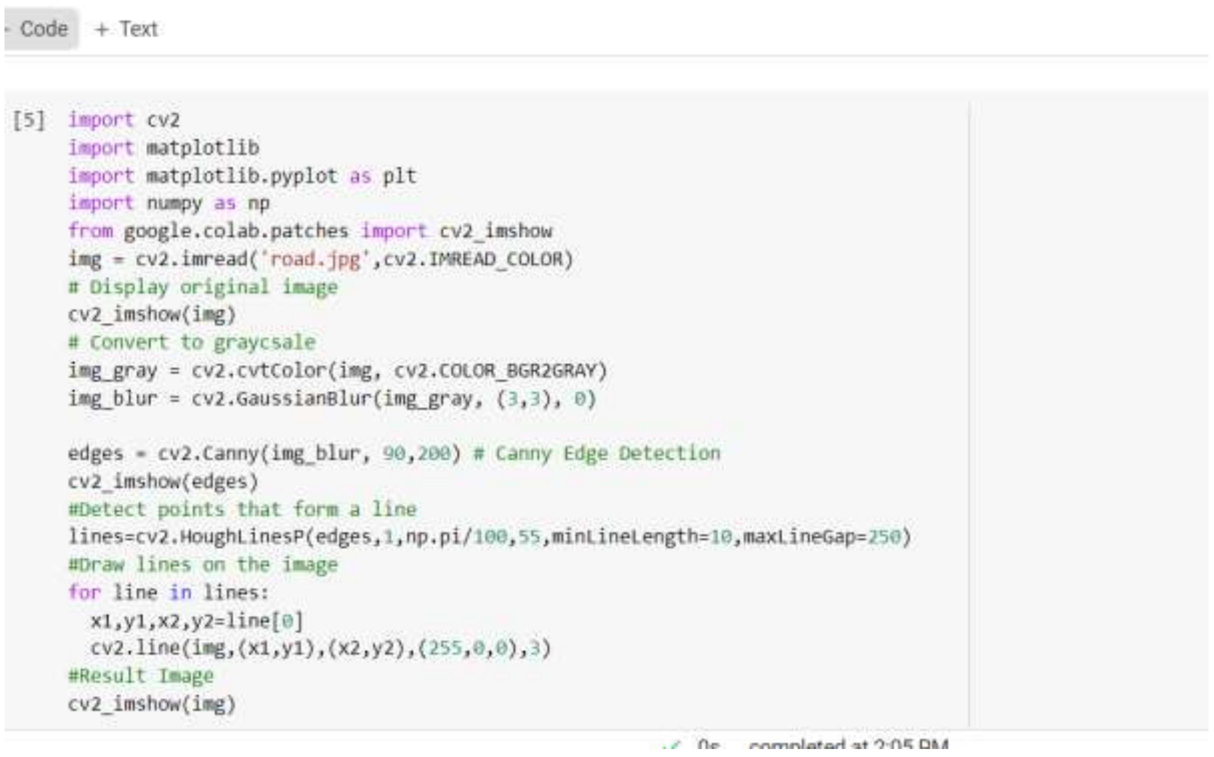
02_GAURAV BANE

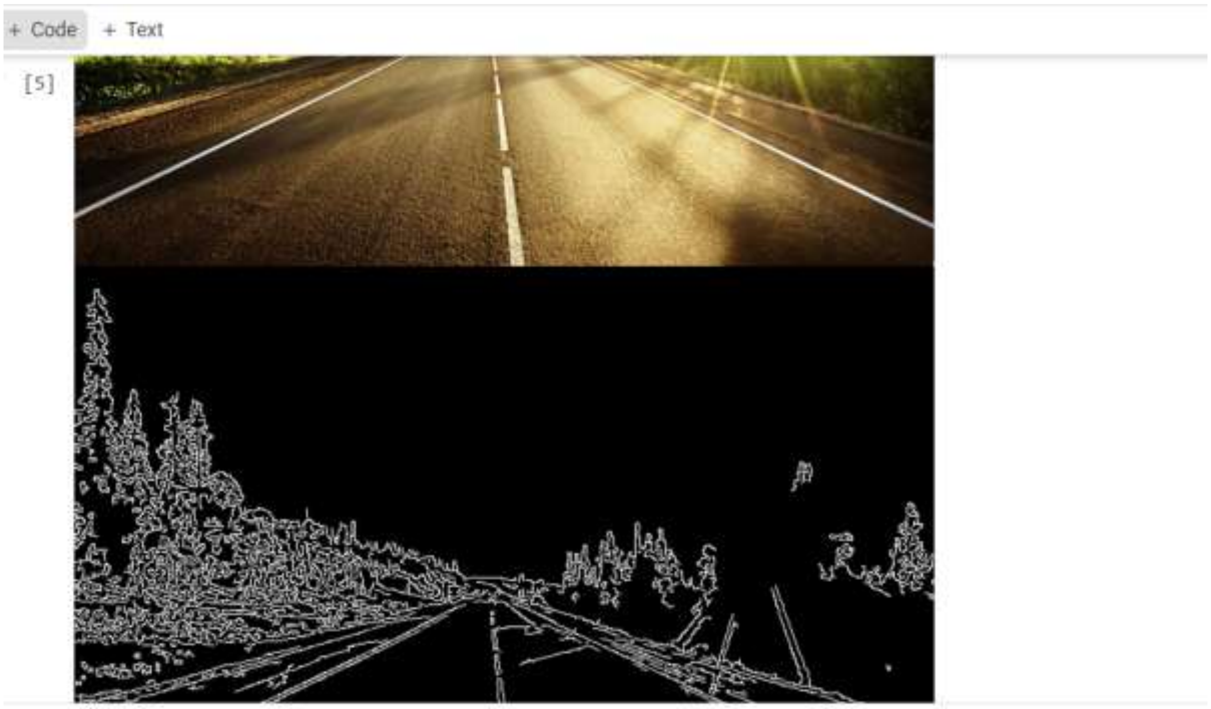
```
import cv2
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
from google.colab.patches import cv2_imshow
img = cv2.imread('road.jpg',cv2.IMREAD_COLOR)
# Display original image
cv2_imshow(img)
# Convert to graycsale
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
edges = cv2.Canny(img_gray, 50,200) # Canny Edge Detection
cv2_imshow(edges)
#Detect points that form a line
lines=cv2.HoughLinesP(edges,1,np.pi/100,55,minLineLength=10,maxLineGap=
250)
#Draw lines on the image
for line in lines:
  x1,y1,x2,y2=line[0]
  cv2.line(img,(x1,y1),(x2,y2),(255,0,0),3)
#Result Image
cv2_imshow(img)
```

