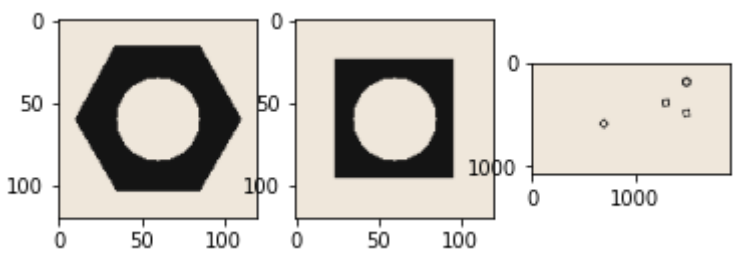
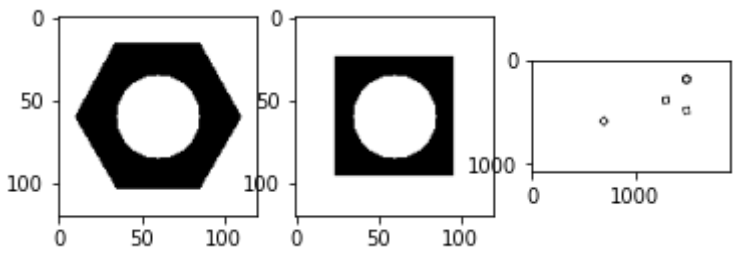
```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

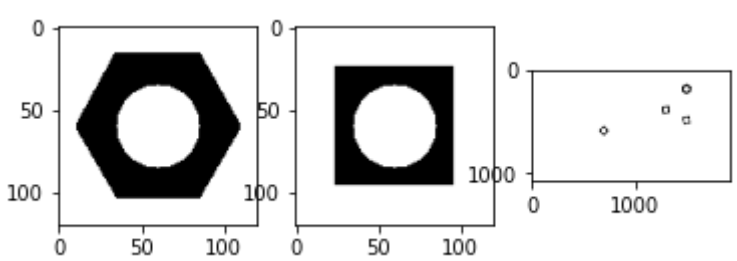
```
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt

hexnut_template = cv.imread('/content/gdrive/My Drive/ColabNotebooks/assignment3/hexnut_template.png', cv.IMREAD_COLOR)
squarenut_template =  cv.imread('/content/gdrive/My Drive/ColabNotebooks/assignment3/squarenut_template.png', cv.IMREAD_COLOR)
conveyor_f100 =  cv.imread('/content/gdrive/My Drive/ColabNotebooks/assignment3/conveyor_f100.png', cv.IMREAD_COLOR)

fig, ax = plt. subplots(1,3)
ax[0].imshow(cv.cvtColor(hexnut_template, cv.COLOR_RGB2BGR))
ax[1].imshow(cv.cvtColor(squarenut_template, cv.COLOR_RGB2BGR))
ax[2].imshow(cv.cvtColor(conveyor_f100, cv.COLOR_RGB2BGR))
plt.show()
```




```
hexnut_gray = cv.cvtColor(hexnut_template,cv.COLOR_BGR2GRAY)
squarenut_gray = cv.cvtColor(squarenut_template,cv.COLOR_BGR2GRAY)
conveyor_gray = cv.cvtColor(conveyor_f100,cv.COLOR_BGR2GRAY)

#blur1 = cv.GaussianBlur(hexnut_gray,(5,5),0)
#blur2= cv.GaussianBlur(squarenut_gray,(5,5),0)
#blur3 = cv.GaussianBlur(conveyor_gray,(5,5),0)

ret1,th1 = cv.threshold(hexnut_gray,0,255,cv.THRESH_BINARY+cv.THRESH_OTSU)
ret2,th2 = cv.threshold(squarenut_gray,0,255,cv.THRESH_BINARY+cv.THRESH_OTSU)
ret3,th3 = cv.threshold(conveyor_gray,0,255,cv.THRESH_BINARY+cv.THRESH_OTSU)

fig, ax = plt. subplots(1,3)
ax[0].imshow(th1,'gray')
ax[1].imshow(th2,'gray')
ax[2].imshow(th3,'gray')
plt.show()
```

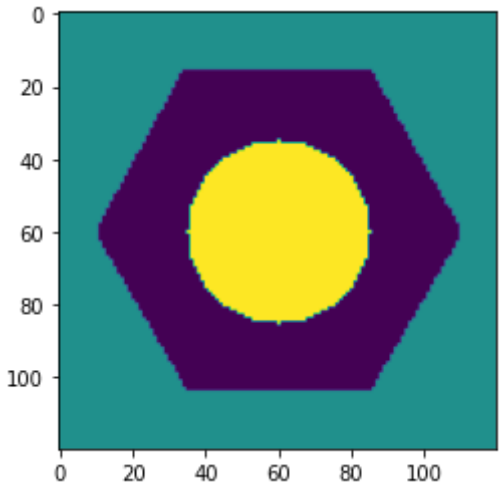



