Nama: Akram Farrasanto

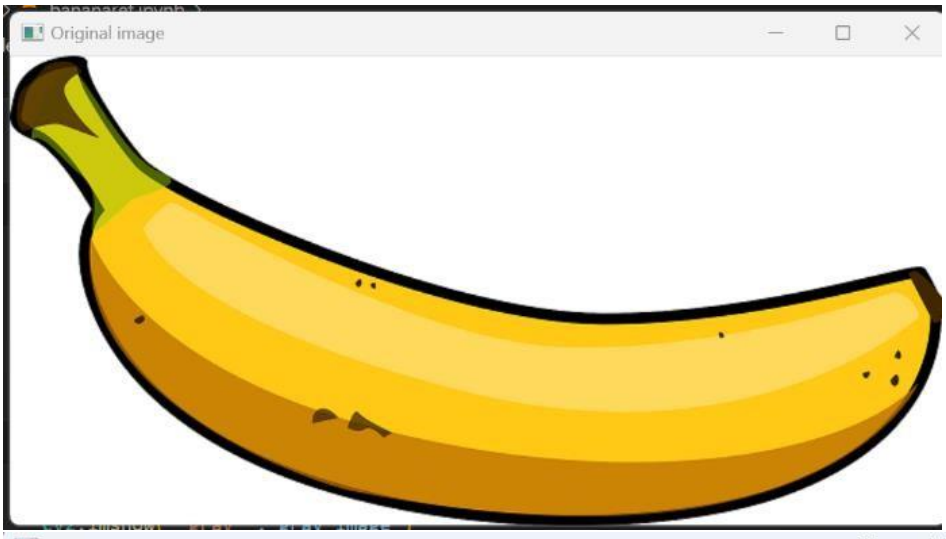
NIM: 312210245

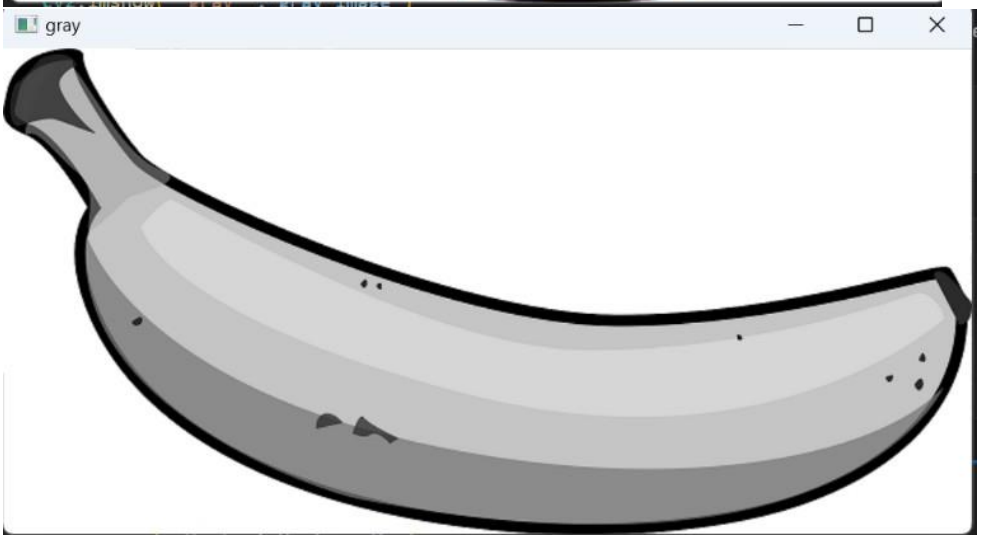
Kelas: TI.22.A.2

Bananaref.ipynb

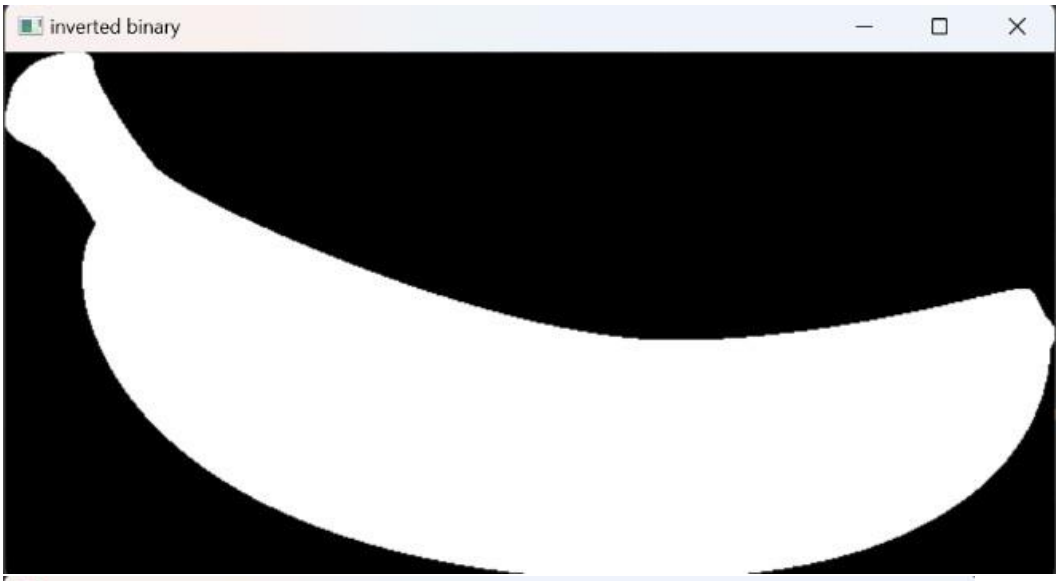
```python
# %%
import cv2

# %%
image = cv2.imread('data/bananaref.png')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()


```

Output:


Original image


gray

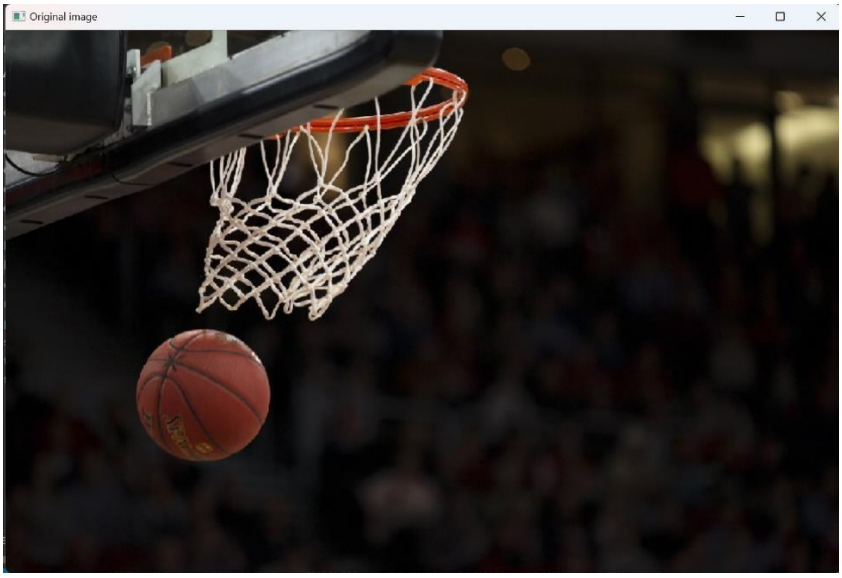

binary



Basketball.ipynb

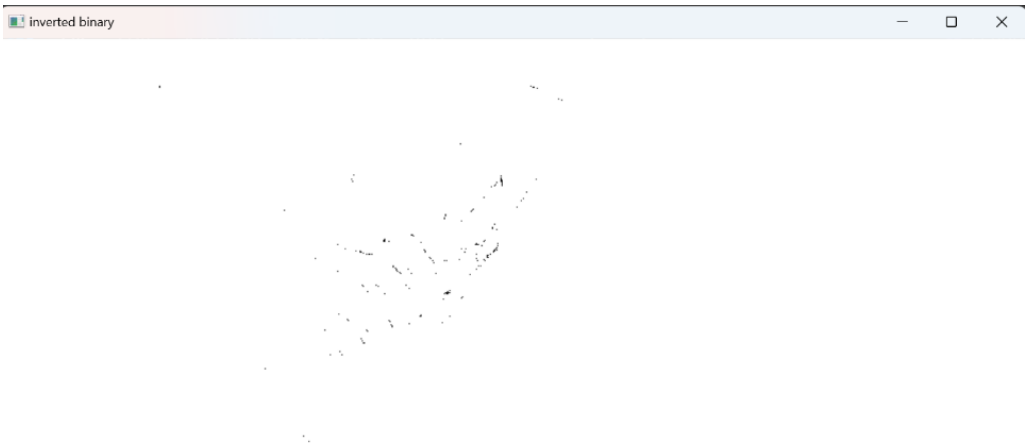
```python
# %%
import cv2

# %%
image = cv2.imread('data/basketball.jpg')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()
```