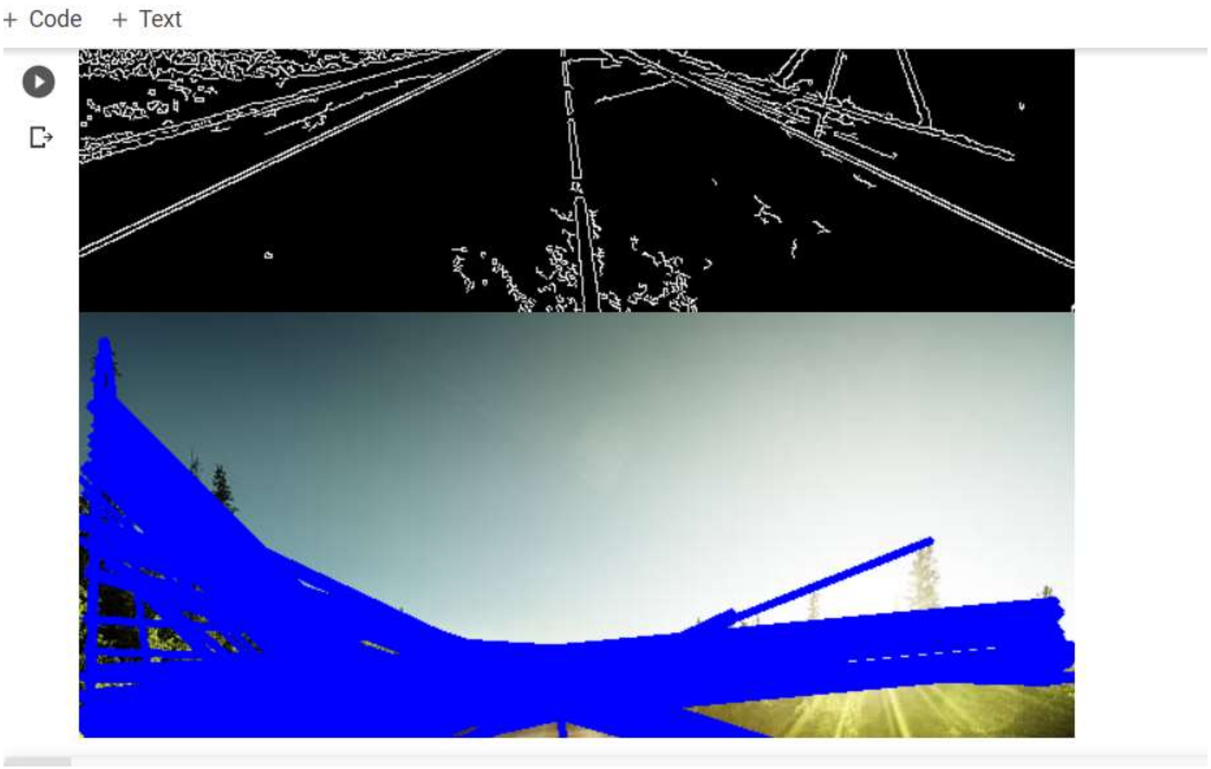


```
[5] import cv2
    import matplotlib
    import matplotlib.pyplot as plt
    import numpy as np
    from google.colab.patches import cv2_imshow
    img = cv2.imread('road.jpg',cv2.IMREAD_COLOR)
    # Display original image
    cv2_imshow(img)
    # Convert to graycsale
    img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    img_blur = cv2.GaussianBlur(img_gray, (3,3), 0)

    edges = cv2.Canny(img_blur, 90,200) # Canny Edge Detection
    cv2_imshow(edges)
    #Detect points that form a line
    lines=cv2.HoughLinesP(edges,1,np.pi/100,55,minLineLength=10,maxLineGap=250)
    #Draw lines on the image
    for line in lines:
      x1,y1,x2,y2=line[0]
      cv2.line(img,(x1,y1),(x2,y2),(255,0,0),3)
    #Result Image
    cv2_imshow(img)
```

✓ 0s completed at 2:05 PM

02_GAURAV BANE

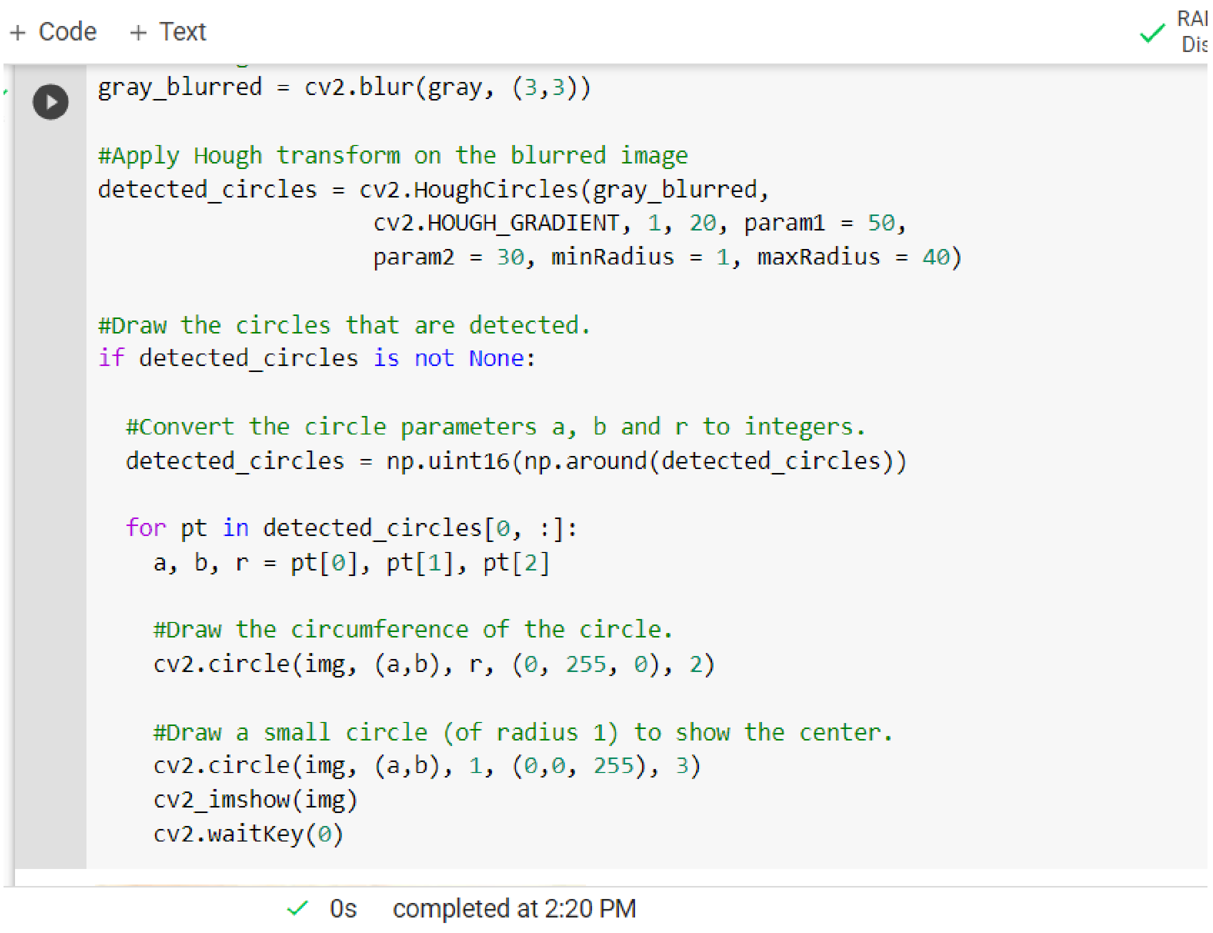




```
#Hue circle
img = cv2.imread('eyes.jfif', cv2.IMREAD_COLOR)

#Convert to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

#Blur using 3 * 3 kernel
```

