الجامعة الإسلامية العالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يونيۏرسيتي إسلام انتارابڠسا مليسيا

KULLIYYAH OF ENGINEERING
DEPARTMENT OF MECHATRONIC ENGINEERING

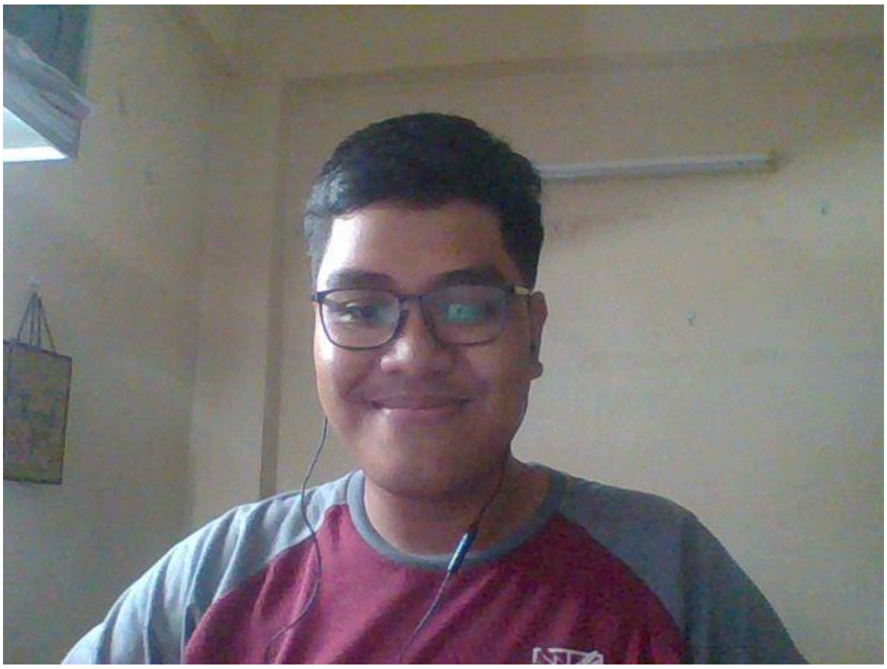
DLCV WORKSHOP

ASSIGNMENT 1

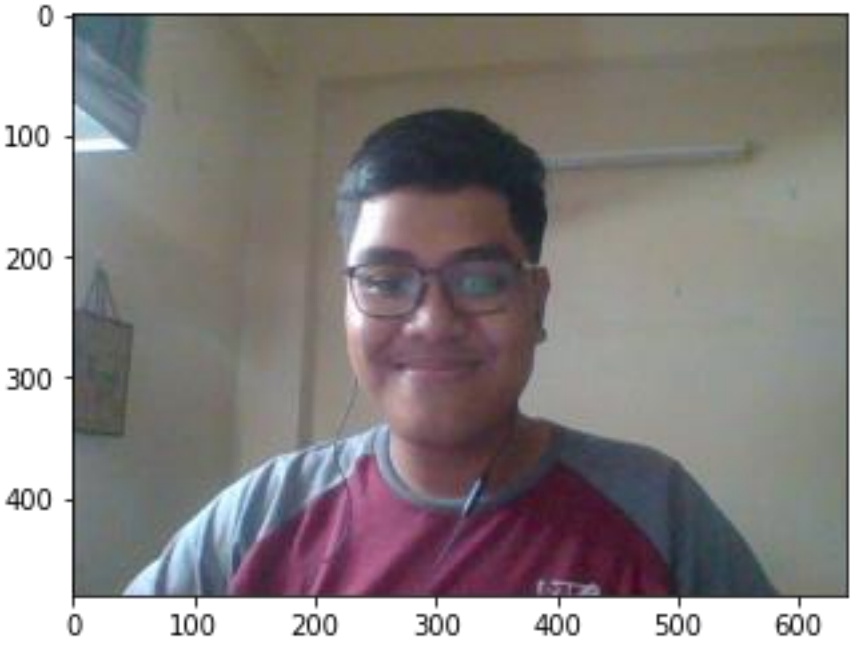
MUHAMMAD FAREED BIN MOHD NOOR 1810757

1 A&B)

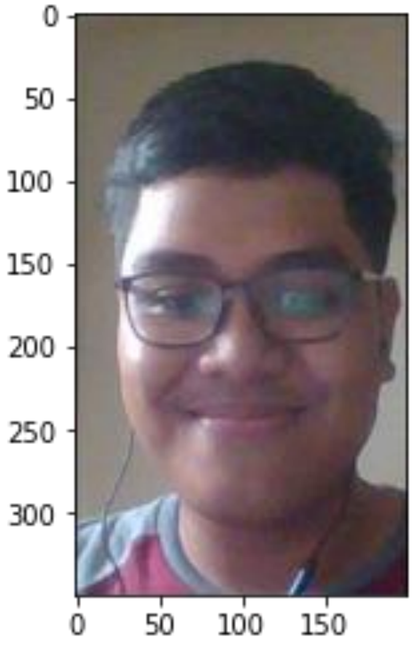
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from IPython.display import display, Javascript
from google.colab.output import eval_js
from base64 import b64decode

def take_photo(filename='smile.jpg', quality=0.8): #save the other two images with different name
  js = Javascript('''
    async function takePhoto(quality) {
      const div = document.createElement('div');
      const capture = document.createElement('button');
      capture.textContent = 'Capture';
      div.appendChild(capture);

      const video = document.createElement('video');
      video.style.display = 'block';
      const stream = await navigator.mediaDevices.getUserMedia({video: true});

      document.body.appendChild(div);
      div.appendChild(video);
      video.srcObject = stream;
      await video.play();

      // Resize the output to fit the video element.