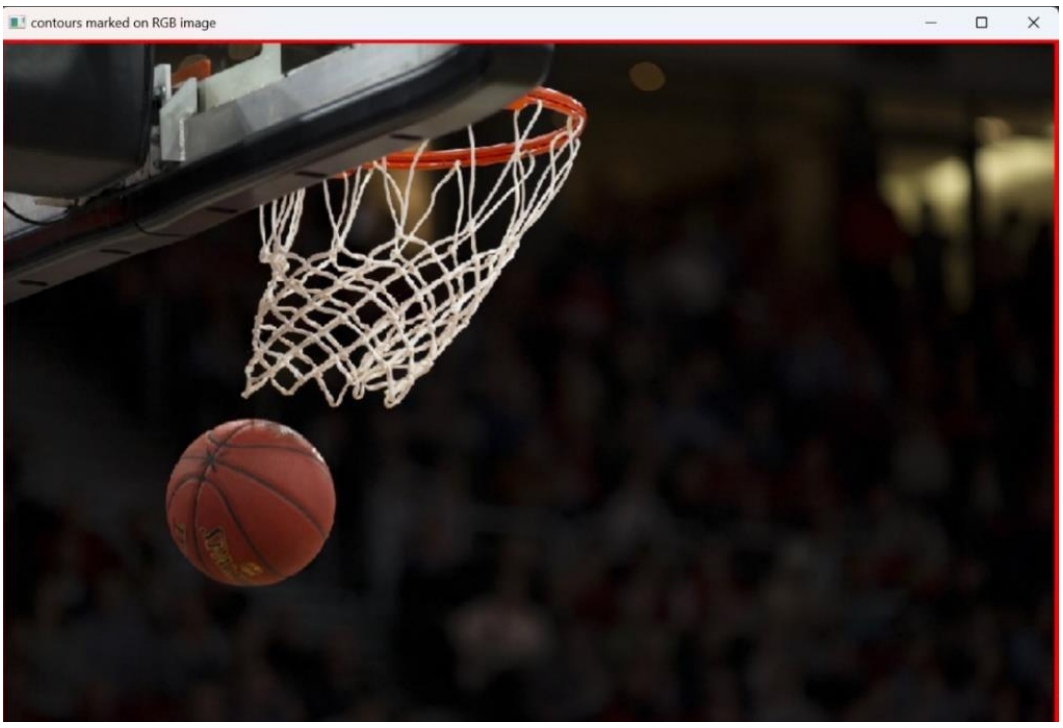
Output:




binary


gray



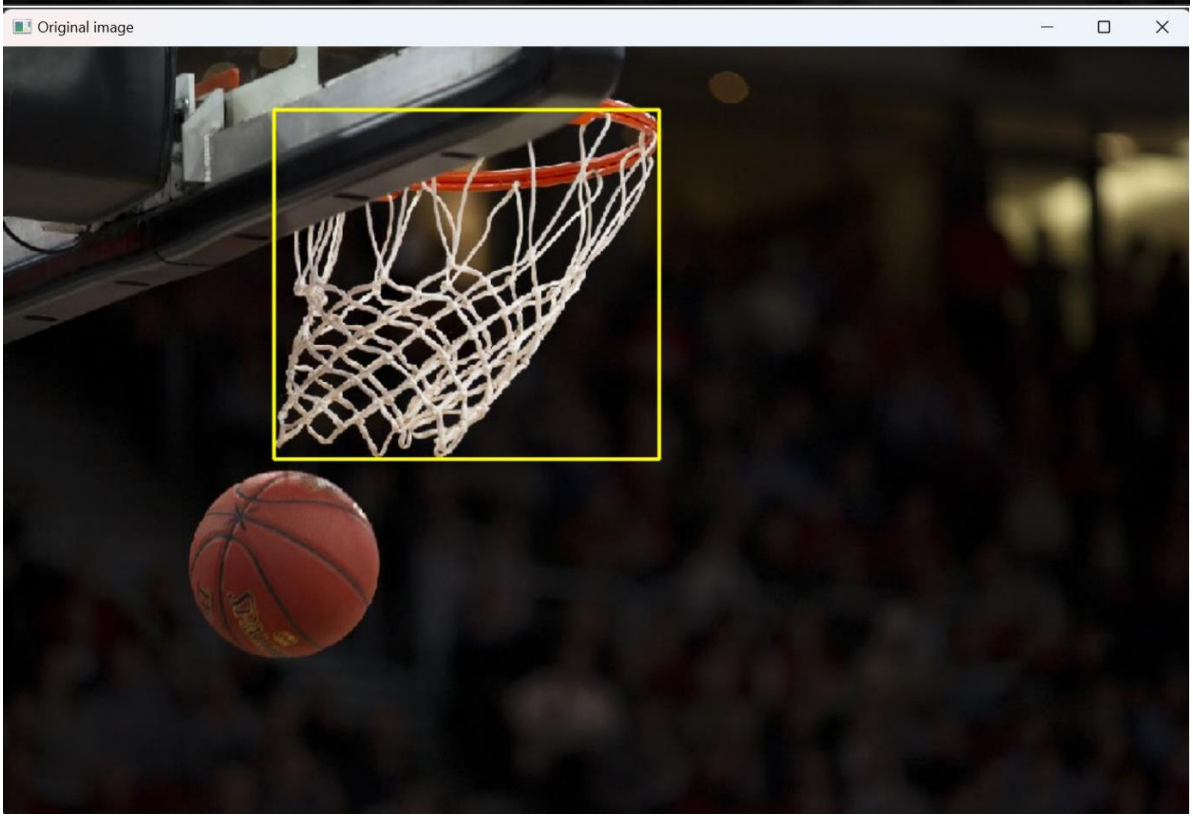


Detected basket.ipynb

```python
# %%
import cv2

# %%
image = cv2.imread('data/detected_basket.png')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()
```