02_GAURAV BANE

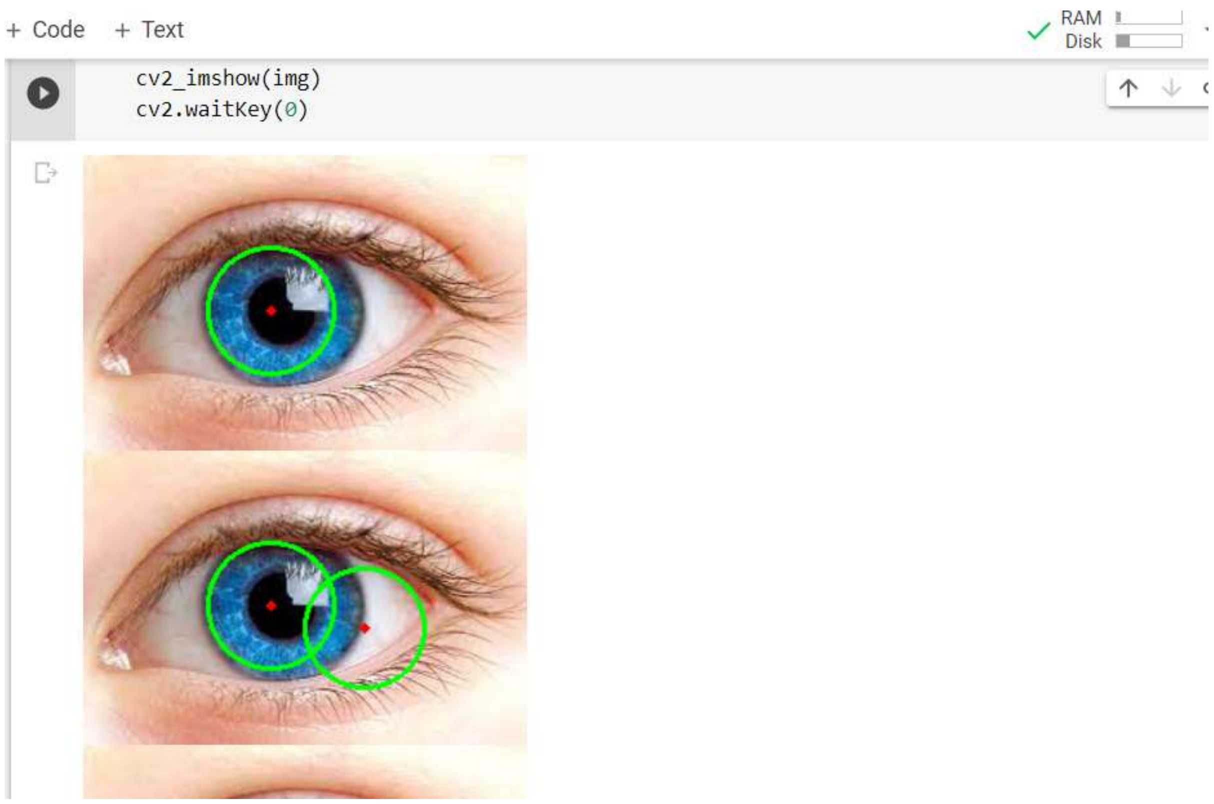
```
gray_blurred = cv2.blur(gray, (3,3))

#Apply Hough transform on the blurred image
detected_circles = cv2.HoughCircles(gray_blurred,
                    cv2.HOUGH_GRADIENT, 1, 20, param1 = 50,
                    param2 = 30, minRadius = 1, maxRadius = 40)

#Draw the circles that are detected.
if detected_circles is not None:

  #Convert the circle parameters a, b and r to integers.
  detected_circles = np.uint16(np.around(detected_circles))

  for pt in detected_circles[0, :]:
    a, b, r = pt[0], pt[1], pt[2]

    #Draw the circumference of the circle.
    cv2.circle(img, (a,b), r, (0, 255, 0), 2)

    #Draw a small circle (of radius 1) to show the center.
    cv2.circle(img, (a,b), 1, (0,0, 255), 3)
    cv2_imshow(img)
    cv2.waitKey(0)
```



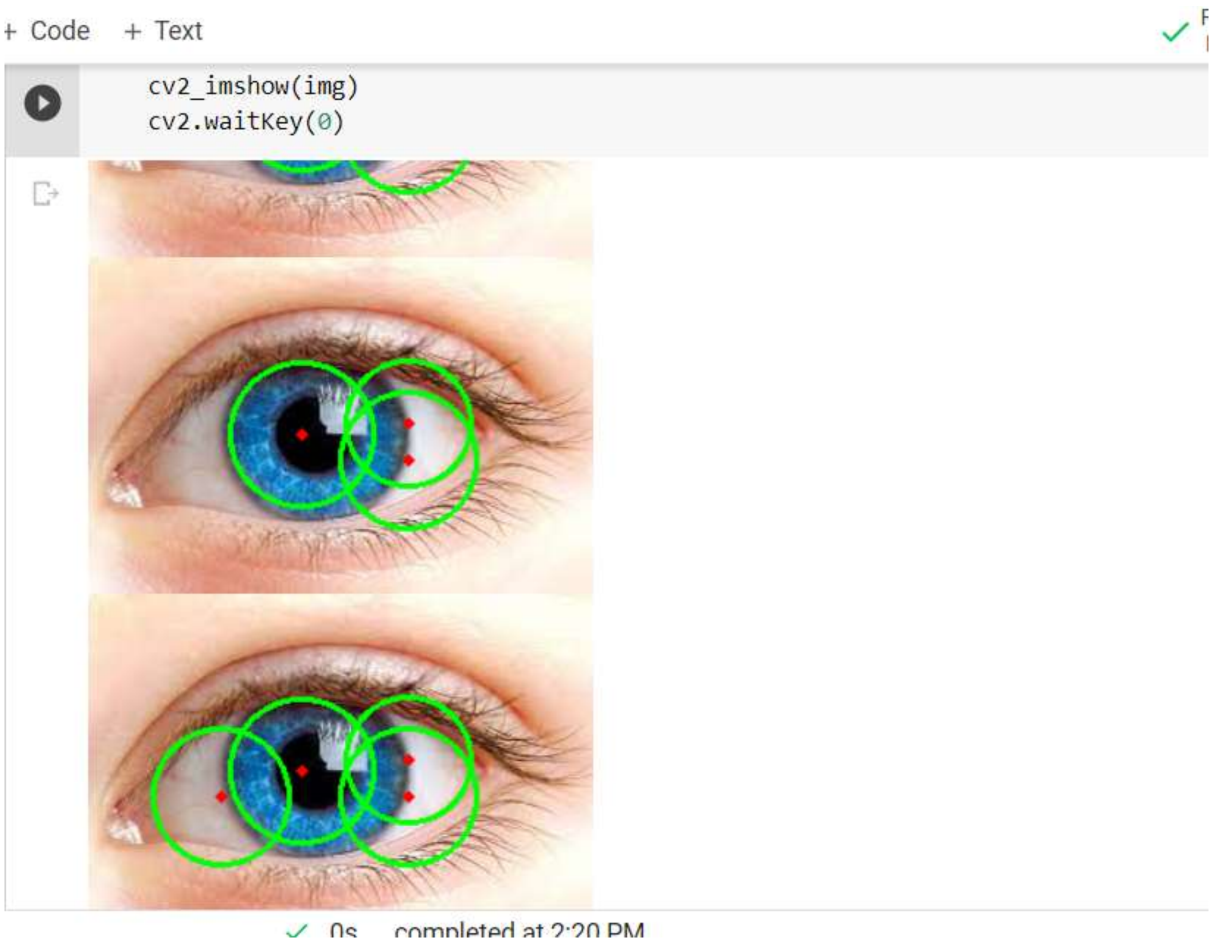
```
gray_blurred = cv2.blur(gray, (3,3))

#Apply Hough transform on the blurred image
detected_circles = cv2.HoughCircles(gray_blurred,
                    cv2.HOUGH_GRADIENT, 1, 20, param1 = 50,
                    param2 = 30, minRadius = 1, maxRadius = 40)

#Draw the circles that are detected.
if detected_circles is not None:

  #Convert the circle parameters a, b and r to integers.
  detected_circles = np.uint16(np.around(detected_circles))

  for pt in detected_circles[0, :]:
    a, b, r = pt[0], pt[1], pt[2]

    #Draw the circumference of the circle.
    cv2.circle(img, (a,b), r, (0, 255, 0), 2)

    #Draw a small circle (of radius 1) to show the center.
    cv2.circle(img, (a,b), 1, (0,0, 255), 3)
    cv2_imshow(img)
    cv2.waitKey(0)
```

✓ 0s completed at 2:20 PM

02_GAURAV BANE

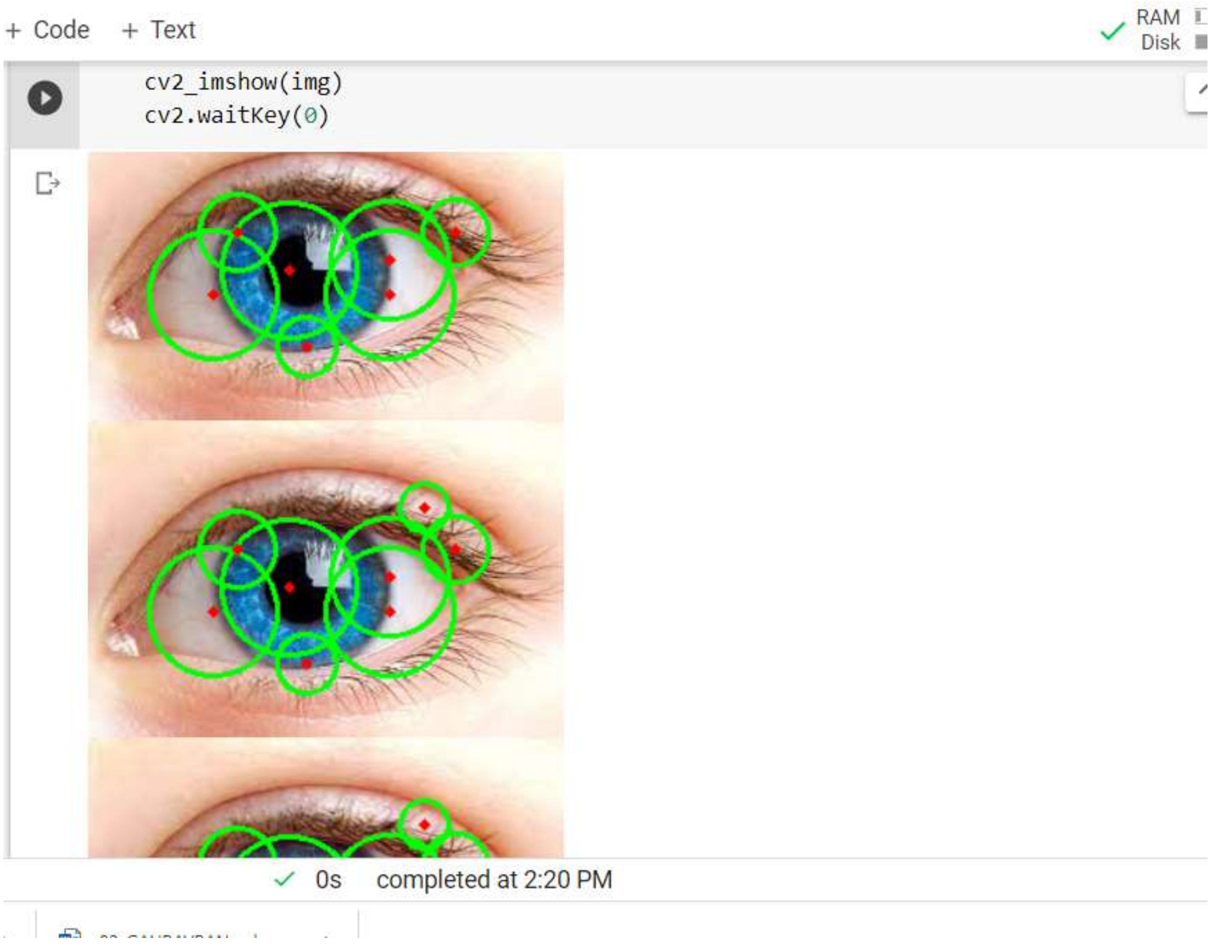


```
cv2_imshow(img)
cv2.waitKey(0)
```



```
cv2_imshow(img)
cv2.waitKey(0)
```