      google.colab.output.setIframeHeight(document.documentElement.scrollHeight, true);

      // Wait for Capture to be clicked.
      await new Promise((resolve) => capture.onclick = resolve);

      const canvas = document.createElement('canvas');
      canvas.width = video.videoWidth;
      canvas.height = video.videoHeight;
      canvas.getContext('2d').drawImage(video, 0, 0);
      stream.getVideoTracks()[0].stop();
      div.remove();
      return canvas.toDataURL('image/jpeg', quality);
    }
    ''')
  display(js)
  data = eval_js('takePhoto({})'.format(quality))
  binary = b64decode(data.split(',')[1])
  with open(filename, 'wb') as f:
    f.write(binary)
  return filename
from IPython.display import Image
try:
  filename = take_photo()
  print('Saved to {}'.format(filename))

  # Show the image which was just taken.
  display(Image(filename))

except Exception as err:
  # Errors will be thrown if the user does not have a webcam or if they
 do not
  # grant the page permission to access it.
  print(str(err))
```



```
img = cv2.imread("/content/smile.jpg") #edit the path and name of the c
aptured image

color_coverted = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

plt.imshow (color_coverted )
plt.show()
```

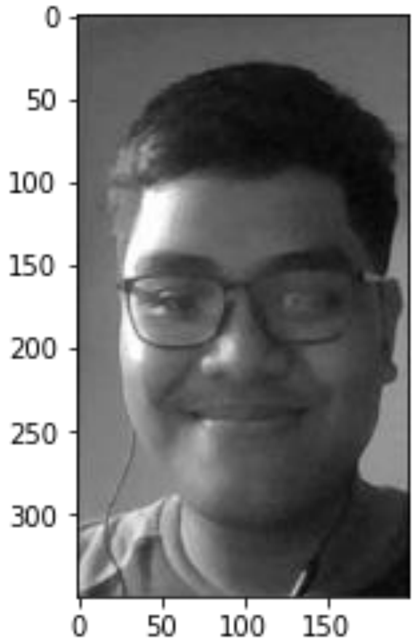