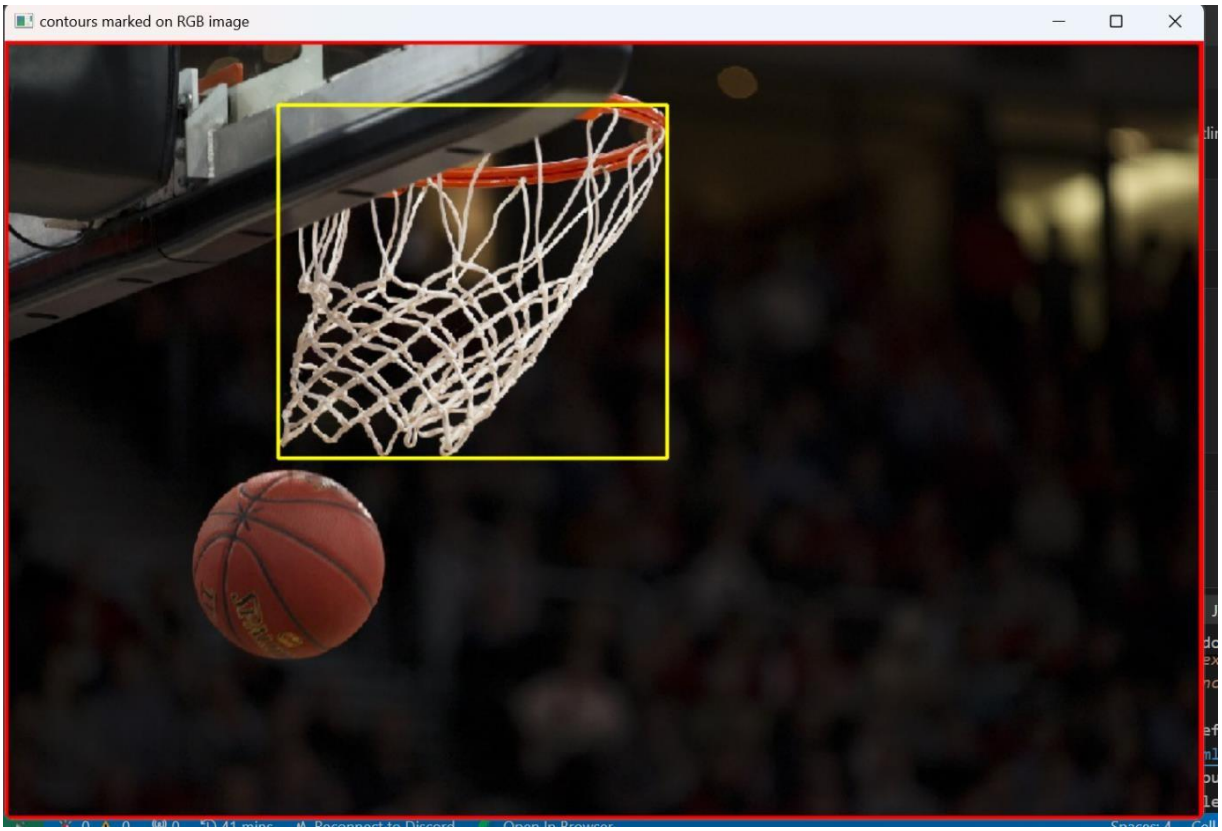
Output:


