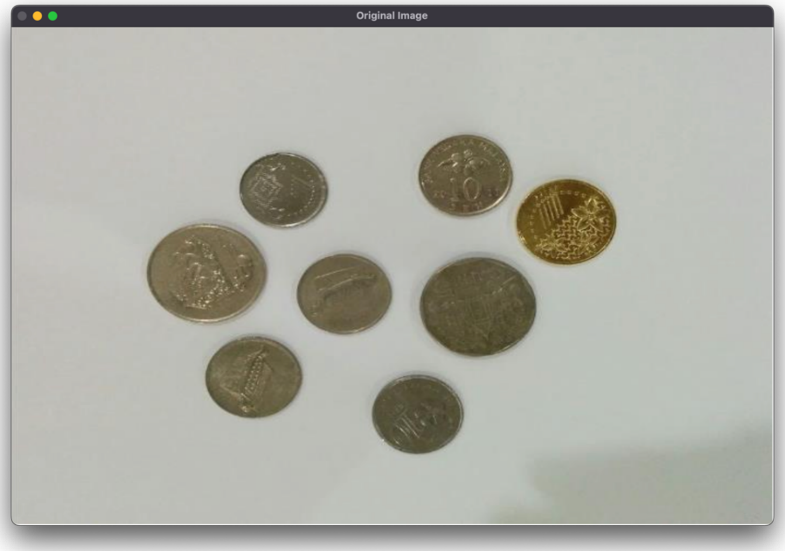
The third task was to read the image and convert it to a binary image.

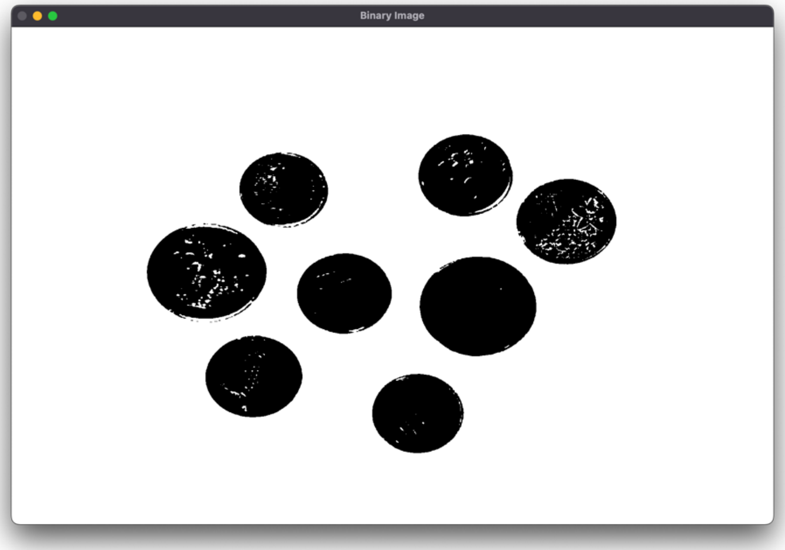
Approach

Initially, the cvtColor function from cv2 is used to make the image grayscale. Then a threshold function (from cv2) is used to convert the image to a binary format. The threshold function takes four arguments. The first argument is the image object. The second and third arguments are the thresh and maxval values. Essentially, values greater than thresh will be mapped onto maxval while the rest will be mapped to zero. The fourth argument is an Enum which describes the type of thresholding required.



```python
def convertToBinary(image):
    # convert to grayscale
    image = cvtColor(image, COLOR_BGR2GRAY)
    ret, image = threshold(image, 127, 255, THRESH_BINARY)
    return image
```





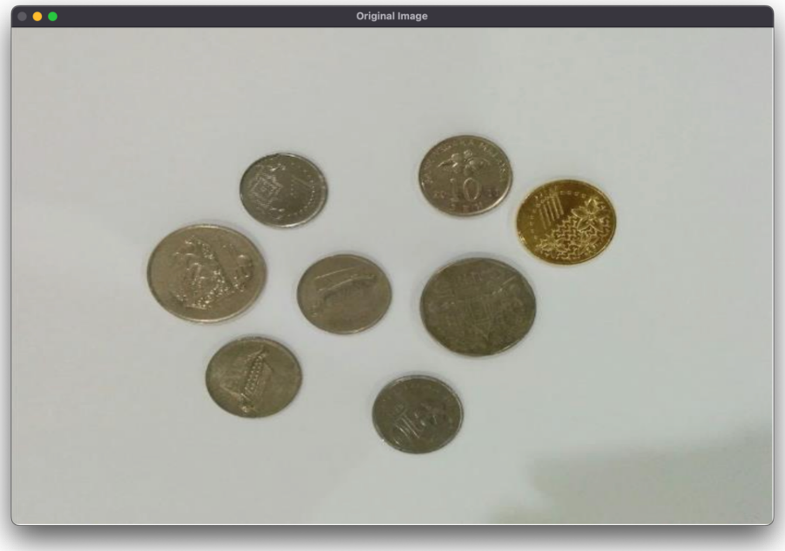
Task 4

The fourth task was to read the image, segment it, and count the number of coins in it.

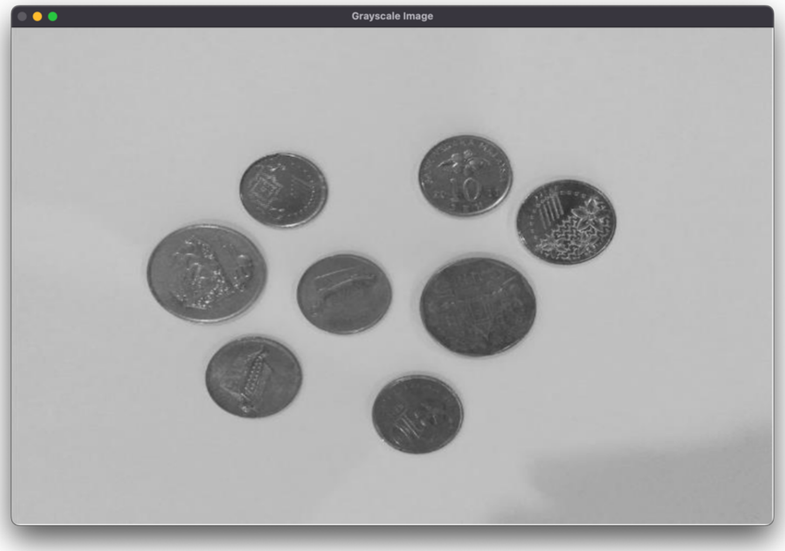
Approach

The image is first read and then converted to grayscale. After that, it is blurred using the GuassianBlur function from cv2. The GuassianBlur function is used because it applies more natural blurring which will be important later in edge detection. Other filters in the cv2 library may overblur the image. It takes 3 arguments. The first one is the image object. The second argument is a tuple that contains Gaussian Kernel size (larger size means more blur). The third argument is the standard deviation (0 means it is calculated automatically). After applying blur, the image is then converted to binary, and then edge detection is applied using the cv2 Canny function. The Canny function takes three arguments. The first argument, as always, is the image object. The second and third arguments are the thresholds. Canny does use two thresholds (upper and lower): If a pixel gradient is higher than the upper threshold, the pixel is accepted as an edge. If a pixel gradient value is below the lower threshold, then it is rejected. After that cv2 function findContours is used. The function retrieves contours from the binary image using the algorithm. The contours are a useful tool for shape analysis and object detection and recognition. After getting all the contours, all the contours that have an area greater than 100 are rendered on the original image, using the drawContours function. It takes five arguments. The first argument is the image object, the second argument is the contour object, third argument is the contour index, -1 indicates that all the contours passed must be drawn. The fourth argument is the color (0,255,0) indicates green color. The last argument is the line thickness. All the contours with an area greater than 100 indicate a coin. Therefore, the number of contours with an area greater than 100 indicates a coin. The number of coins is printed in the terminal.

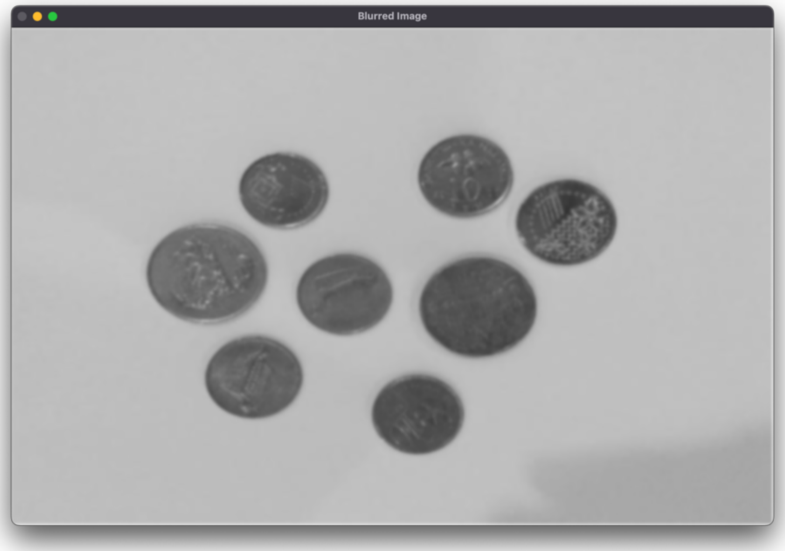
Original Image



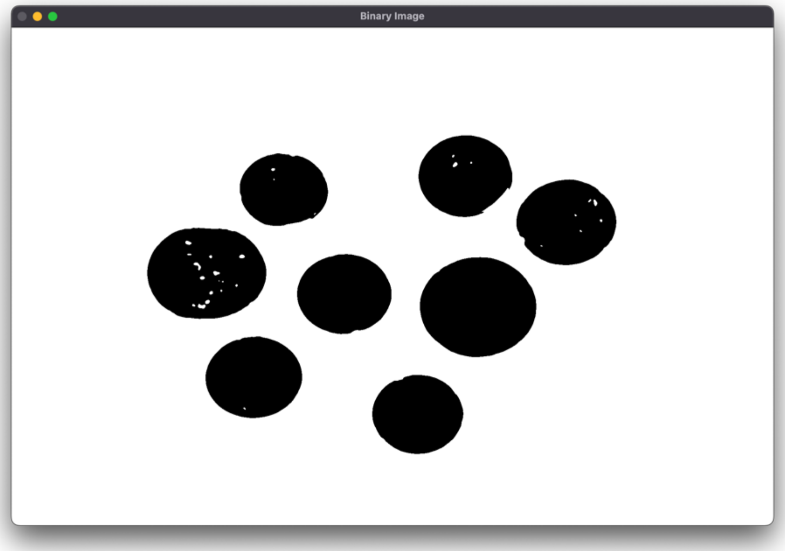
Grayscale Image



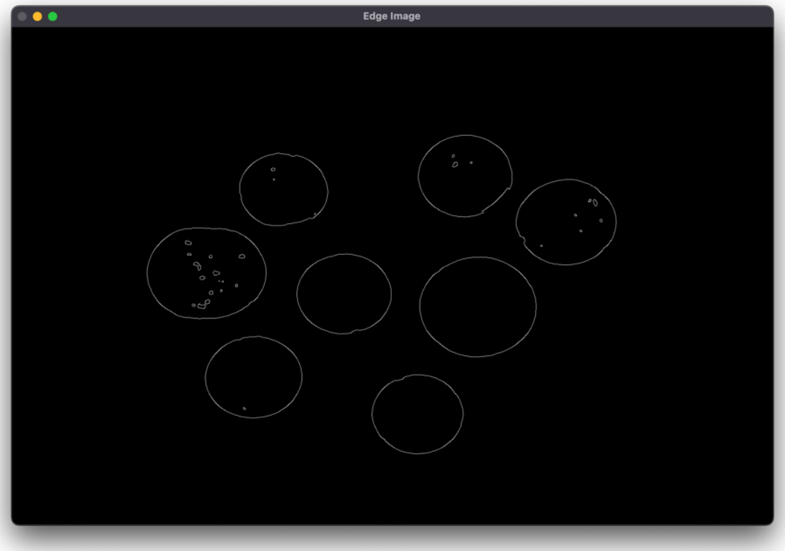
Blurred Image



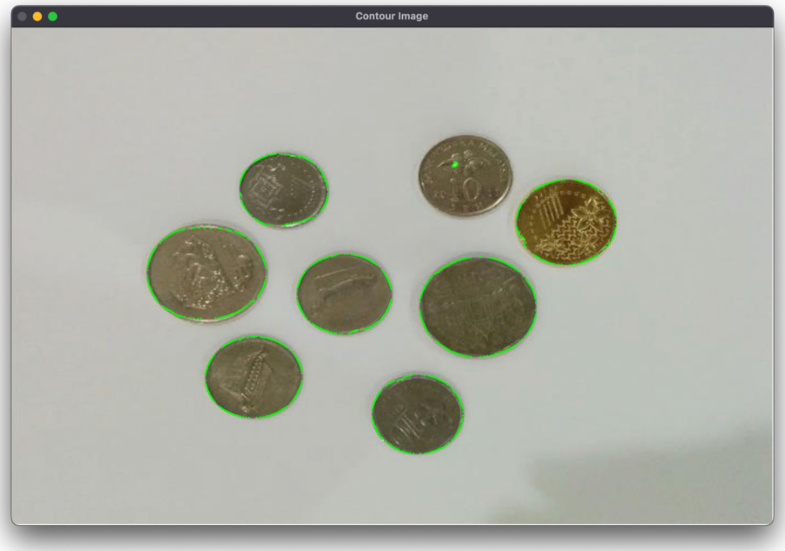
Binary Image



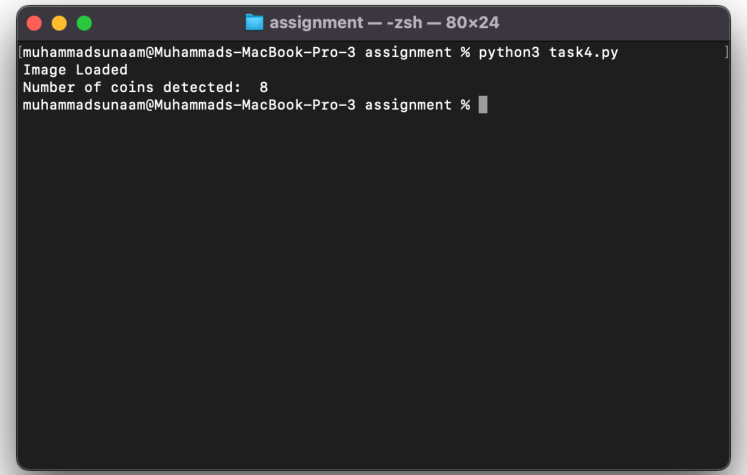
Edge Image



Contour Image



Console



```
muhammadsunaam@Muhammads-MacBook-Pro-3 assignment % python3 task4.py
Image Loaded
Number of coins detected:  8
muhammadsunaam@Muhammads-MacBook-Pro-3 assignment %
```

Conclusion

In conclusion, the open cv library cv2 was used to successfully count the number of objects in an image, by using several techniques such as edge detection and contouring.