```
kernel = np.ones((3,3),np.uint8)
closing1 = cv.morphologyEx(th1, cv.MORPH_CLOSE, kernel)
closing2 = cv.morphologyEx(th2, cv.MORPH_CLOSE, kernel)
closing3 = cv.morphologyEx(th3, cv.MORPH_CLOSE, kernel)

fig, ax = plt. subplots(1,3)
ax[0].imshow(closing1,'gray')
ax[1].imshow(closing2,'gray')
ax[2].imshow(closing3,'gray')
plt.show()
```



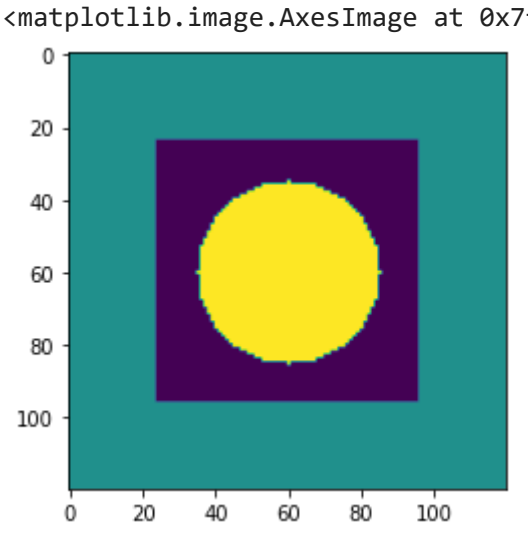
```
output1 = cv.connectedComponentsWithStats(closing1)
print('There are {} connected components in hexnut template.'.format(output1[0]))
print('Statistics:')
print(output1[2])
print()
print('Centroids:')
print(output1[3])
plt.imshow(output1[1])
```

```
There are 3 connected components in hexnut template.
Statistics:
[[  11   16   99   88 4722]
 [   0    0  120  120 7717]
 [  35   35   51   51 1961]]

Centroids:
[[59.83354511 59.22257518]
 [59.168848   59.54269794]
 [60.         60.        ]]
<matplotlib.image.AxesImage at 0x7f72efb99b50>
```



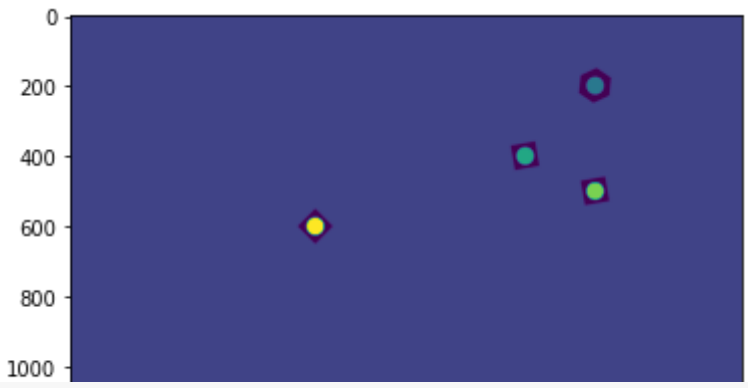

```
output2 = cv.connectedComponentsWithStats(closing2)
print('There are {} connected components in hexnut template.'.format(output2[0]))
print('Statistics:')
print(output2[2])
print()
print('Centroids:')
print(output2[3])
plt.imshow(output2[1])
```

```
There are 3 connected components in hexnut template.
Statistics:
[[  24   24   72   72 3223]
 [   0    0  120  120 9216]
 [  35   35   51   51 1961]]

Centroids:
[[59.19578033 59.19578033]
 [59.5        59.5       ]
 [60.         60.        ]]
<matplotlib.image.AxesImage at 0x7f72efac2790>
```



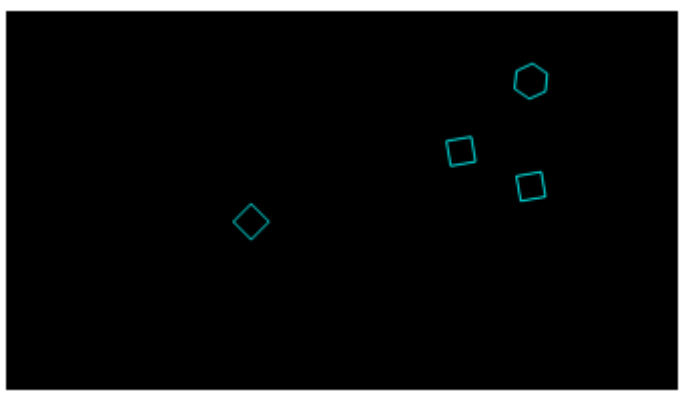

```
output3 = cv.connectedComponentsWithStats(closing3)
print('There are {} connected components in hexnut template.'.format(output3[0]))
print('Statistics:')
print(output3[2])
print()
print('Centroids:')
print(output3[3])
plt.imshow(output3[1])
```

```
There are 6 connected components in hexnut template.
Statistics:
[[    651     151     895     499   13930]
 [      0       0    1920    1080 2051826]
 [   1475     175      51      51    1961]
 [   1275     375      51      51    1961]
 [   1475     475      51      51    1961]
 [    675     575      51      51    1961]]

Centroids:
[[1275.02110553  400.11083991]
 [ 956.24734066  540.88404962]
 [1500.          200.        ]
 [1300.          400.        ]
 [1500.          500.        ]
 [ 700.          600.        ]]
<matplotlib.image.AxesImage at 0x7f72ef9d2d50>
```




```
back_ground=np.zeros(conveyor_f100.shape)

contours, hierarchy = cv.findContours(closing3, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
cnt = [contours[i] for i in range(1,9,2)]
cv.drawContours(back_ground, cnt, -1, (0,255,150), 3)
plt.imshow(back_ground)
plt.axis('off')
plt.show()
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```



✓ 0s completed at 15:51