gray


Original image


binary

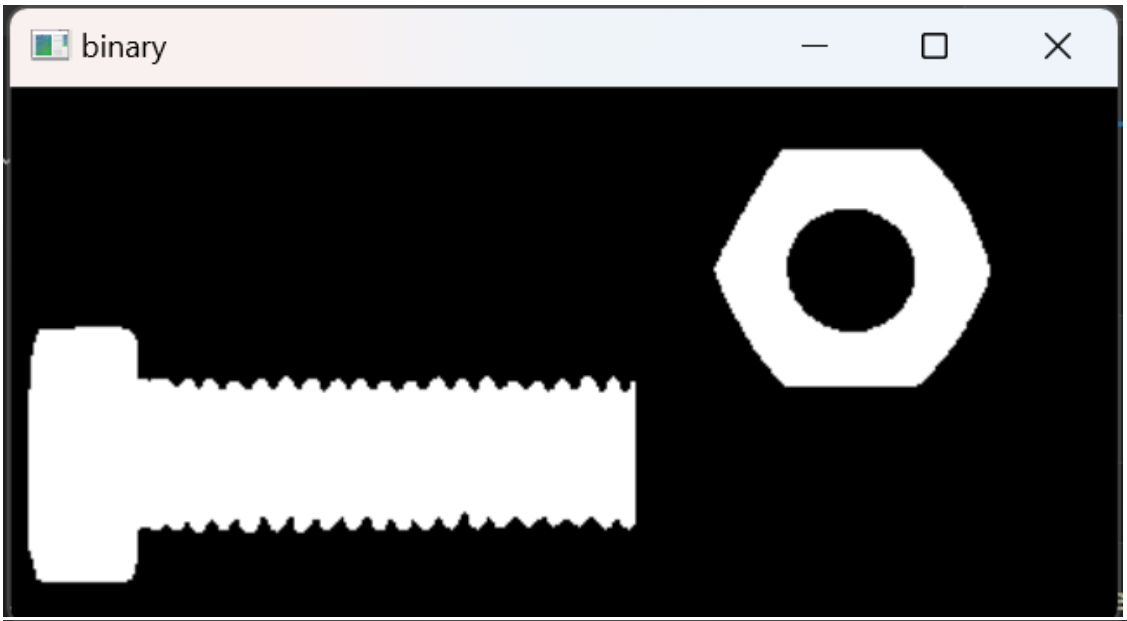

inverted binary

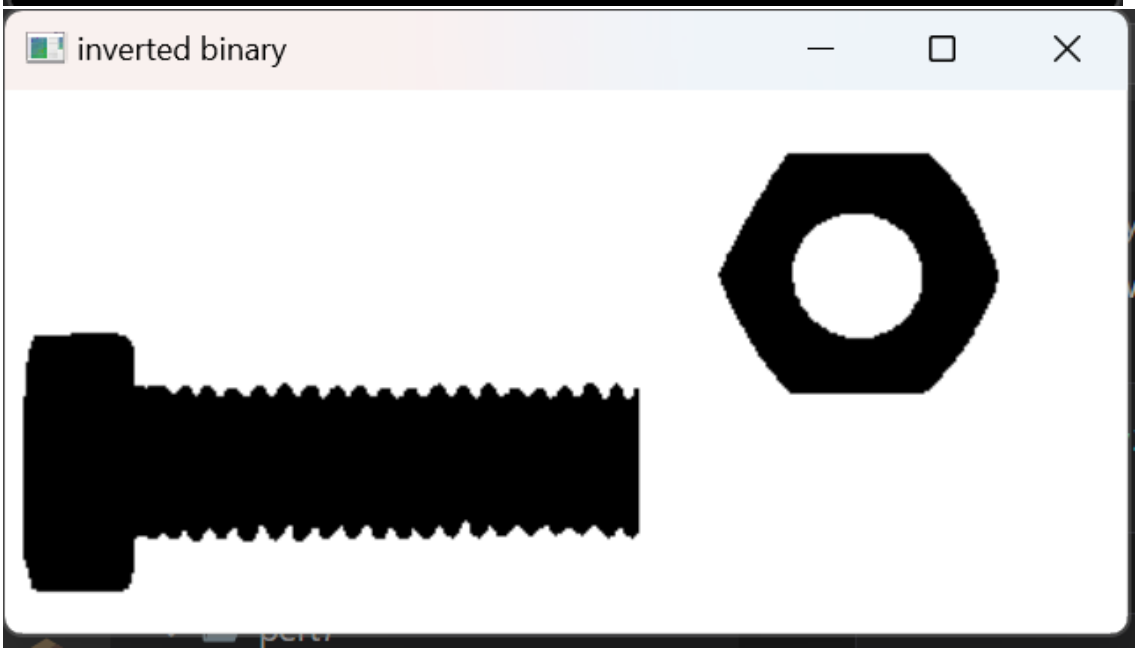


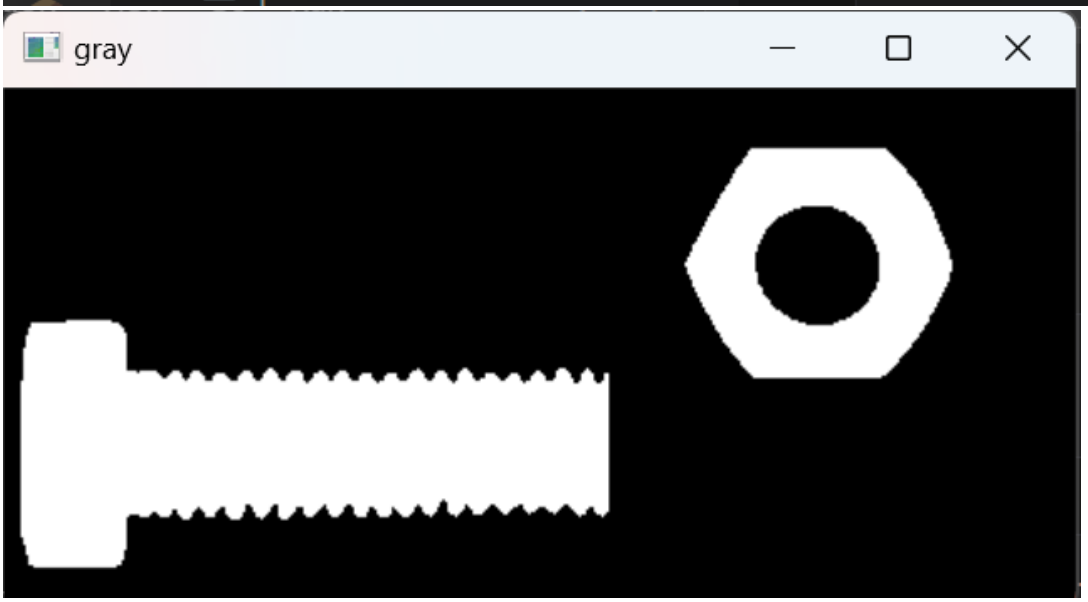
Nult_bolt.ipynb

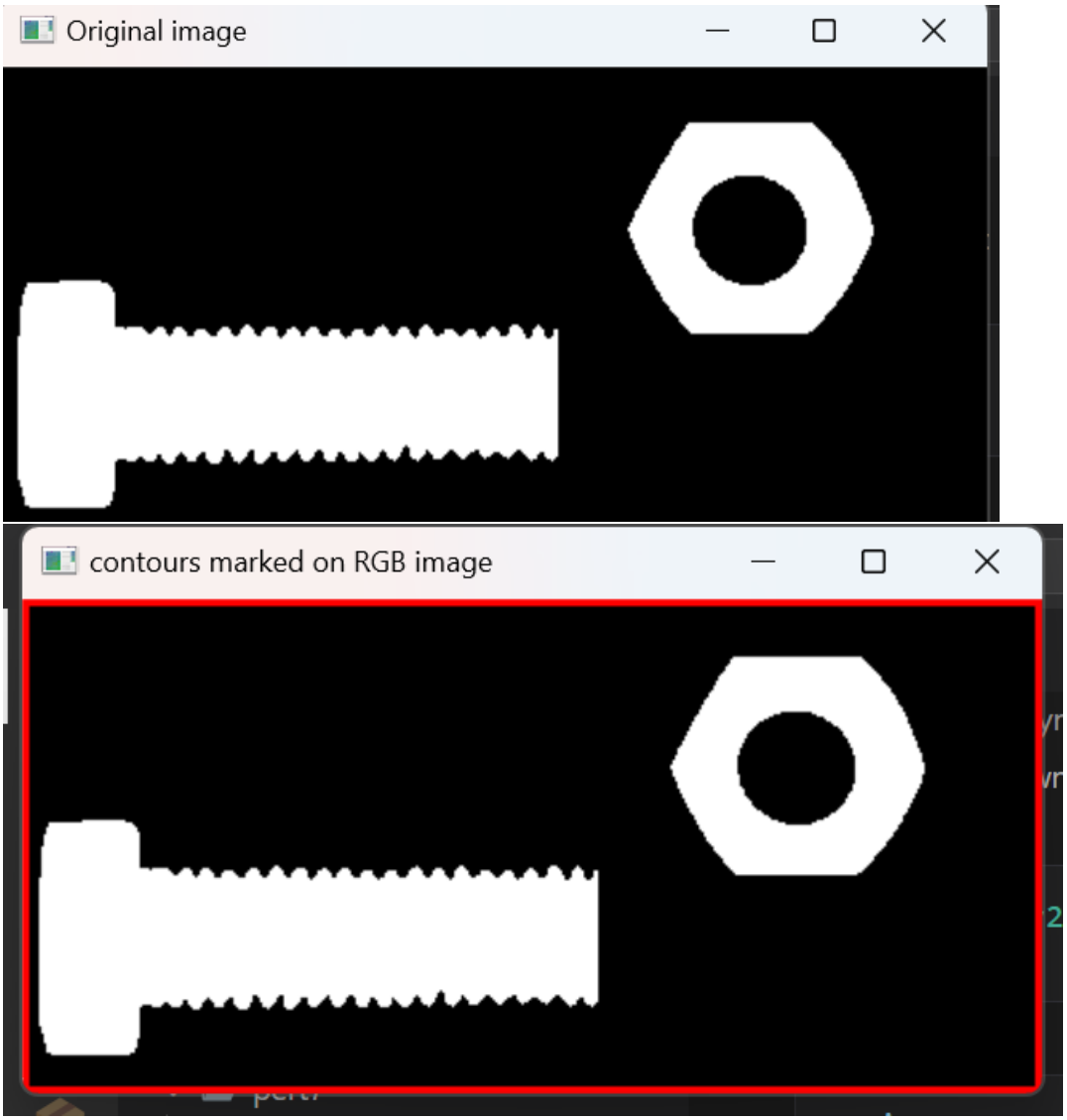
```python
# %%
import cv2

# %%
image = cv2.imread('data/nut_bolt.png')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()


```

Output:


binary


inverted binary


gray



Phrase_handwriting.ipynb

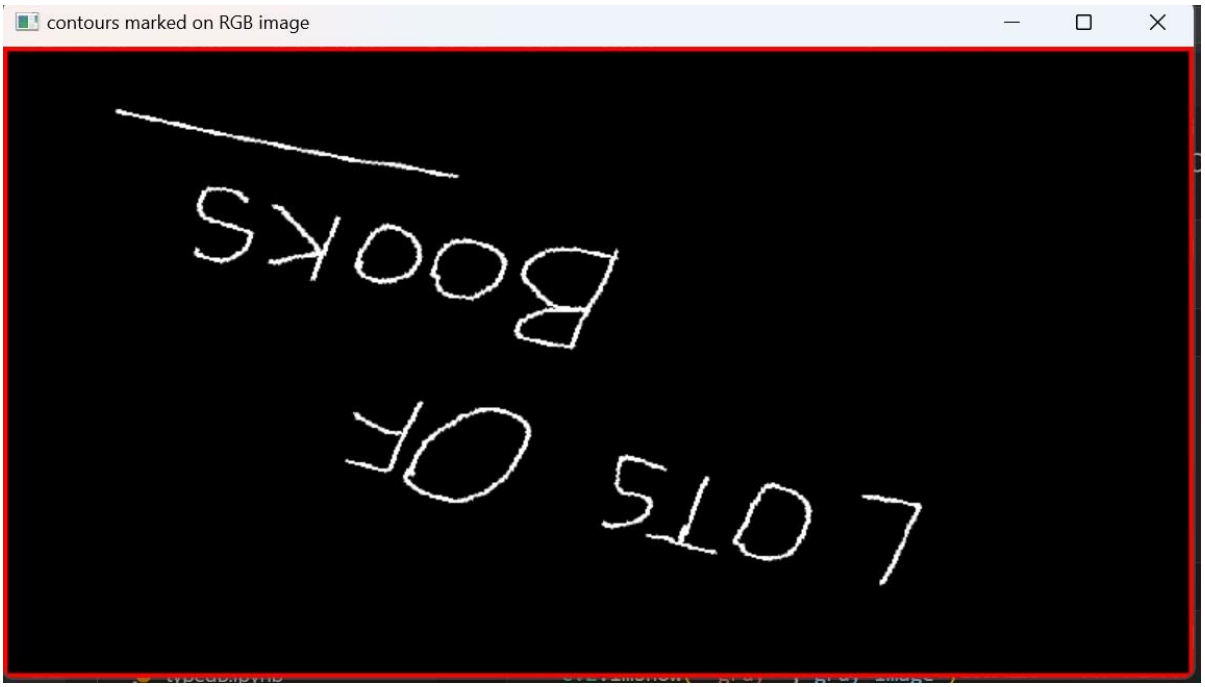
```python
# %%
import cv2

# %%
image = cv2.imread('data/phrase_handwritten.png')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()