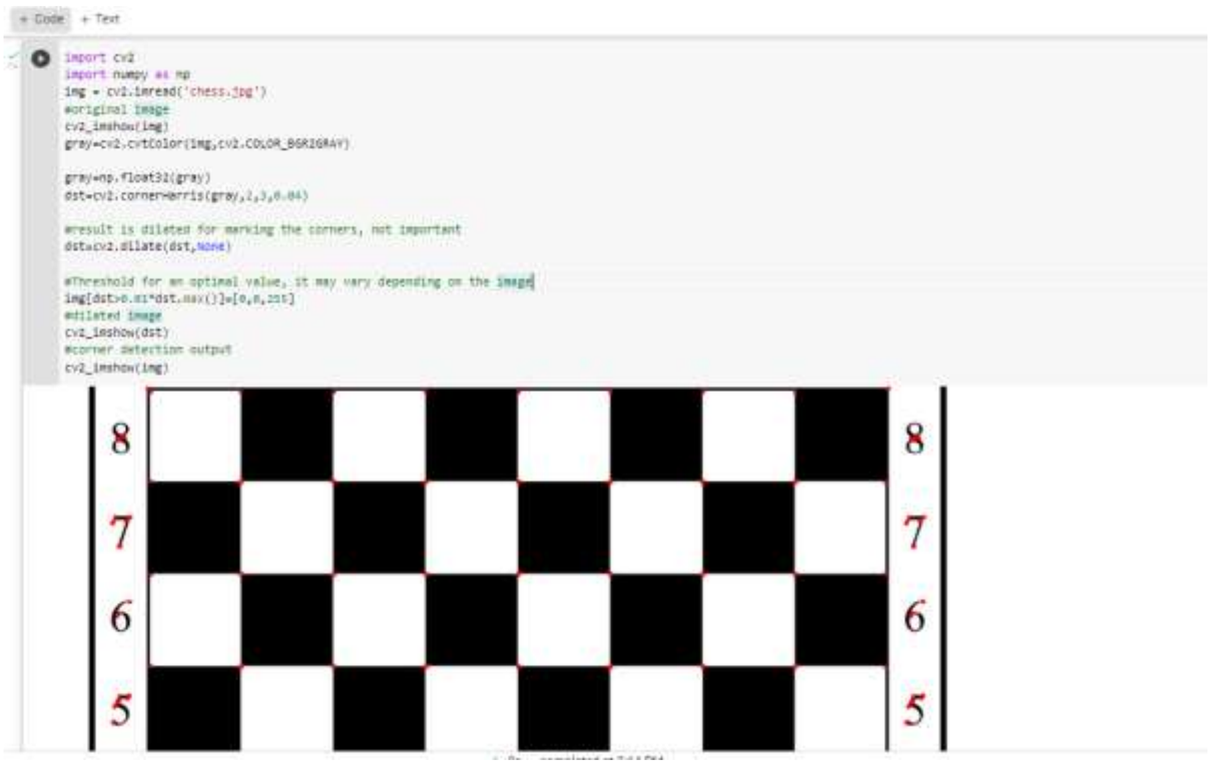
02_GAURAV BANE



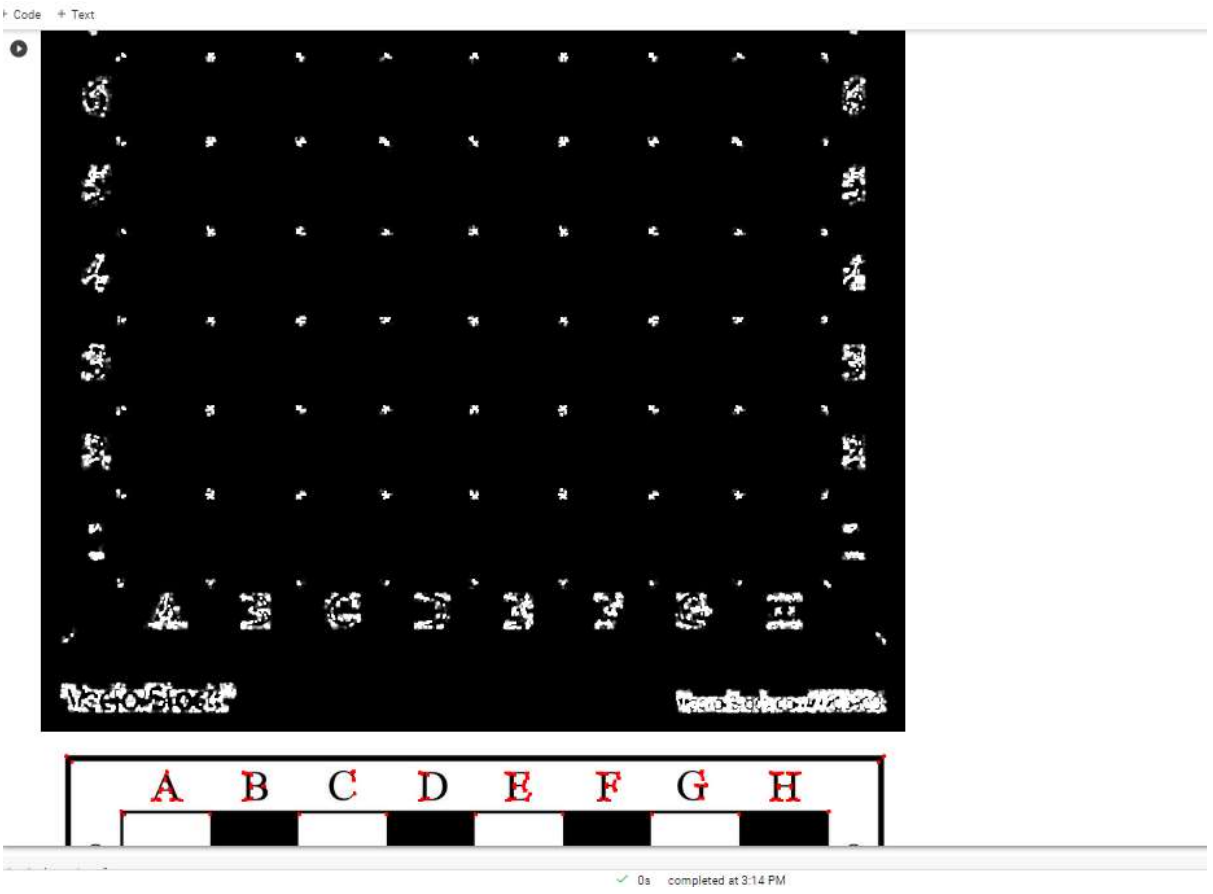
```
cv2_imshow(img)
cv2.waitKey(0)
```

```
import cv2
import numpy as np
img = cv2.imread('chess.jpg')
#original image
cv2_imshow(img)
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

gray=np.float32(gray)
dst=cv2.cornerHarris(gray,2,3,0.04)

#result is dilated for marking the corners, not important
dst=cv2.dilate(dst,None)

#Threshold for an optimal value, it may vary depending on the image
img[dst>0.01*dst.max()]=[0,0,255]
#dilated image
cv2_imshow(dst)
#corner detection output
cv2_imshow(img)
```

02_GAURAV BANE

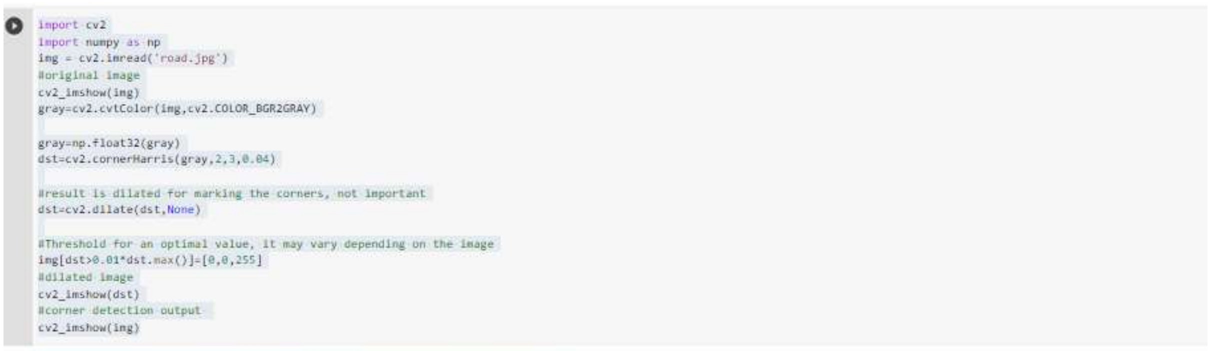


```
import cv2
import numpy as np
img = cv2.imread('chess.jpg')
#original image
cv2_imshow(img)
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

gray=np.float32(gray)
dst=cv2.cornerHarris(gray,2,3,0.04)

#result is dilated for marking the corners, not important
dst=cv2.dilate(dst,None)

#Threshold for an optimal value, it may vary depending on the image
img[dst>0.01*dst.max()]=[0,0,255]
#dilated image
cv2_imshow(dst)
#corner detection output
cv2_imshow(img)
```

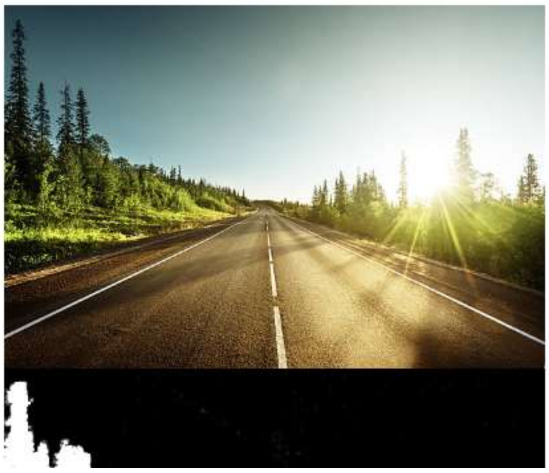


```
import cv2
import numpy as np
img = cv2.imread('road.jpg')
#original image
```