```
face=color_coverted[50:400, 200:400]
plt.imshow(face)
```

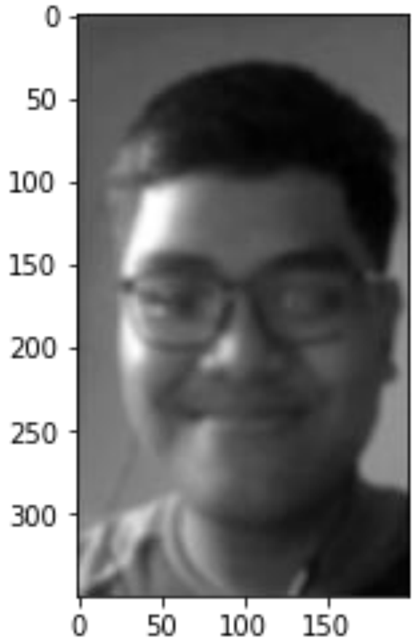


```
def displayImage(image):
    if len(image.shape)==3:
        face = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        plt.imshow(face)
        plt.show()

    else:
        plt.imshow(image, cmap="gray")
        plt.show()
grayImage = cv2.cvtColor(face, cv2.COLOR_RGB2GRAY)
```

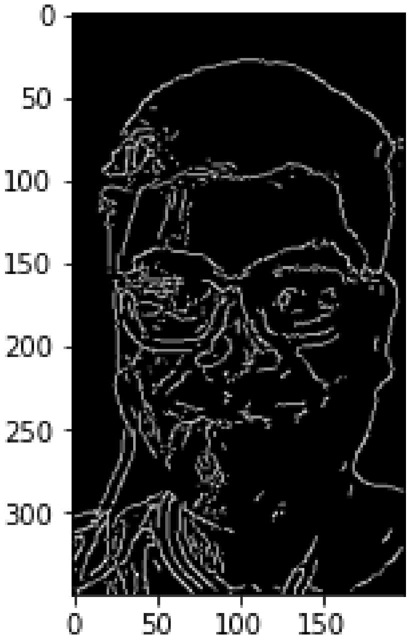
```
displayImage(grayImage)
```

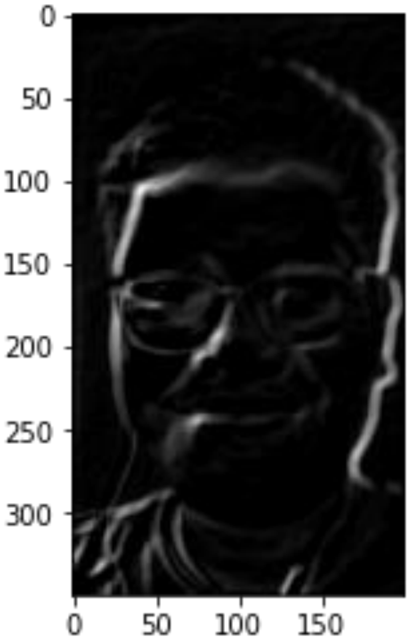


```
imgBlur=cv2.GaussianBlur(grayImage,(7,7),4)
displayImage(imgBlur)
```



```
# canny
imgCanny=cv2.Canny(imgBlur,20,25)

# sobel
sobelx = cv2.Sobel(imgBlur, cv2.CV_8U,1,0,ksize=3)
sobely = cv2.Sobel(imgBlur, cv2.CV_8U,0,1,ksize=3)
imgSobel = sobelx + sobely

displayImage(imgCanny)
displayImage(imgSobel)
```



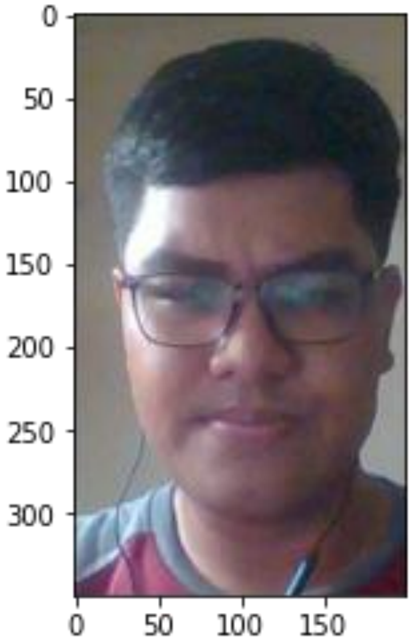


For other results using the same code adjust cropping, threshold value, etc (Angry, Surprise)

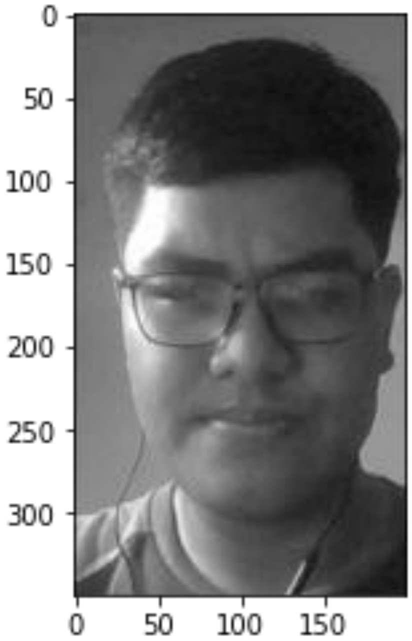
Angry:

Captured Image

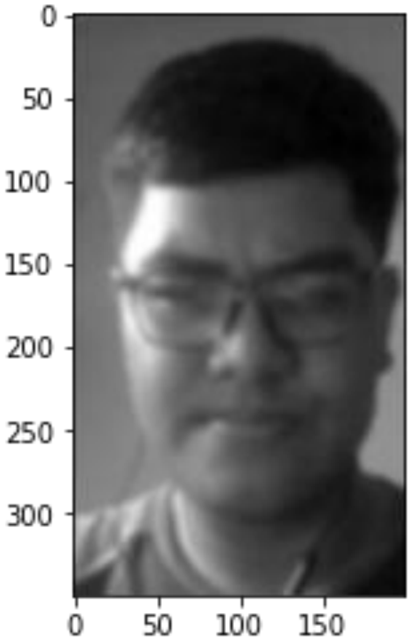
Color Coverted



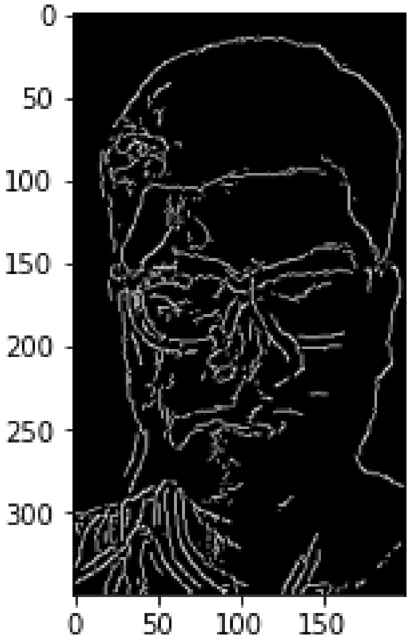
Cropped Image



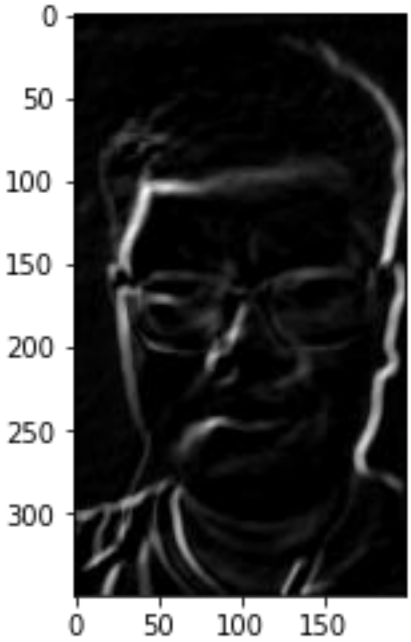
Gray Image



Blur Image



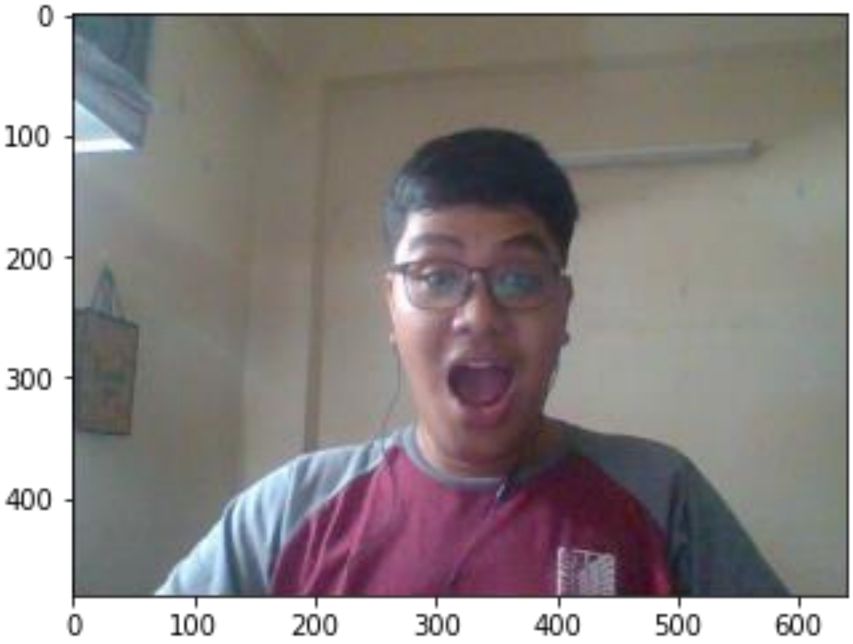
Canny Edge Detection



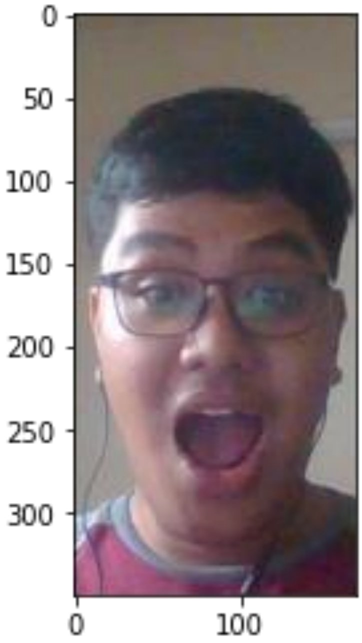
Sobel Edge Detection

Surprise:

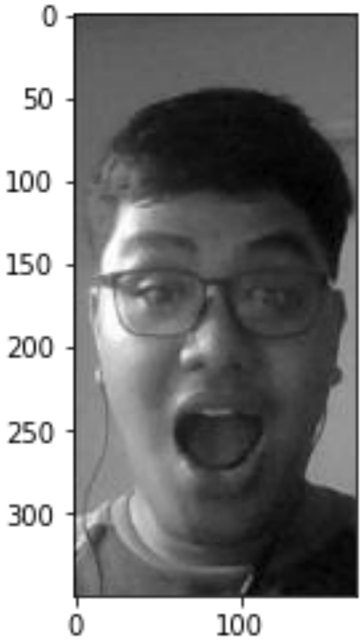
Captured Image



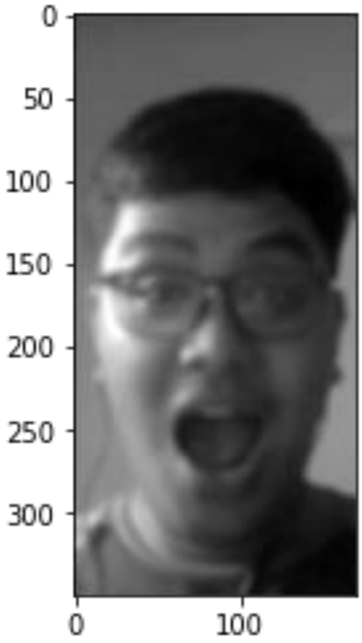
Color Coverted



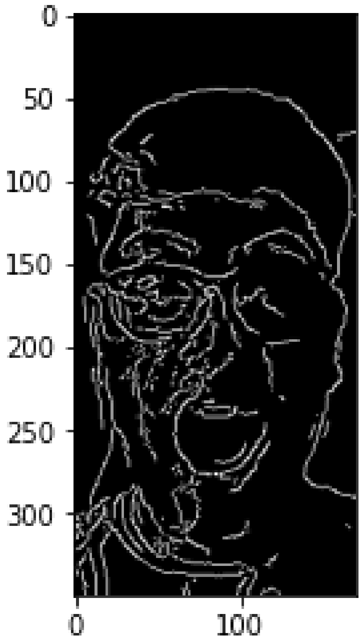
Cropped Image



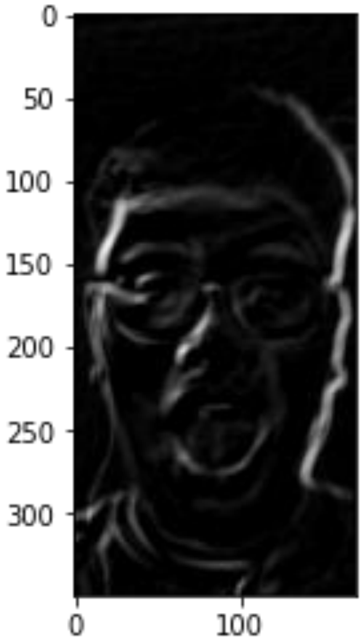
Gray Image



Blur Image



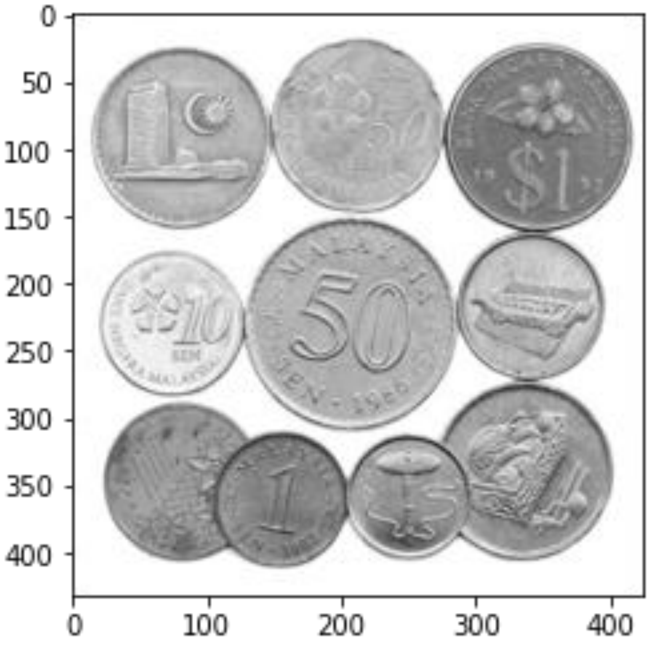
Canny Edge Detection



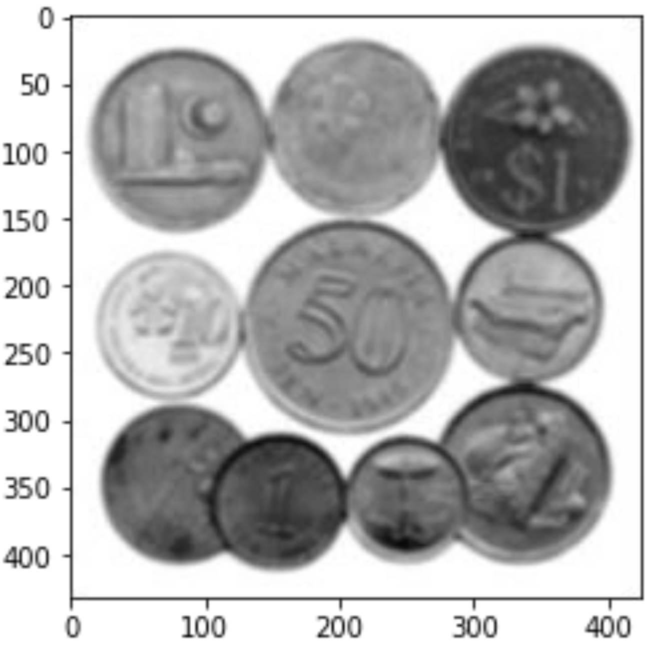
Sobel Edge Detection

1 C)

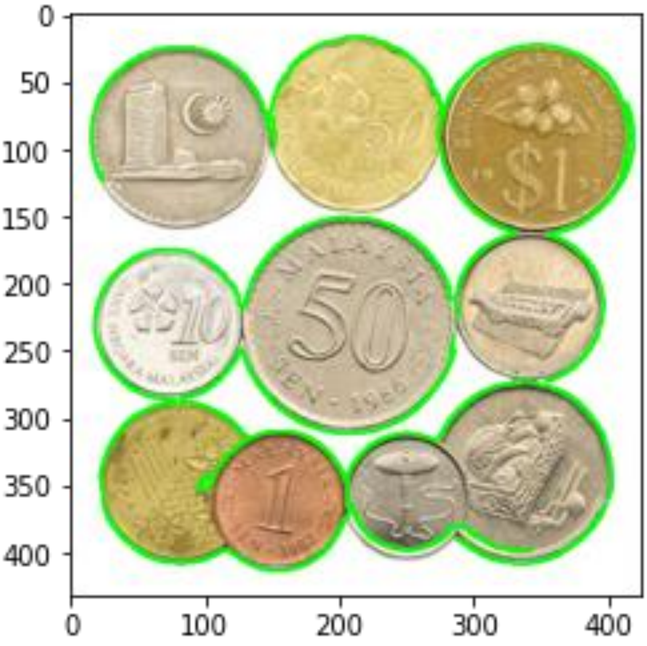
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from IPython.display import Image
from google.colab import drive
drive.mount('/content/gdrive')
path = "/content/gdrive/MyDrive/DLIVACV_workshop/Assignment/Assignment_
1/"
def displayImage(image):
    if len(image.shape)==3:
        color_coverted = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        plt.imshow(color_coverted)
        plt.show()

    else:
        plt.imshow(image, cmap="gray")
        plt.show()

img = cv2.imread(path+'coin.jpeg')
grayImage =cv2.cvtColor(img,cv2.COLOR_RGB2GRAY)
displayImage(grayImage)
```



```
blurImage=cv2.GaussianBlur(grayImage,(7,7),5)
displayImage(blurImage)
```



```
cannyImage = cv2.Canny(blurImage,25,210)
displayImage(cannyImage)
(cnt,heirarchy)=cv2.findContours(cannyImage.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_NONE)
rgb=cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
cv2.drawContours(rgb, cnt, -1, (0,255,0),2)

plt.imshow(rgb)
```



```
print ('Coins in the image: ', len(cnt))
```

```
Coins in the image:  10
```