```

Output:

gray
BOOKS
OF
LOTS
Original image
BOOKS
OF
LOTS

binary
BOOKS
OF
LOTS 7

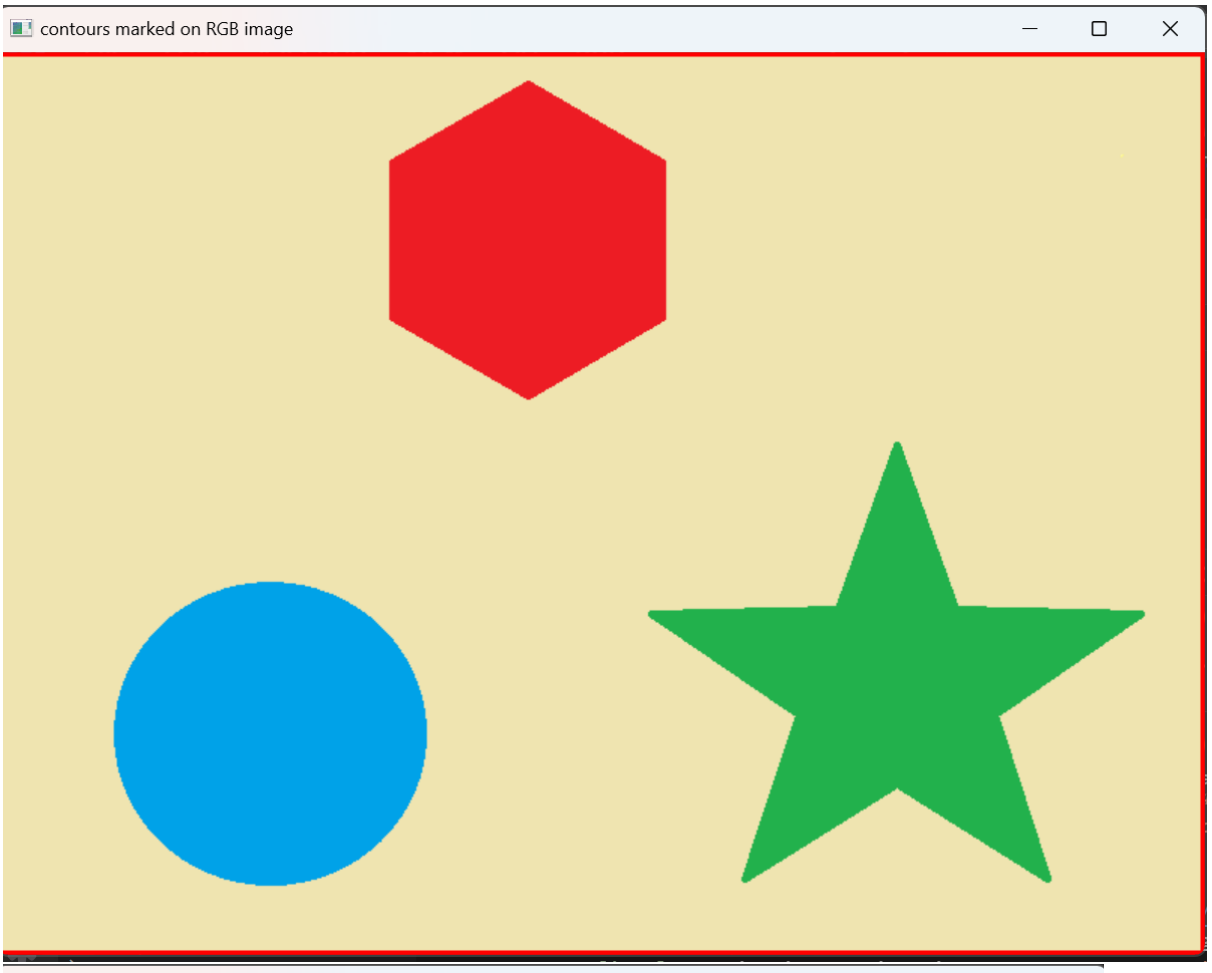
inverted binary
BOOKS
OF
LOTS 7

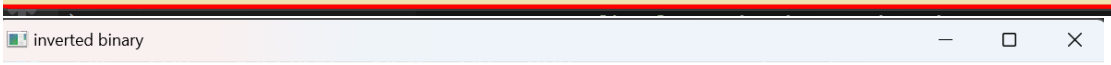


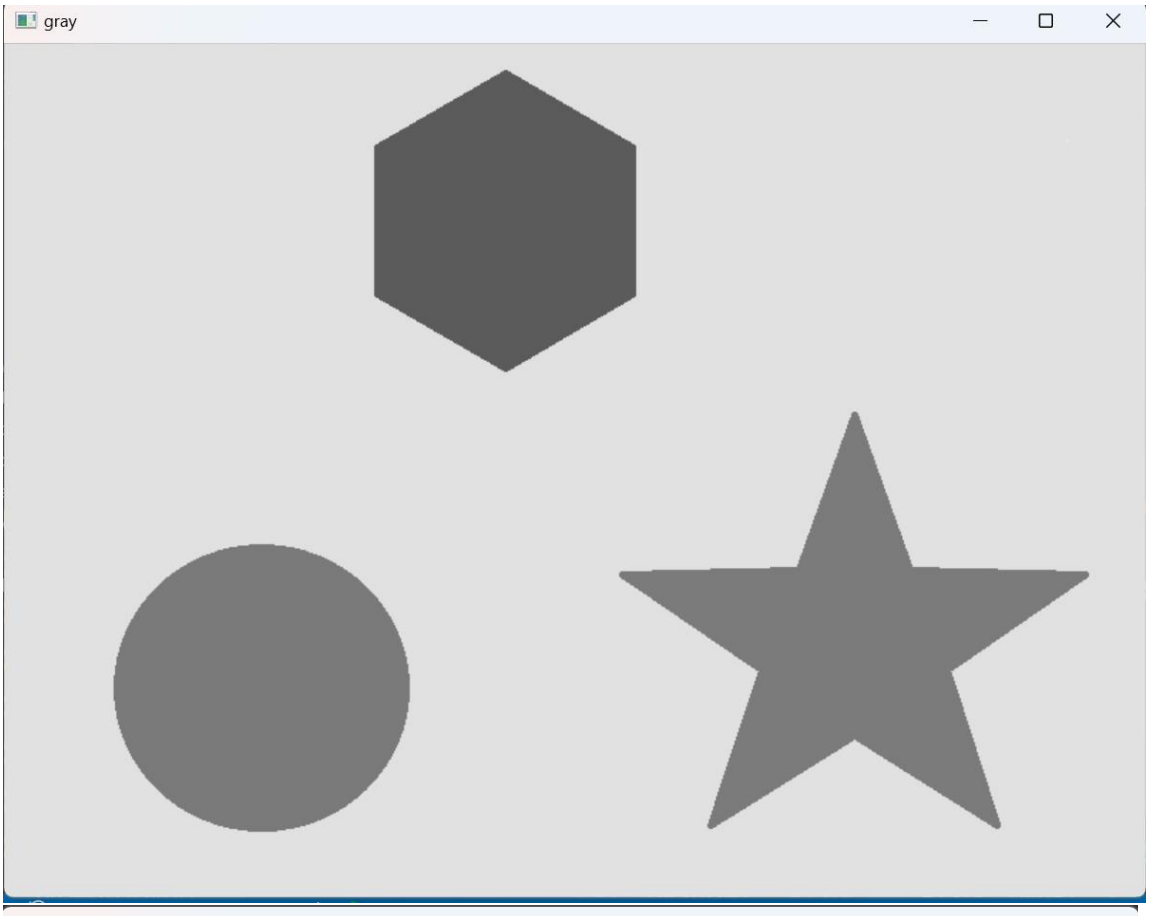