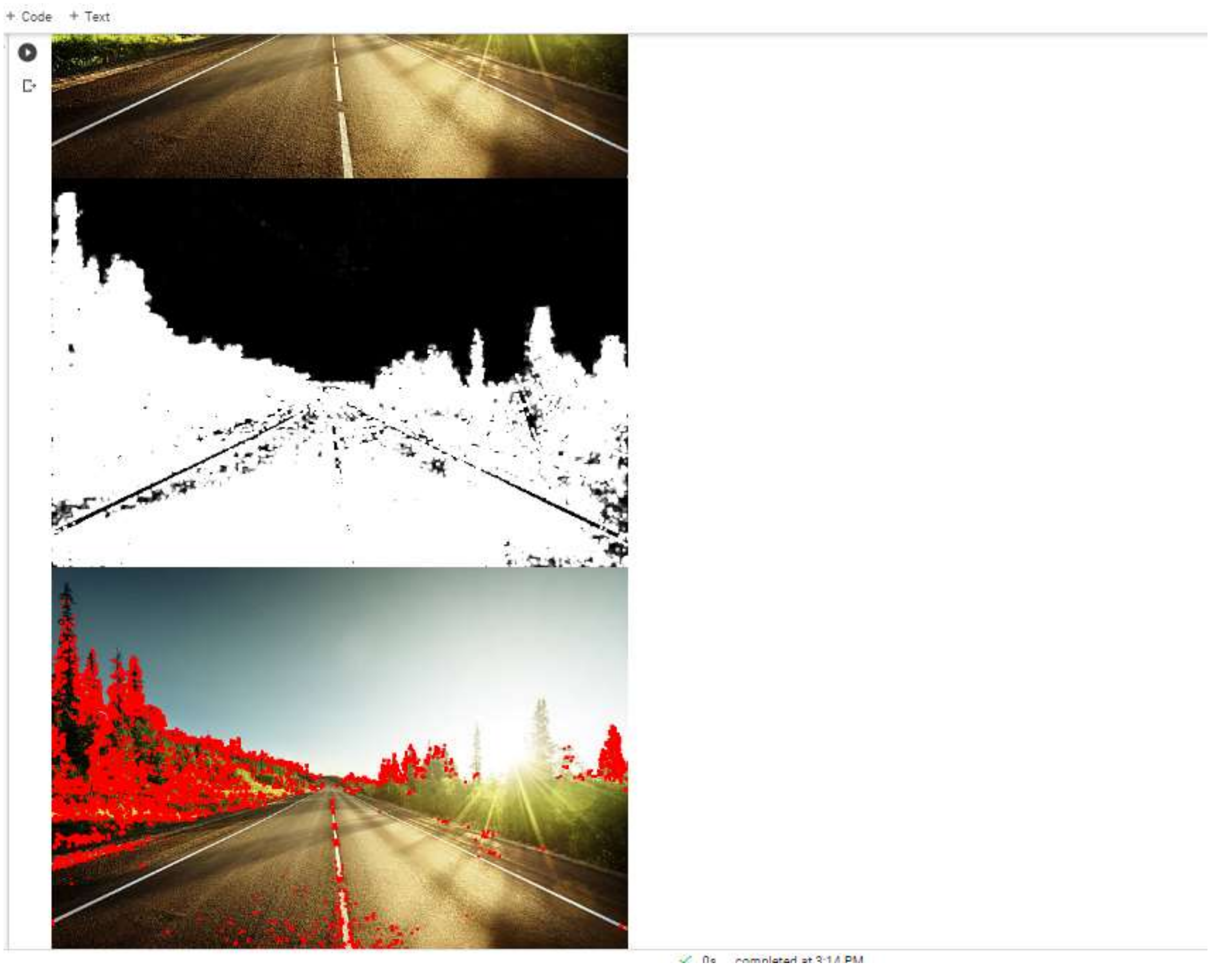
```
cv2_imshow(img)
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

gray=np.float32(gray)
dst=cv2.cornerHarris(gray,2,3,0.04)

#result is dilated for marking the corners, not important
dst=cv2.dilate(dst,None)

#Threshold for an optimal value, it may vary depending on the image
img[dst>0.01*dst.max()]=[0,0,255]
#dilated image
cv2_imshow(dst)
#corner detection output
cv2_imshow(img)
```



```
import cv2
import numpy as np
img = cv2.imread('road.jpg')
#original image
cv2_imshow(img)
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

gray=np.float32(gray)
dst=cv2.cornerHarris(gray,2,3,0.04)

#result is dilated for marking the corners, not important
dst=cv2.dilate(dst,None)

#Threshold for an optimal value, it may vary depending on the image
img[dst>0.01*dst.max()]=[0,0,255]
#dilated image
cv2_imshow(dst)
#corner detection output
cv2_imshow(img)
```

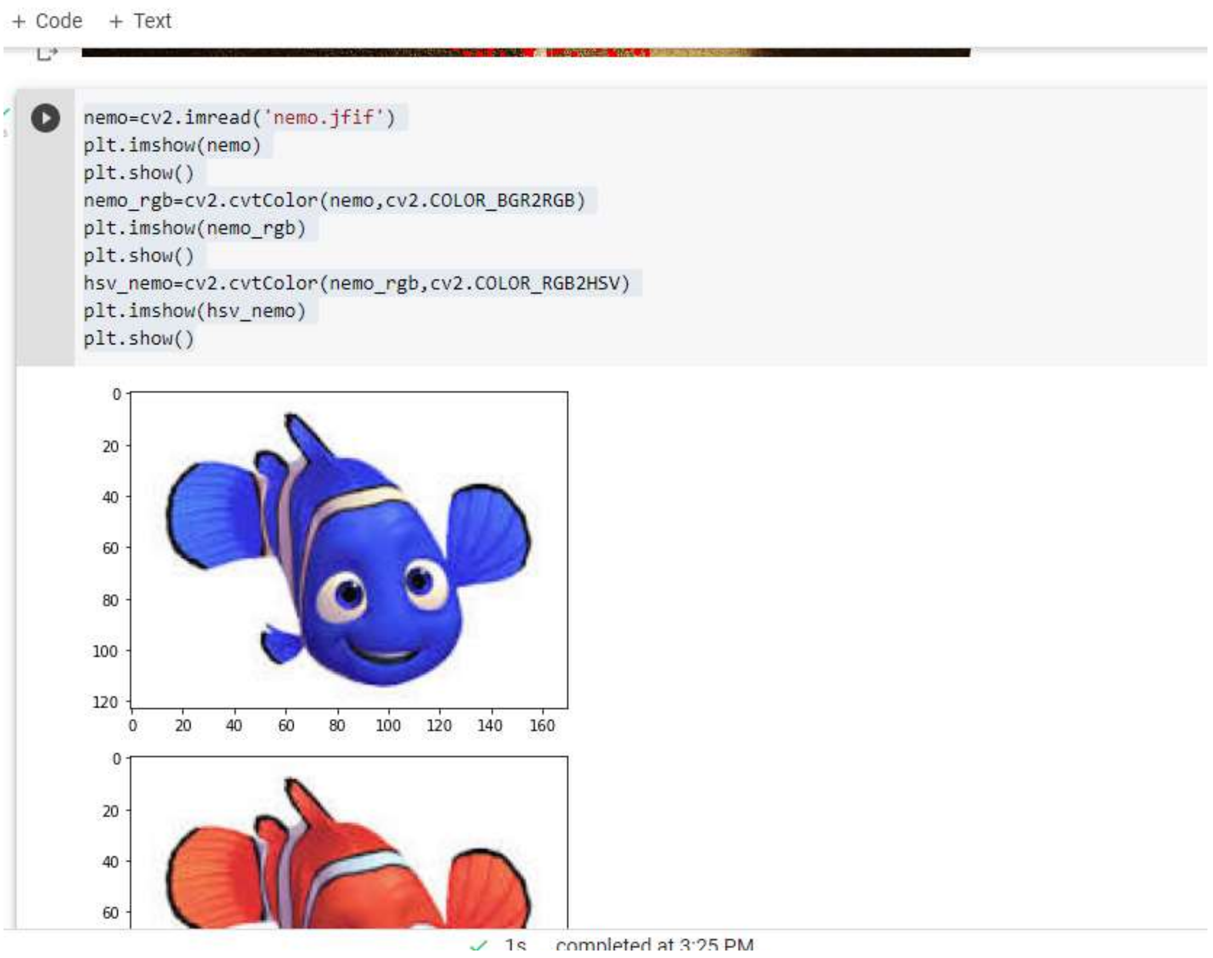


02_GAURAV BANE



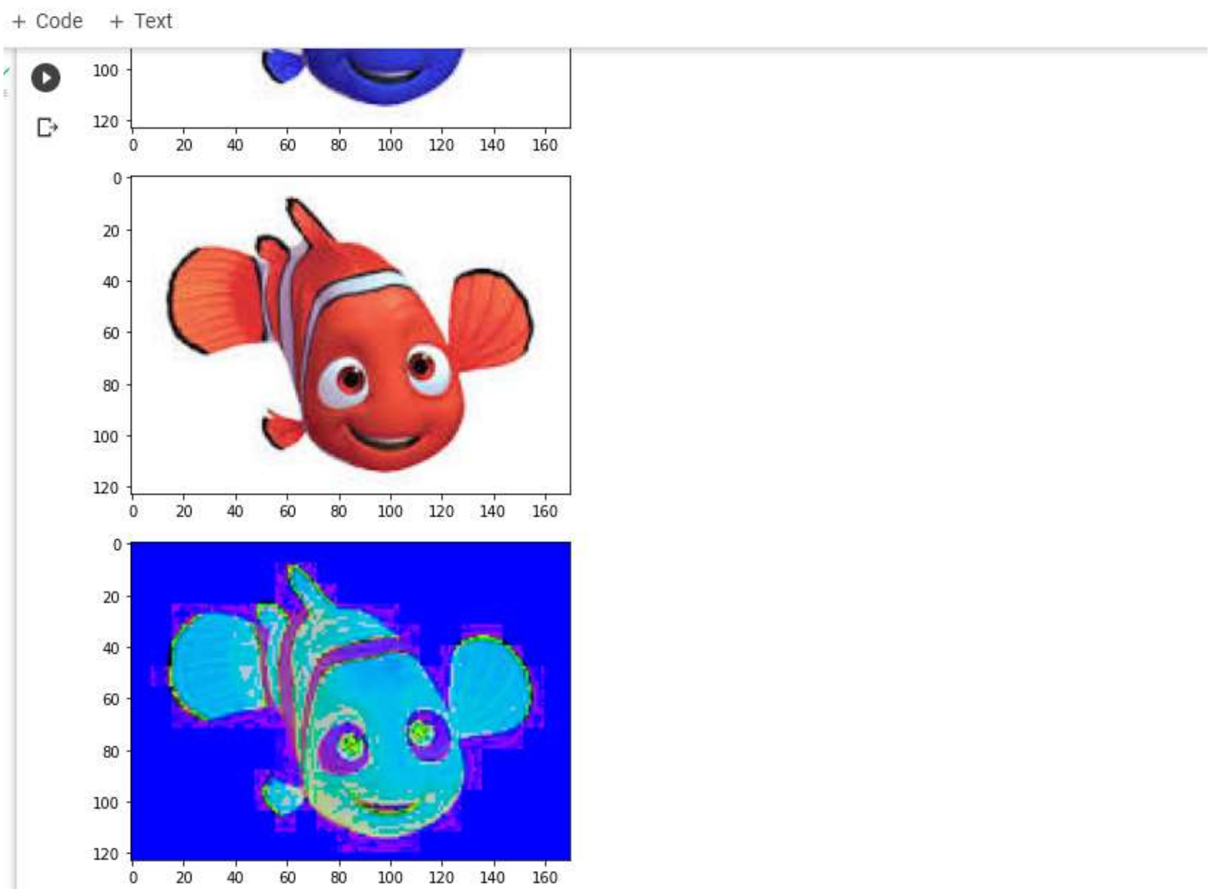
```
nemo=cv2.imread('nemo.jfif')
plt.imshow(nemo)
plt.show()
nemo_rgb=cv2.cvtColor(nemo,cv2.COLOR_BGR2RGB)
plt.imshow(nemo_rgb)
plt.show()
hsv_nemo=cv2.cvtColor(nemo_rgb,cv2.COLOR_RGB2HSV)
plt.imshow(hsv_nemo)
plt.show()
```

02_GAURAV BANE

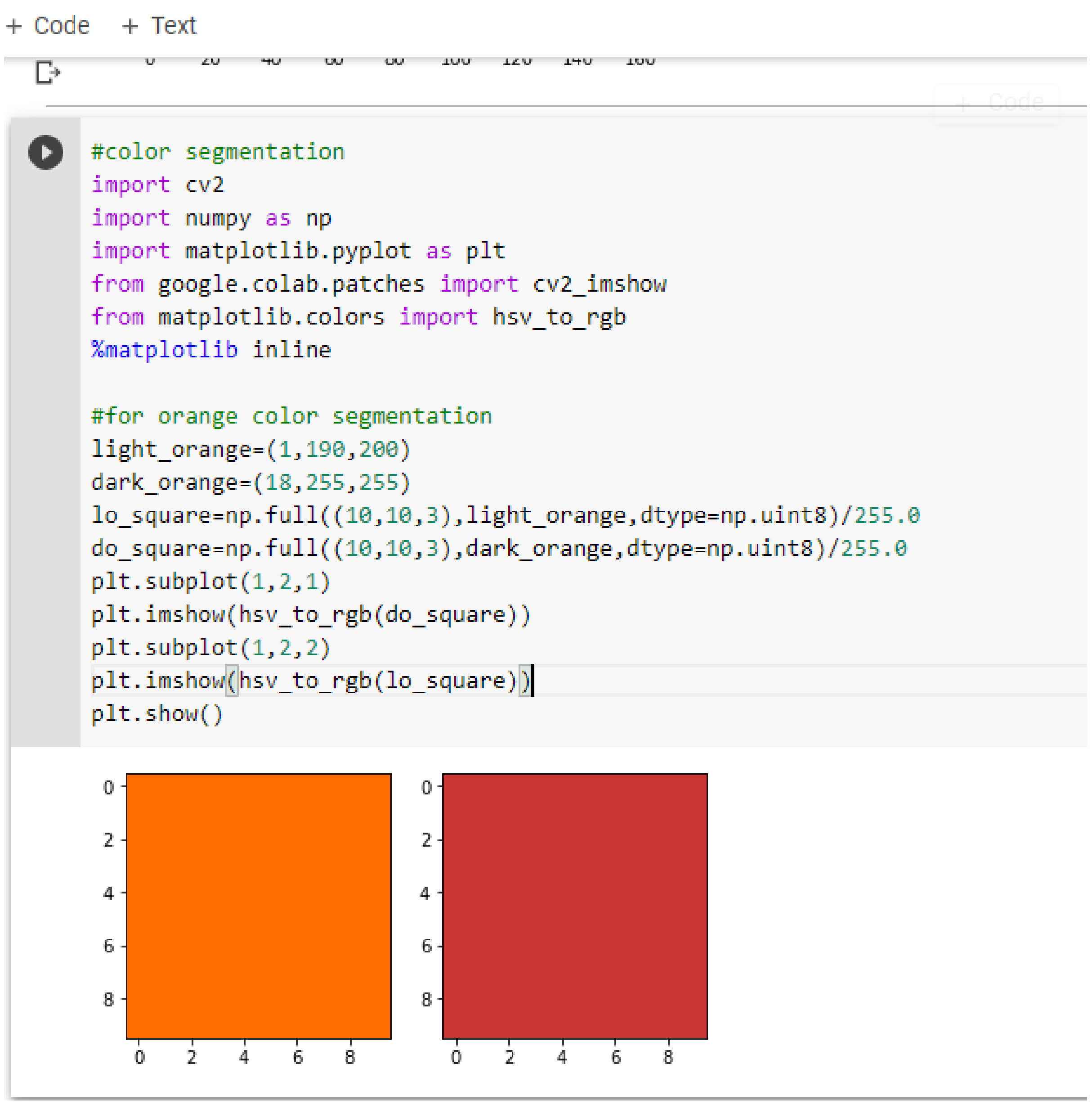


```
nemo=cv2.imread('nemo.jfif')
plt.imshow(nemo)
plt.show()
nemo_rgb=cv2.cvtColor(nemo,cv2.COLOR_BGR2RGB)
plt.imshow(nemo_rgb)
plt.show()
hsv_nemo=cv2.cvtColor(nemo_rgb,cv2.COLOR_RGB2HSV)
plt.imshow(hsv_nemo)
plt.show()
```

02_GAURAV BANE



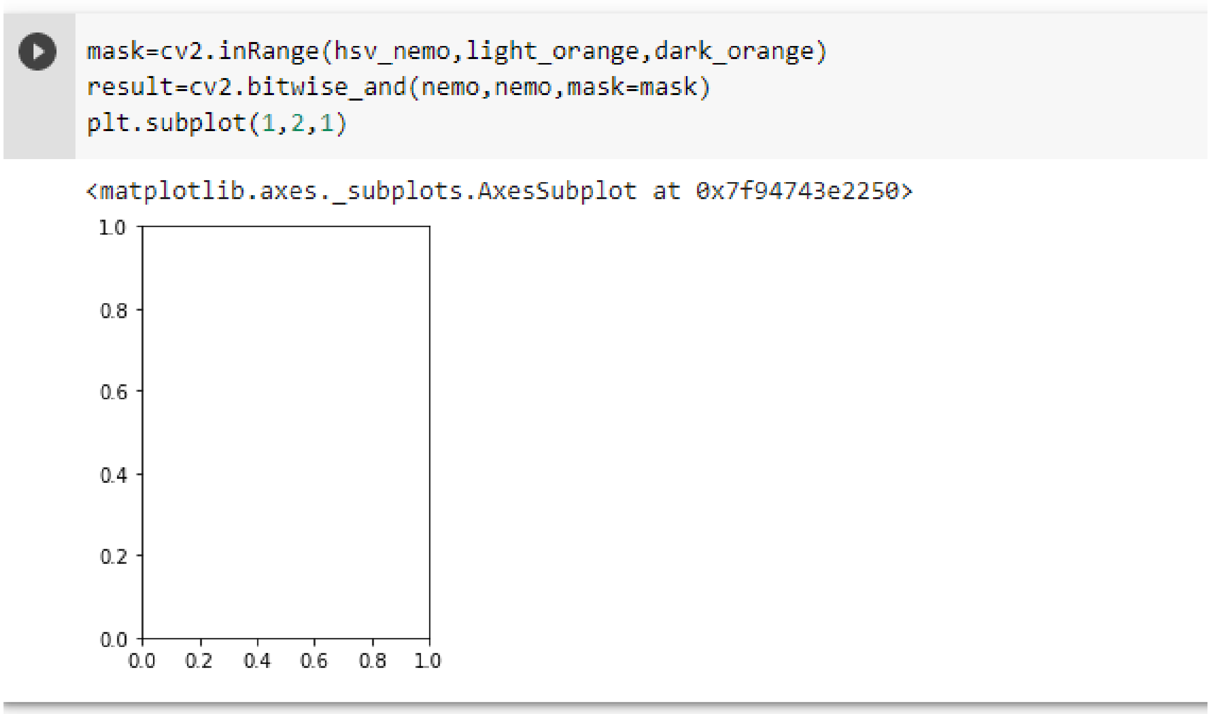
```
#color segmentation
import cv2
import numpy as np
import matplotlib.pyplot as plt
from google.colab.patches import cv2_imshow
from matplotlib.colors import hsv_to_rgb
%matplotlib inline

#for orange color segmentation
light_orange=(1,190,200)
dark_orange=(18,255,255)
lo_square=np.full((10,10,3),light_orange,dtype=np.uint8)/255.0
do_square=np.full((10,10,3),dark_orange,dtype=np.uint8)/255.0
plt.subplot(1,2,1)
plt.imshow(hsv_to_rgb(do_square))
plt.subplot(1,2,2)
plt.imshow(hsv_to_rgb(lo_square))
plt.show()
```



```
#color segmentation
import cv2
import numpy as np
import matplotlib.pyplot as plt
from google.colab.patches import cv2_imshow
from matplotlib.colors import hsv_to_rgb
%matplotlib inline

#for orange color segmentation
light_orange=(1,190,200)
dark_orange=(18,255,255)
lo_square=np.full((10,10,3),light_orange,dtype=np.uint8)/255.0
do_square=np.full((10,10,3),dark_orange,dtype=np.uint8)/255.0
plt.subplot(1,2,1)
plt.imshow(hsv_to_rgb(do_square))
plt.subplot(1,2,2)
plt.imshow(hsv_to_rgb(lo_square))
plt.show()
```

```
mask=cv2.inRange(hsv_nemo,light_orange,dark_orange)
result=cv2.bitwise_and(nemo,nemo,mask=mask)
plt.subplot(1,2,1)
```

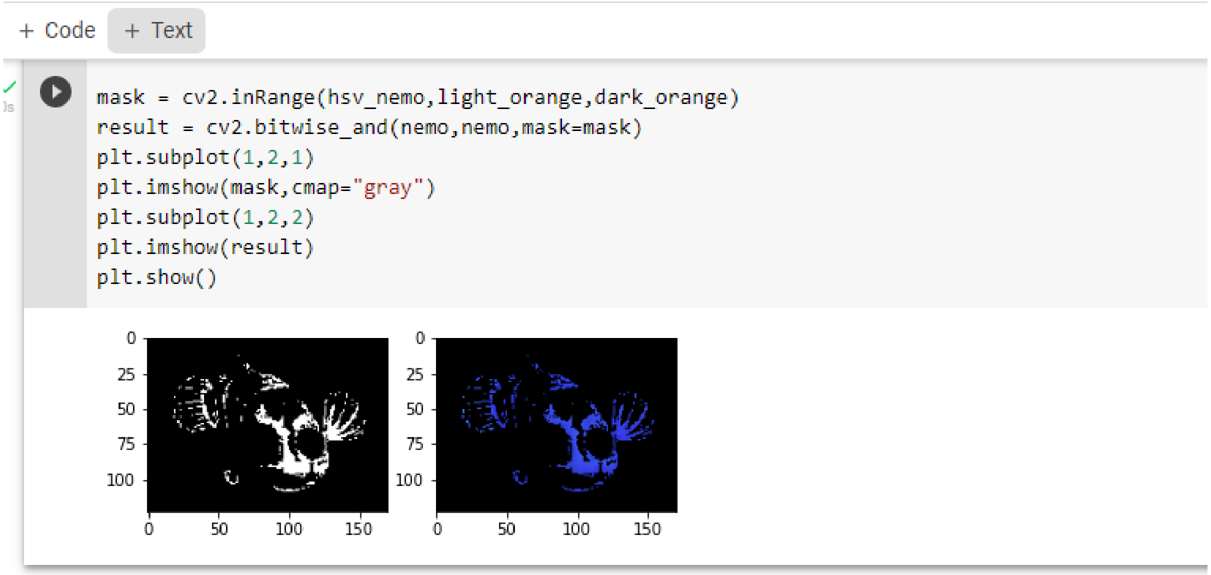
02_GAURAV BANE



```
mask=cv2.inRange(hsv_nemo,light_orange,dark_orange)
result=cv2.bitwise_and(nemo,nemo,mask=mask)
plt.subplot(1,2,1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f94743e2250>
```

```
mask = cv2.inRange(hsv_nemo,light_orange,dark_orange)
result = cv2.bitwise_and(nemo,nemo,mask=mask)
plt.subplot(1,2,1)
plt.imshow(mask,cmap="gray")
plt.subplot(1,2,2)
plt.imshow(result)
plt.show()
```



```
mask = cv2.inRange(hsv_nemo,light_orange,dark_orange)
result = cv2.bitwise_and(nemo,nemo,mask=mask)
plt.subplot(1,2,1)
plt.imshow(mask,cmap="gray")
plt.subplot(1,2,2)
plt.imshow(result)
plt.show()
```