Sample_shapes.ipynb

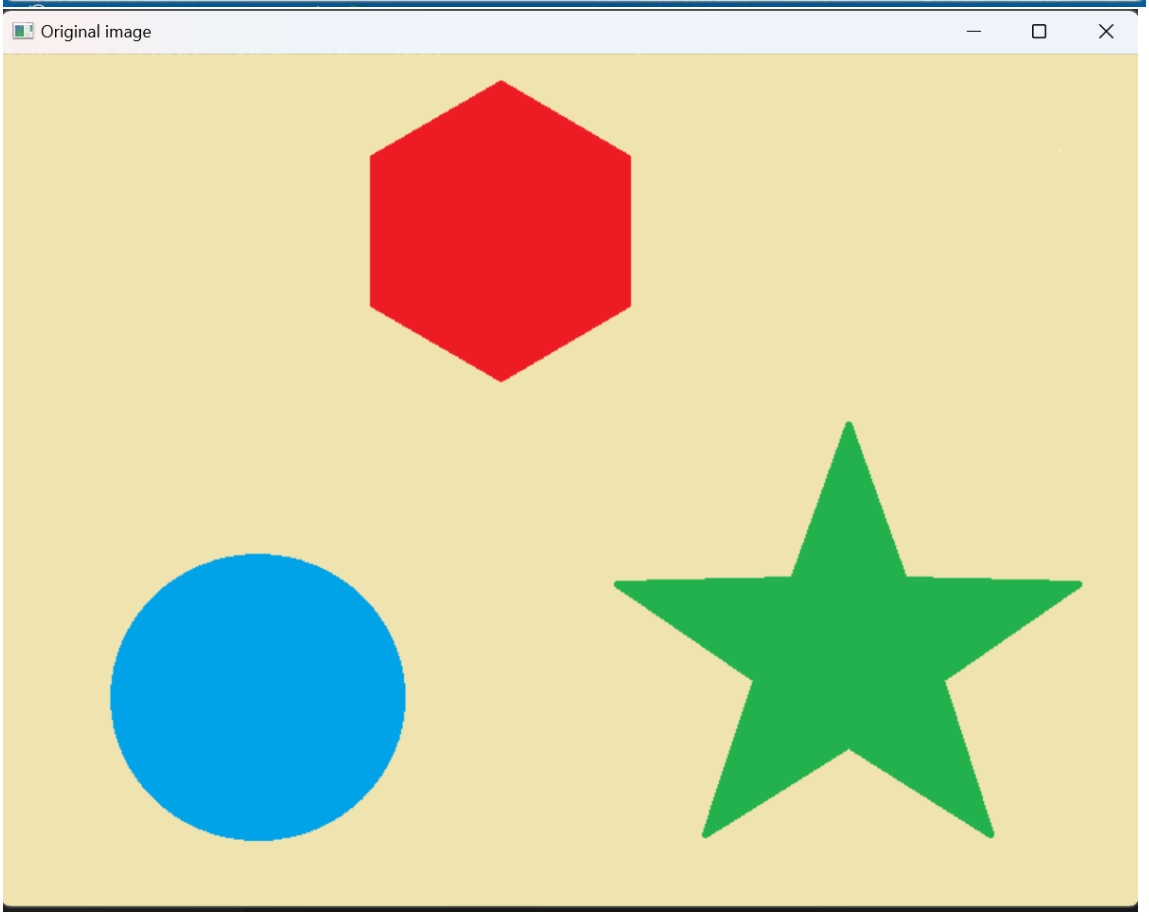
```python
# %%
import cv2

# %%
image = cv2.imread('data/sample_shapes.png')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()


```

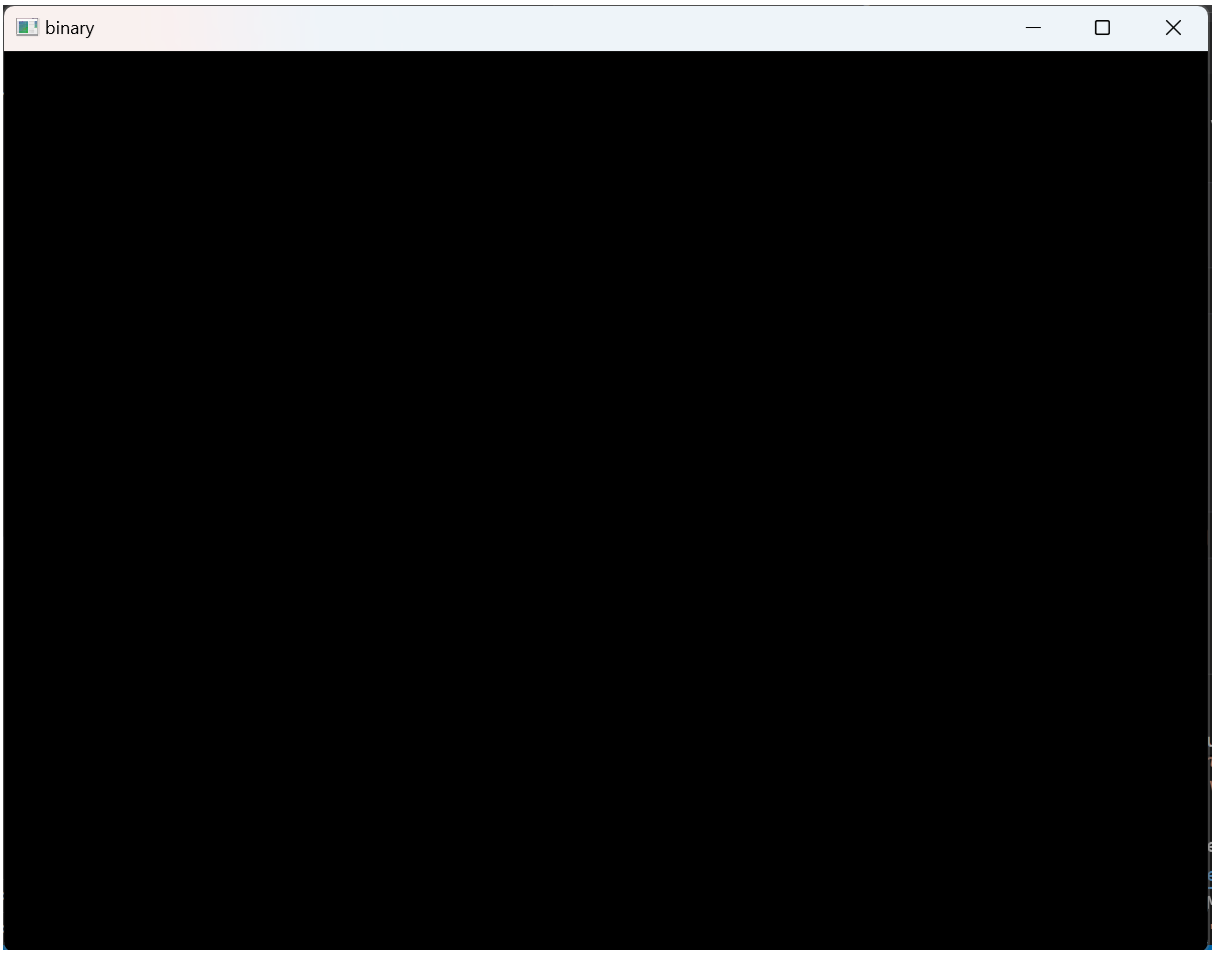
Output:


contours marked on RGB image


inverted binary


gray

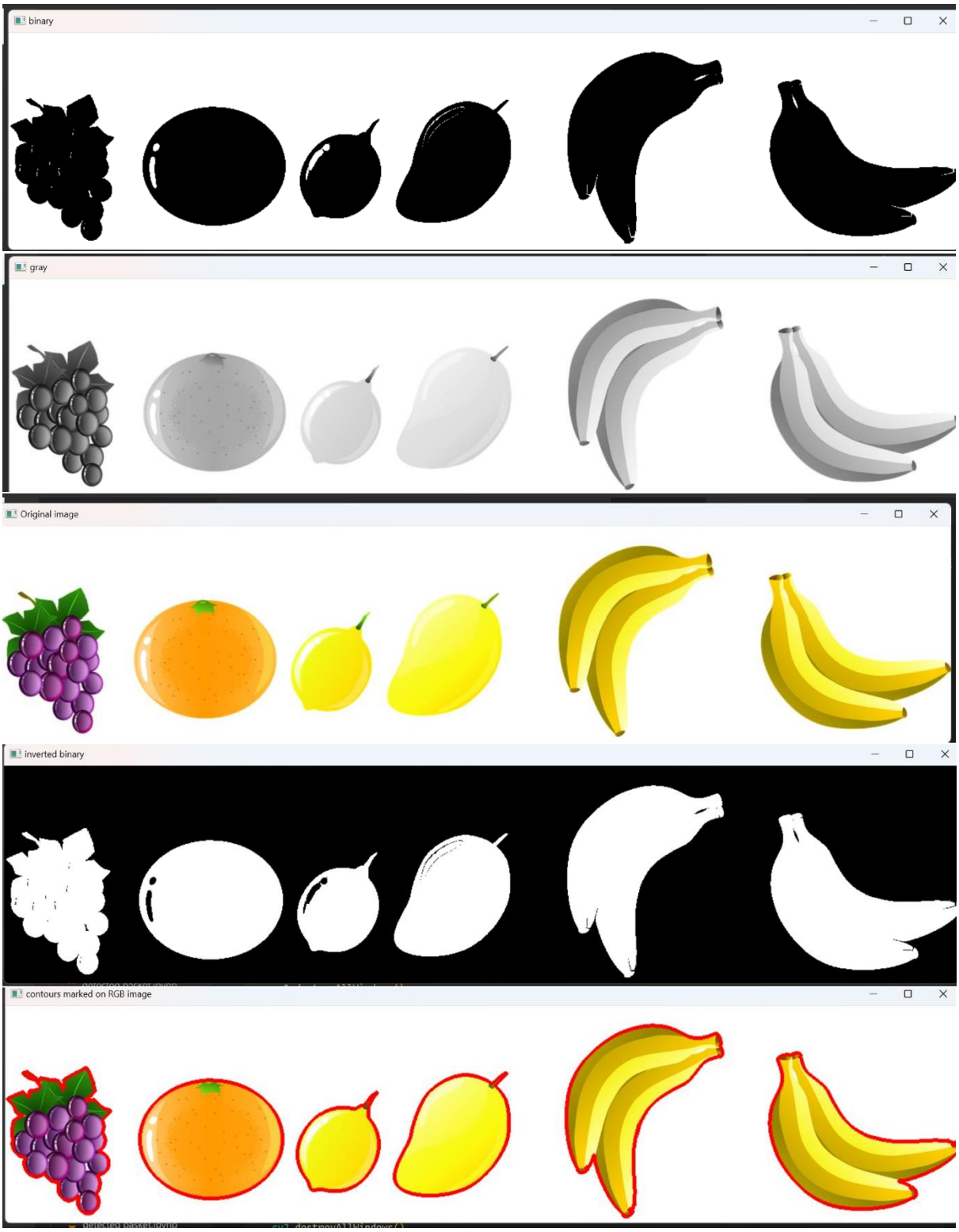

Original image

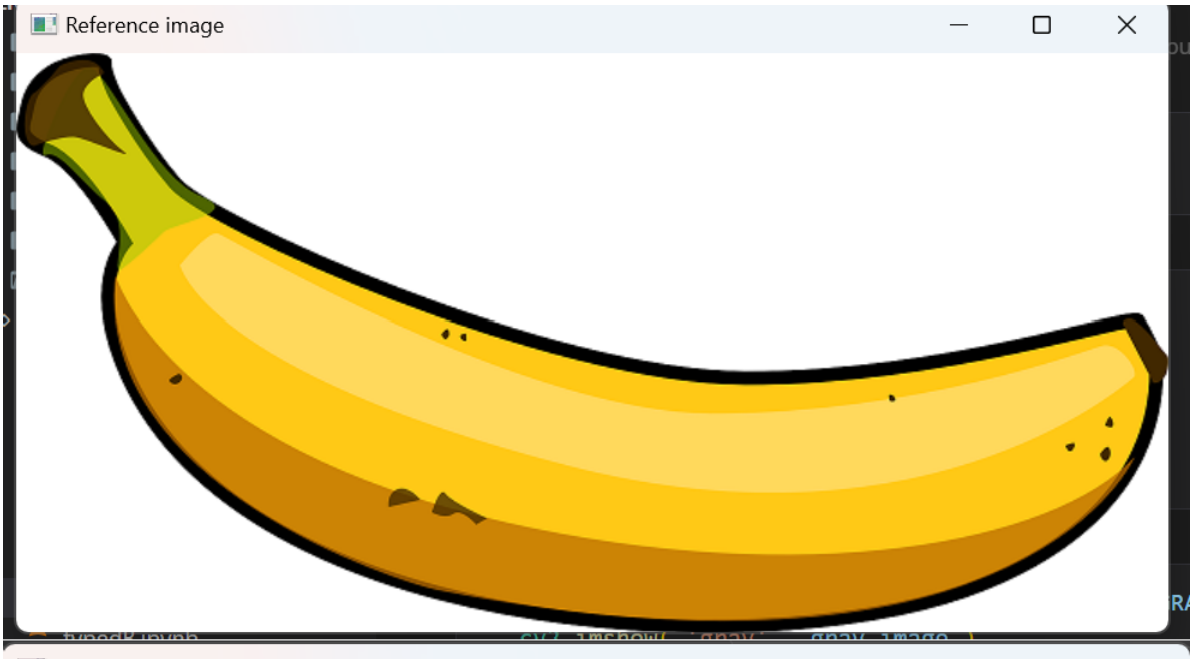


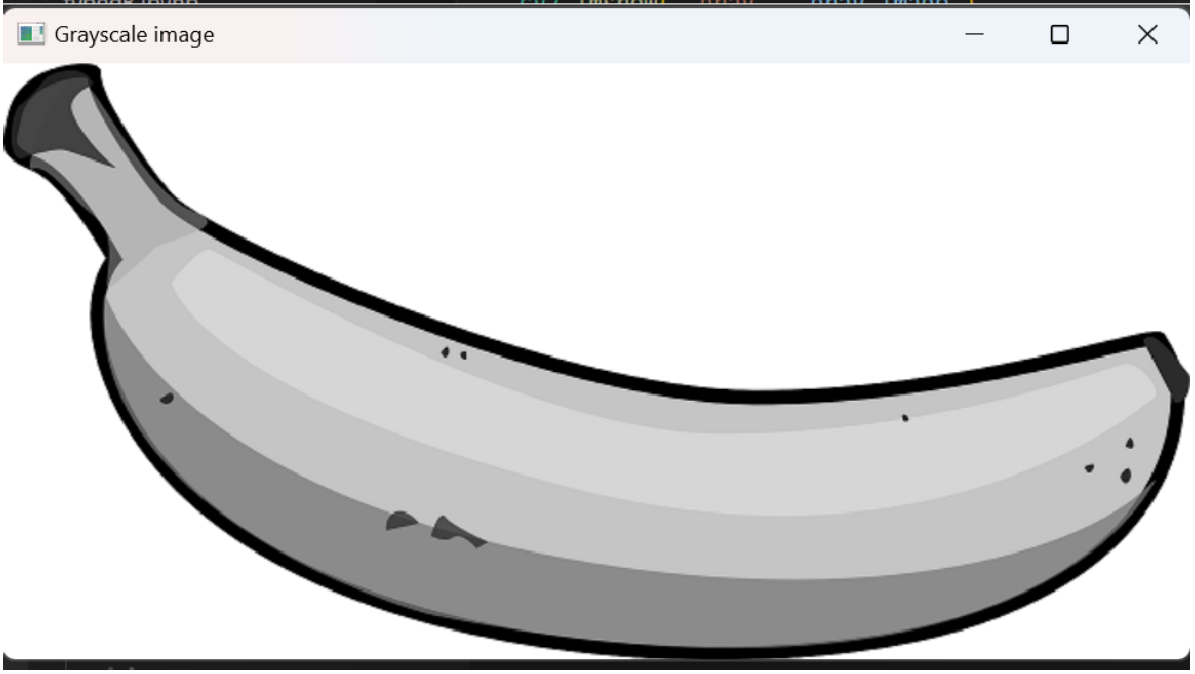
Many_fruit.ipynb

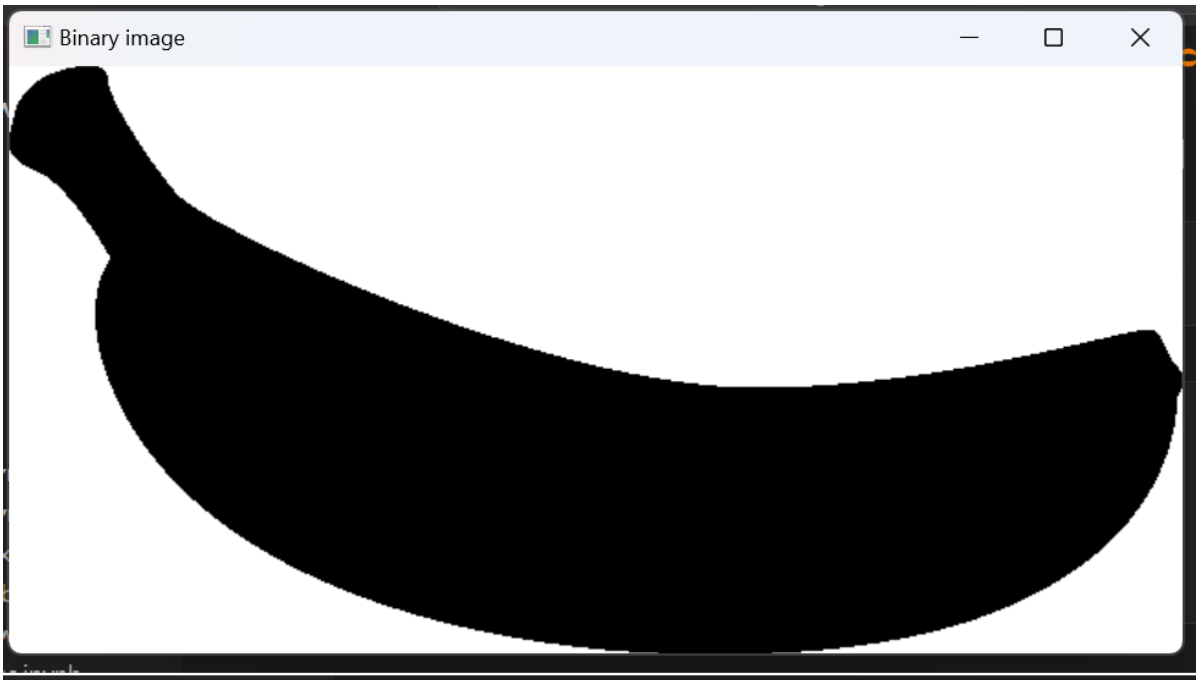
```python
# %%
import cv2

# %%
image = cv2.imread('data/many fruits.png')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ref_image = cv2.imread('data/bananaref.png')
cv2.imshow( 'Reference image' , ref_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(ref_image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'Grayscale image' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'Binary image' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary image' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
dist_list = []
for cnt in contours:
    retval = cv2.matchShapes(cnt, binary_im, 1, 0)  # reference_countour diganti dengan binary_im
    dist_list.append(retval)

# %%
sorted_list= dist_list.copy()
sorted_list.sort() # sorts the list from smallest to largest
ind1_dist= dist_list.index(sorted_list[0])
ind2_dist= dist_list.index(sorted_list[1])

# %%
banana_cnts= []
banana_cnts.append(contours[ind1_dist])
banana_cnts.append(contours[ind2_dist])

# %%
with_contours = cv2.drawContours(image,banana_cnts,-1,(255,0,0),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
for cnt in banana_cnts:
    x,y,w,h = cv2.boundingRect(cnt)
    if h>w:
        cv2.rectangle(imagecopy,(x,y),(x+w,y+h),(255,0,0),2)
cv2.imshow( 'Upright banana marked on RGB image' , imagecopy )
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:

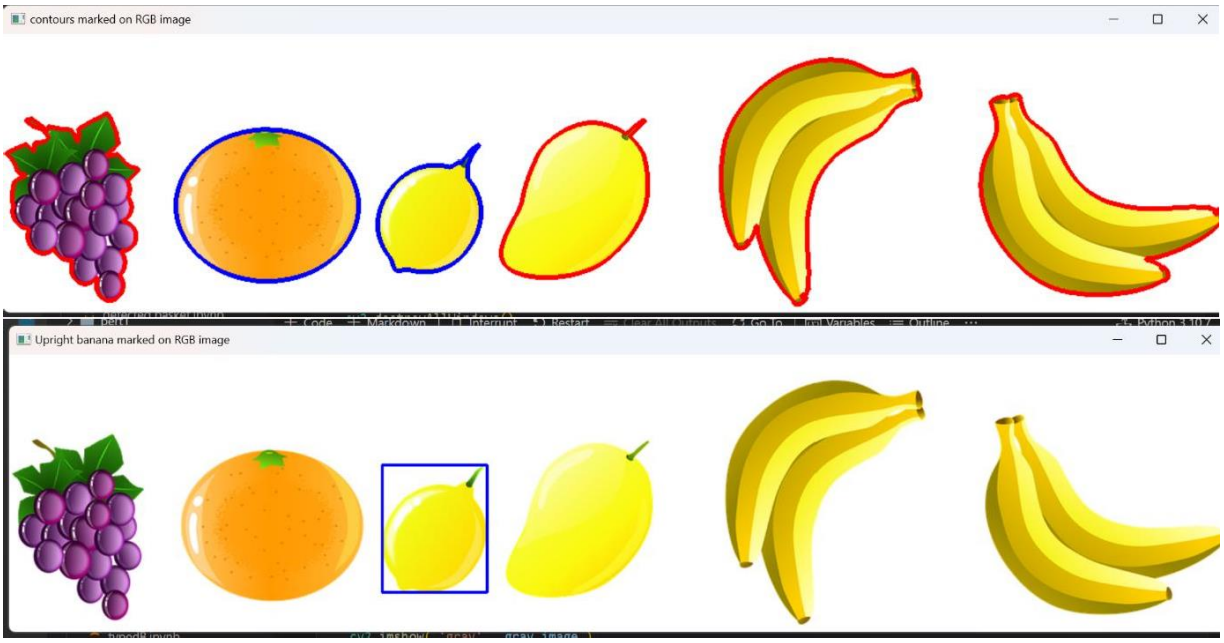

binary
gray
Original image
inverted binary
contours marked on RGB image


Reference image


Grayscale image


Binary image


inverted binary image



Typed_b.ipynb

```python
# %%
import cv2

# %%
image = cv2.imread('data/typed_B.png')
imagecopy= image.copy()
cv2.imshow( 'Original image' , image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
cv2.imshow( 'gray' , gray_image )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
ret,binary_im = cv2.threshold(gray_image,245,255,cv2.THRESH_BINARY)
cv2.imshow( 'binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
binary_im= ~binary_im
cv2.imshow( 'inverted binary' , binary_im )
cv2.waitKey(0)
cv2.destroyAllWindows()

# %%
#find the external contours from binary image
contours,hierarchy = cv2.findContours(binary_im,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

# %%
with_contours = cv2.drawContours(image,contours,-1,(0,0,255),3)
cv2.imshow( 'contours marked on RGB image' , with_contours )
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output: