Importing Modules

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import datetime
```

Reading image file in a seperate window

cv2.IMREAD_COLOR (1) , cv2.IMREAD_GRAYSCALE (0), CV2.IMREAD_UNCHANGED (-1)

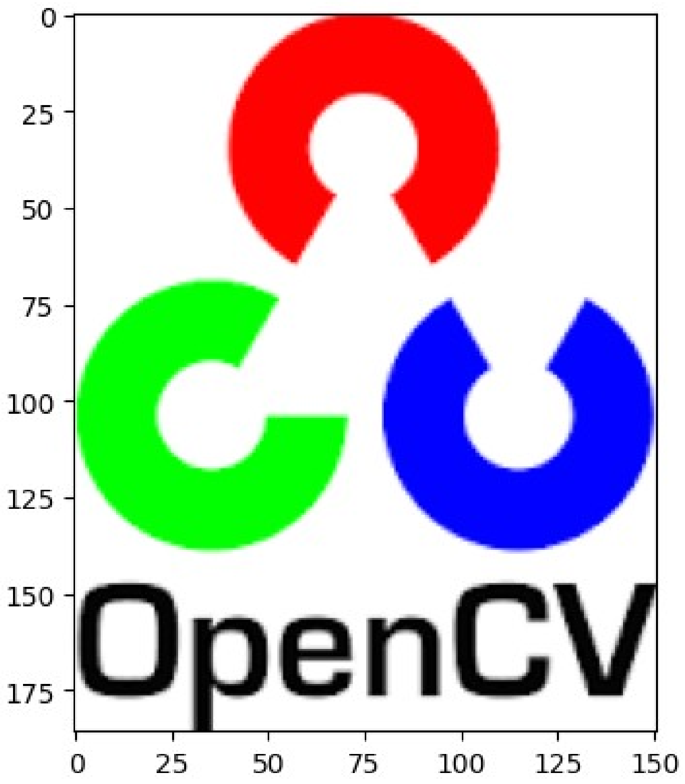
```
img = cv2.imread('opencv.png',1)
cv2.imshow('Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
-----------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
Cell In[1], line 1
----> 1 img = cv2.imread('opencv.png',1)
      2 cv2.imshow('Image',img)
      3 cv2.waitKey(0)

NameError: name 'cv2' is not defined
```

Image in Larger Window

```
img = cv2.imread('opencv.png',1)
cv2.namedWindow('Image',cv2.WINDOW_NORMAL)
cv2.resizeWindow('Image',500,500)
cv2.imshow('Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()

#plt.imshow(img)
plt.imshow(img[:,:,::-1])
```

```
<matplotlib.image.AxesImage at 0x164629c4cb0>
```



Writing the image to a different file

```
img = cv2.imread('opencv.png',cv2.IMREAD_COLOR)
cv2.imwrite('opencv.bmp',img)

img = cv2.imread('opencv.png',cv2.IMREAD_GRAYSCALE)
cv2.imwrite('opencv_gray.png',img)
```

Difference between Black and White image and opening image in Grayscale

```
plt.figure(figsize=[10, 5])

img1=cv2.imread('opencv.bmp',1)
plt.subplot(121);plt.imshow(img1)

img2=cv2.imread('opencv_gray.png',1)
plt.subplot(122);plt.imshow(img2,cmap='gray')

#img2=cv2.imread('opencv_gray.png',0)
#plt.subplot(122);plt.imshow(img2,cmap='gray')

print(img.shape,img1.shape,img2.shape)
```

```
(186, 151) (186, 151, 3) (186, 151, 3)
```

```
blue = img2[155,5]
print(blue)

np.set_printoptions(threshold = sys.maxsize)
print(img)

print(img.shape)
a = img[0]
print(a.shape)
print(a)

print(img[0][0].shape)
print(img[0,0])
img[0,0].shape
```

```
(3,)
[255 255 255]
```

```
(3,)
```

```
for i in range(10):
    for j in range(10):
        print(img[i,j])
```

Changing colour of few pixels in the image

```
for i in range(10):
    for j in range(10):
          img[i,j]=[200,200,60]

plt.imshow(img)
```

Drawing a Line on Image

```
img = cv2.imread('opencv.png',1)

opencvline = img.copy()

# The line starts from (10,25) and ends at (100,25)
# The color of the line is YELLOW (Recall that OpenCV uses BGR format)
# Thickness of line is 5px
# Linetype is cv2.LINE_AA

cv2.line(opencvline, (10,25), (100,25), (0, 255, 255), thickness=1,
lineType=cv2.LINE_AA)

#Arrowed Line
#cv2.arrowedLine(opencvline, (100,25), (10,25), (0, 255, 255),
thickness=1);

# Display the image
```

```
plt.imshow(opencvline[:,:,::-1])
#plt.imshow(opencvline)
```

Drawing Circle, eclipse, rectangel

cv2.circle(image, center_coordinates, radius, color, thickness)

```
opencvcircle=opencvline.copy()
cv2.circle(opencvcircle,(105,50),40,(32,165,218),1,cv2.LINE_8)
plt.imshow(opencvcircle[:,:,::-1])

opencvcirclefill=opencvline.copy()
cv2.circle(opencvcirclefill,(125,25),15,(32,165,218),-2)
plt.imshow(opencvcirclefill[:,:,::-1])
```

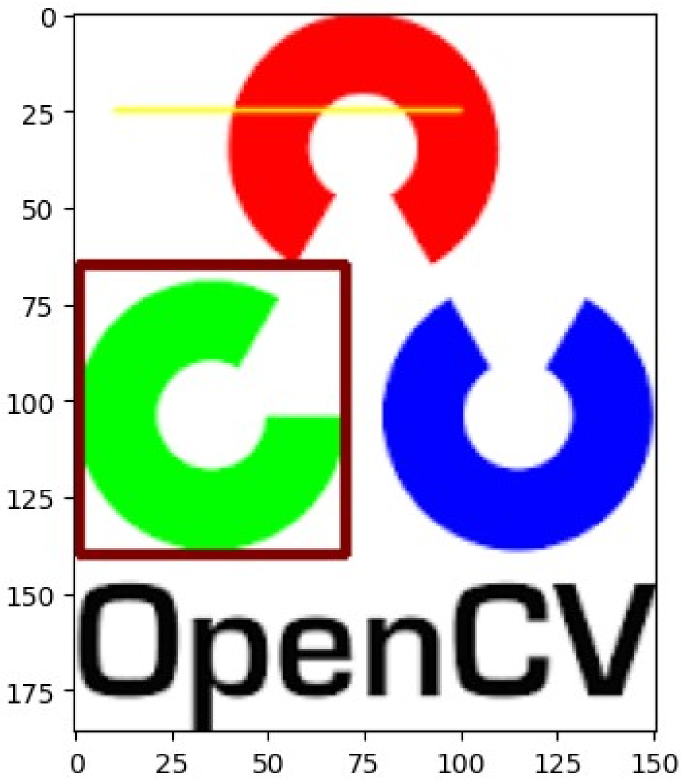
cv2.ellipse(image, center_coordinates, axesLength, angle, startAngle, endAngle, color, thickness)

```
opencvoval=opencvline.copy()
cv2.ellipse(opencvoval,(75,75),(20,10),0,0,360,(255,0,255),-1)
plt.imshow(opencvoval[:,:,::-1])
```

cv2.rectangle(image, start_point, end_point, color, thickness)

```
opencvrect=opencvline.copy()
cv2.rectangle(opencvrect,(1,65),(70,140),(0,0,128),2)
plt.imshow(opencvrect[:,:,::-1])
```

```
<matplotlib.image.AxesImage at 0x194a6ffb590>
```



cv2.putText(image, 'OpenCV', org, font, fontScale, color, thickness,lineType)

```
opencvtext=opencvline.copy()
#cv2.putText(opencvtext,'MVMC',(30,40),cv2.FONT_HERSHEY_PLAIN,2,
(255,0,0),1,cv2.LINE_AA)
d = str(datetime.datetime.now())
cv2.putText(opencvtext,d,(10,40),cv2.FONT_HERSHEY_PLAIN,1,
(255,0,0),1,cv2.LINE_AA)
plt.imshow(opencvtext[:,:,::-1])
```

Image Cropping

```
img = cv2.imread('opencv.png',1)
cv2.imshow('Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()

crop = img[1:60,30:120]
cv2.imshow('Origianal Image',img)
cv2.moveWindow('Origianal Image',10,10)
cv2.imshow('Cropped Image',crop)
cv2.moveWindow('Cropped Image',10,300)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Resizing Image - Fix number of pixels / n times the original image

```
img = cv2.imread('opencv.png',1)
cv2.imshow('Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
img.shape
```

```
(186, 151, 3)
```

```
img = cv2.imread('opencv.png',1)
img=cv2.resize(img,(500,500))
cv2.imshow('Resized Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
img = cv2.imread('opencv.png',1)
img=cv2.resize(img,(0,0),fx=0.5,fy=0.5)
cv2.imshow('Resized Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
img.shape
```

```
(93, 76, 3)
```

```
img = cv2.imread('opencv.png',1)
img=cv2.resize(img,(0,0),fx=2,fy=2)
cv2.imshow('Resized Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
img.shape
```

```
(372, 302, 3)
```

```
crop=cv2.resize(crop,(0,0),fx=3,fy=3)
cv2.imshow('Resized Image',crop)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Image Rotation

```
img = cv2.imread('opencv.png',1)
img=cv2.rotate(img, cv2.ROTATE_180)
cv2.imshow('Resized Image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Image Concatnation

```
cropred = img[0:60,30:120]
cropgreen = img[60:120,0:90]
cv2.imshow('Red Image',cropred)
cv2.moveWindow('Red Image',10,10)
cv2.imshow('Green Image',cropgreen)
cv2.moveWindow('Green Image',10,300)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
print(cropred.shape,cropgreen.shape)
```

```
(60, 90, 3) (60, 90, 3)
```

```
h_con = cv2.hconcat([cropred, cropgreen])
v_con = cv2.vconcat([cropred, cropgreen])
cv2.imshow('Horizontal',h_con)
cv2.moveWindow('Horizontal',10,10)
cv2.imshow('Verticle',v_con)
cv2.moveWindow('Verticle',10,300)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
cropred = img[0:60,30:120]
cropgreen = img[65:140,0:80]
cv2.imshow('Red Image',cropred)
cv2.moveWindow('Red Image',10,10)
cv2.imshow('Green Image',cropgreen)
cv2.moveWindow('Green Image',10,300)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
print(cropred.shape,cropgreen.shape)
```

```
(60, 90, 3) (75, 80, 3)
```

```
cropred = cv2.resize(cropred,(80,75))
print(cropred.shape,cropgreen.shape)
```

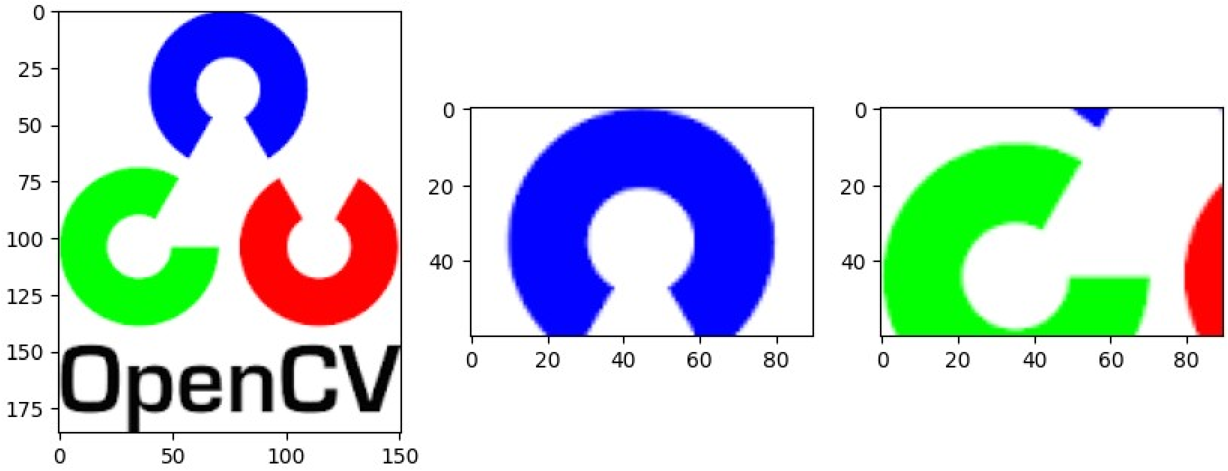
```
(75, 80, 3) (75, 80, 3)
```

```
h_con = cv2.hconcat([cropred, cropgreen])
v_con = cv2.vconcat([cropred, cropgreen])
cv2.imshow('Horizontal',h_con)
cv2.moveWindow('Horizontal',10,10)
cv2.imshow('Verticle',v_con)
cv2.moveWindow('Verticle',10,300)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Concatnating images using numpy - Towards creating Collage

```
plt.figure(figsize=[10, 5])
img=cv2.imread('opencv.png',1)
plt.subplot(131);plt.imshow(img)
plt.subplot(132);plt.imshow(cropred)
plt.subplot(133);plt.imshow(cropgreen)
```

```
<matplotlib.image.AxesImage at 0x2520172b200>
```



```
print(img.shape,cropred.shape,cropgreen.shape)
```

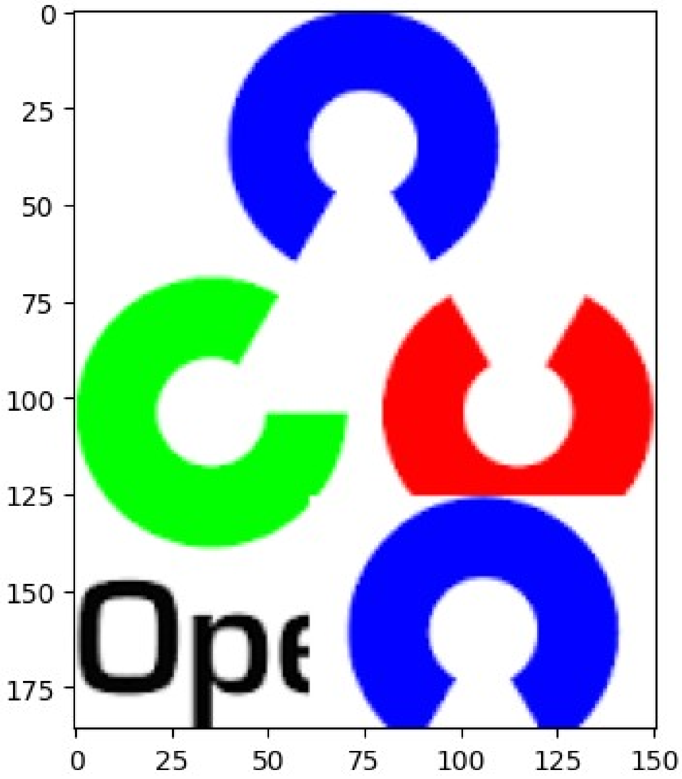
```
(186, 151, 3) (60, 90, 3) (60, 90, 3)
```

```
print(img[126:,61:].shape)
```

```
(60, 90, 3)
```

```
img[126:,61:]=cropred
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x25201729700>
```



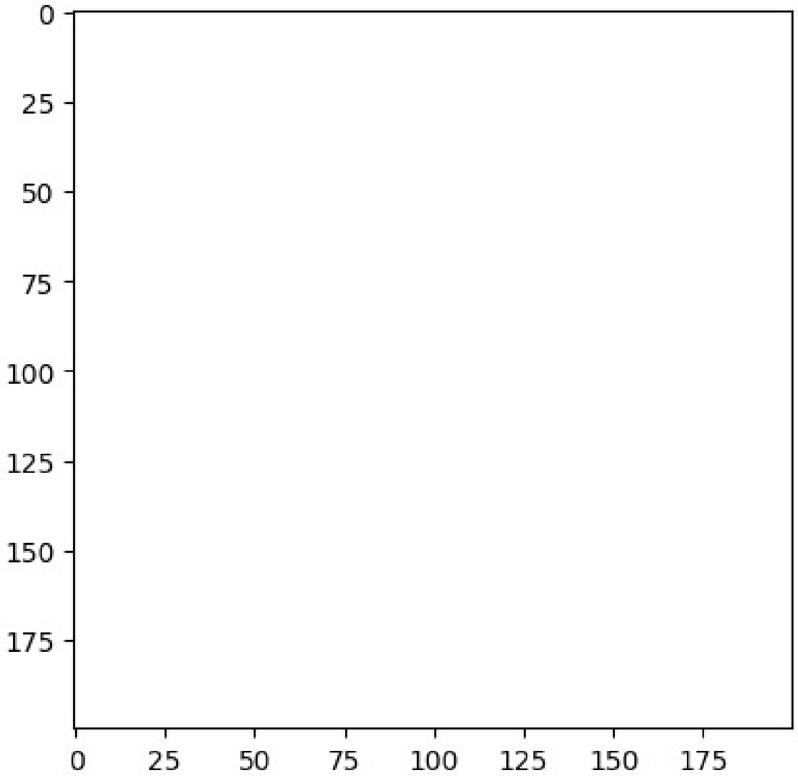
Creating an image from numpy array

Creating a numpy array of desired shape / dimension

```
ran_array = np.random.randint(255,256,(200,200,3))
#print(ran_array)
print(ran_array.ndim, ran_array.shape)

3 (200, 200, 3)
```

Plotting and saving the numpy array as an image

```
plt.imshow(ran_array)
cv2.imwrite('ran_array.png',ran_array)

True
```



Modifying and saving the image for study of bitwise operator

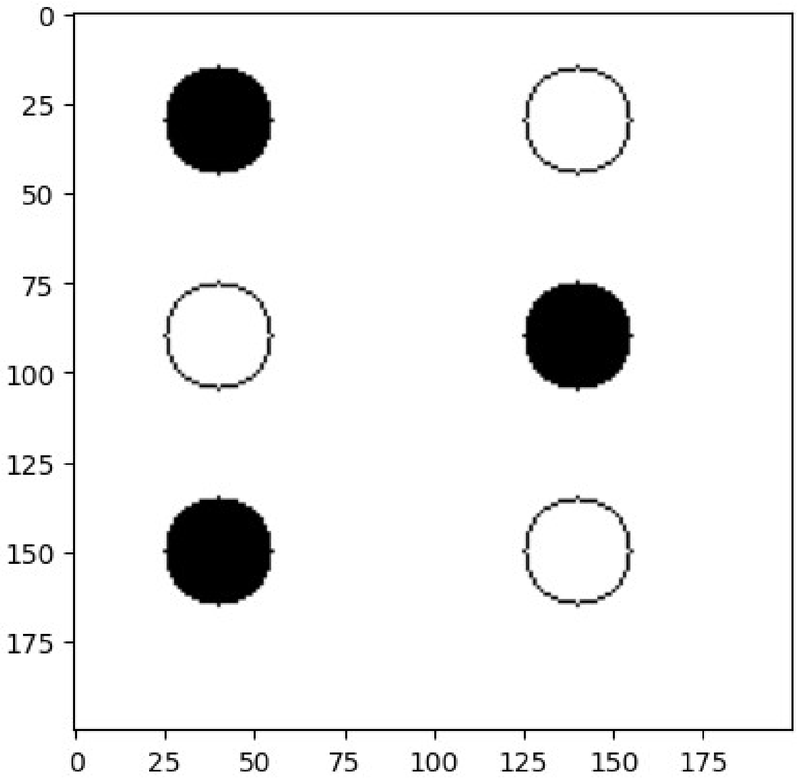
```
img_ran_b = cv2.imread('ran_array.png',1)
cv2.circle(img_ran_b,(40,30),15,(0,0,0),1)
cv2.circle(img_ran_b,(140,30),15,(0,0,0),-1)
cv2.circle(img_ran_b,(40,90),15,(0,0,0),-1)
cv2.circle(img_ran_b,(140,90),15,(0,0,0),1)
cv2.circle(img_ran_b,(40,150),15,(0,0,0),1)
cv2.circle(img_ran_b,(140,150),15,(0,0,0),-1)
plt.imshow(img_ran_b)
```

```
cv2.imwrite('bitwise_1.png',img_ran_b)
```

```
True
```

```
img_ran_b = cv2.imread('ran_array.png',1)
cv2.circle(img_ran_b,(40,30),15,(0,0,0),-1)
cv2.circle(img_ran_b,(140,30),15,(0,0,0),1)
cv2.circle(img_ran_b,(40,90),15,(0,0,0),1)
cv2.circle(img_ran_b,(140,90),15,(0,0,0),-1)
cv2.circle(img_ran_b,(40,150),15,(0,0,0),-1)
cv2.circle(img_ran_b,(140,150),15,(0,0,0),1)
plt.imshow(img_ran_b)
```

```
<matplotlib.image.AxesImage at 0x252018eb4a0>
```

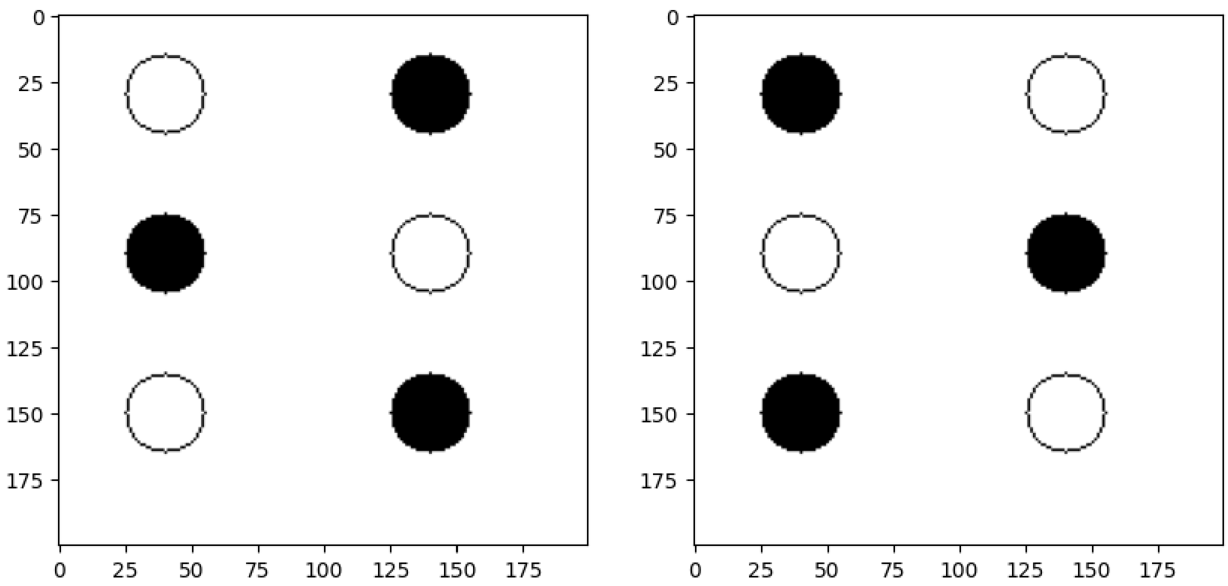


```
cv2.imwrite('bitwise_2.png',img_ran_b)
```

```
True
```

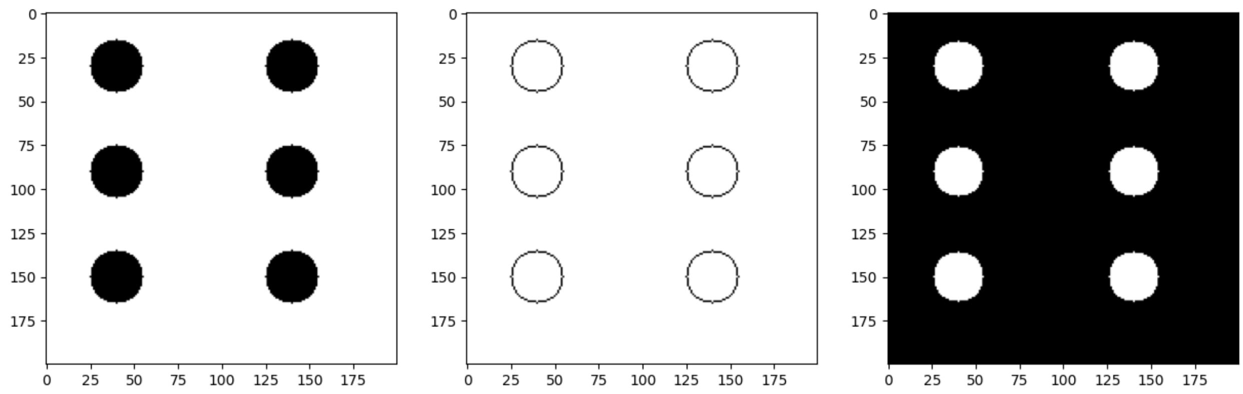
```
plt.figure(figsize=[10, 5])
bit1 = cv2.imread('bitwise_1.png',1)
bit2 = cv2.imread('bitwise_2.png',1)
plt.subplot(121);plt.imshow(bit1)
plt.subplot(122);plt.imshow(bit2)
```

```
<matplotlib.image.AxesImage at 0x287b5387c80>
```

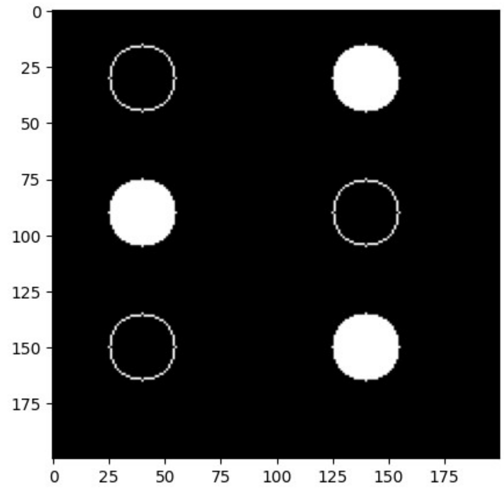


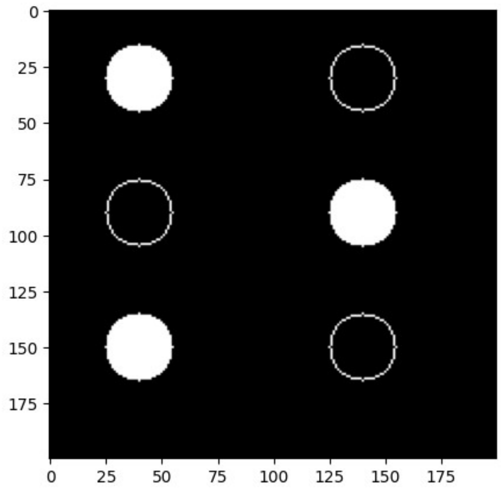
Bitwise AND

```
plt.figure(figsize=[15, 5])
bit_and = cv2.bitwise_and(bit1,bit2,mask=None)
bit_or = cv2.bitwise_or(bit1,bit2,mask=None)
bit_xor = cv2.bitwise_xor(bit1,bit2,mask=None)
plt.subplot(131);plt.imshow(bit_and)
plt.subplot(132);plt.imshow(bit_or)
plt.subplot(133);plt.imshow(bit_xor)

<matplotlib.image.AxesImage at 0x287b52147a0>
```



```
plt.figure(figsize=[15, 5])
bit1_not = cv2.bitwise_not(bit1,mask=None)
bit2_not = cv2.bitwise_not(bit2,mask=None)
plt.subplot(121);plt.imshow(bit1_not)
plt.subplot(122);plt.imshow(bit2_not)

<matplotlib.image.AxesImage at 0x287b1ccea50>
```



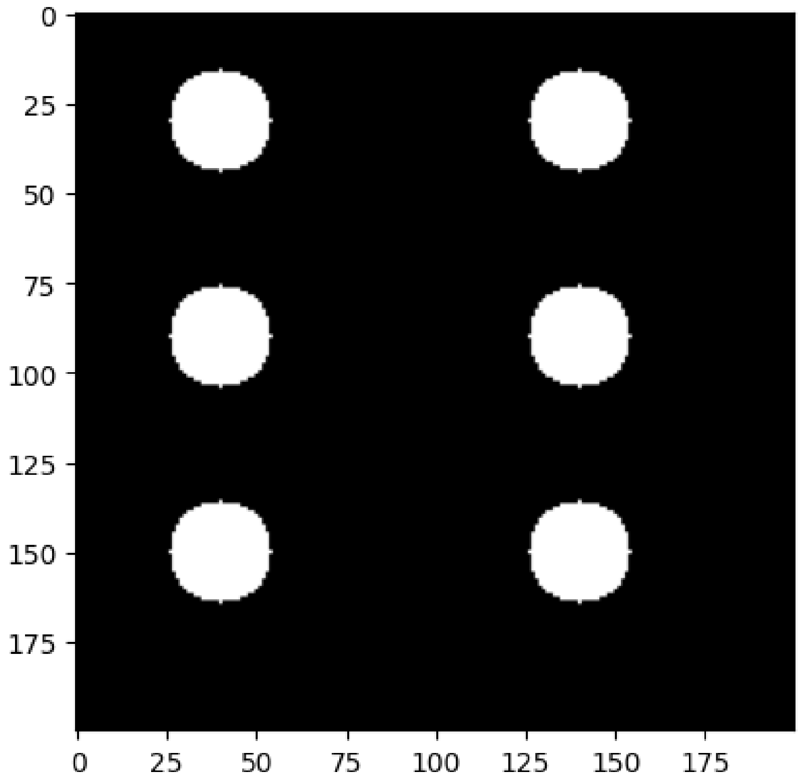


Creating image MASK

```
mask= cv2.cvtColor(bit_xor,cv2.COLOR_BGR2GRAY)
print(mask.shape)
plt.imshow(mask, cmap='gray')
```

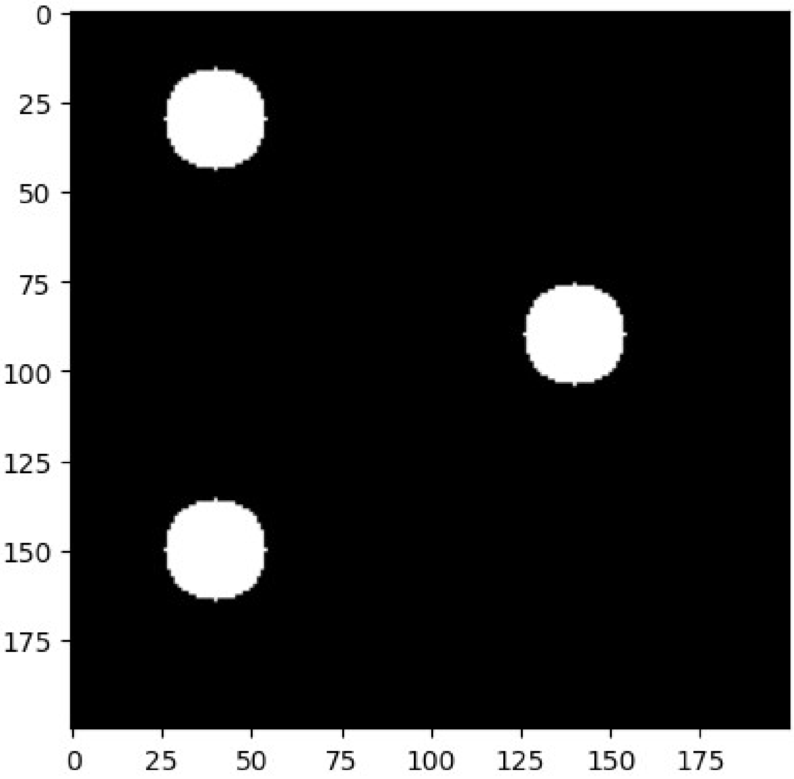
```
(200, 200)
```

```
<matplotlib.image.AxesImage at 0x287b4ee0f20>
```



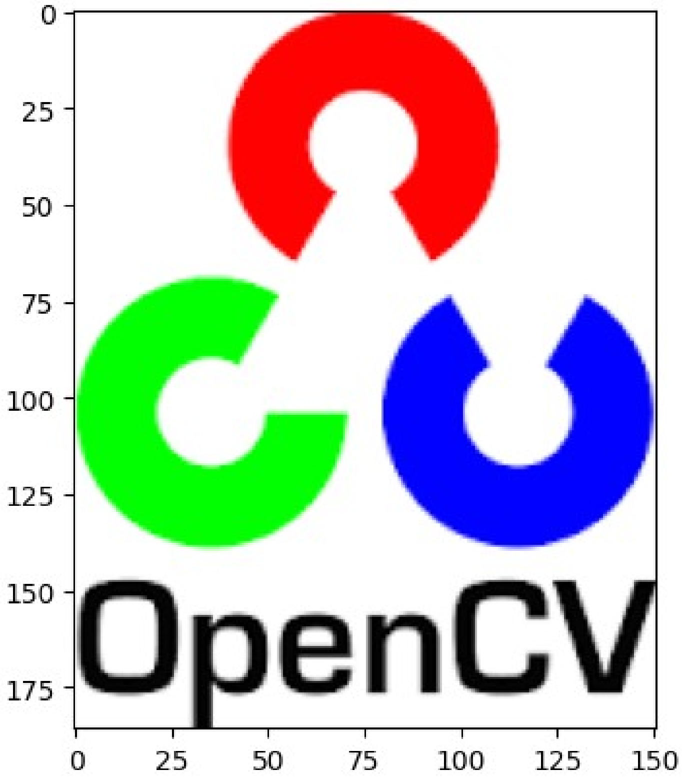
```
bit_and_mask=cv2.bitwise_and(bit1,bit1,mask=mask)
plt.imshow(bit_and_mask)
```

```
<matplotlib.image.AxesImage at 0x287b6ff5fd0>
```



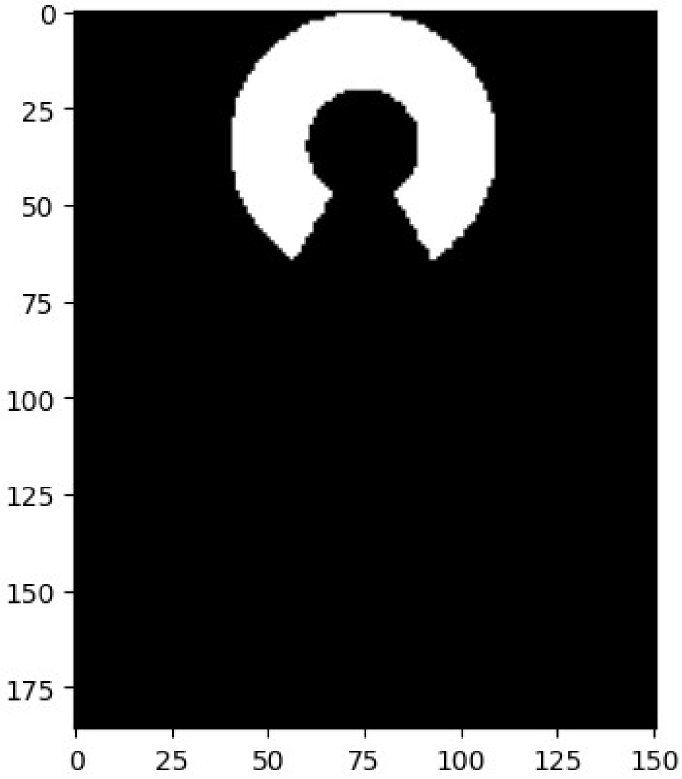
```
img=cv2.imread('opencv.png',1)
plt.imshow(img[:,:,::-1])
```

```
<matplotlib.image.AxesImage at 0x16465987d10>
```



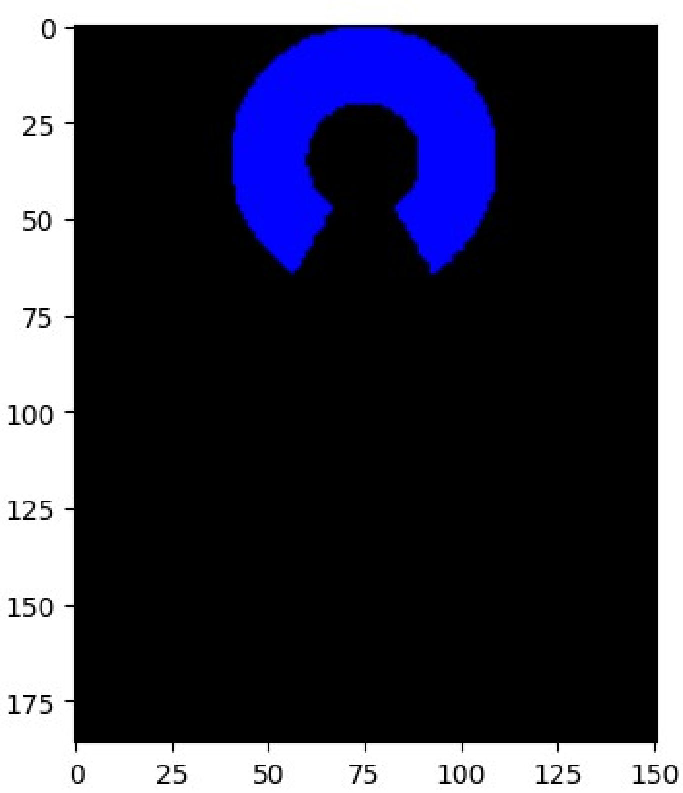
```
lower_red = np.array([0,0,250])
upper_red = np.array([1,1,255])
opencv_mask = cv2.inRange(img, lower_red,upper_red)
plt.imshow(opencv_mask,cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x1646637a270>
```



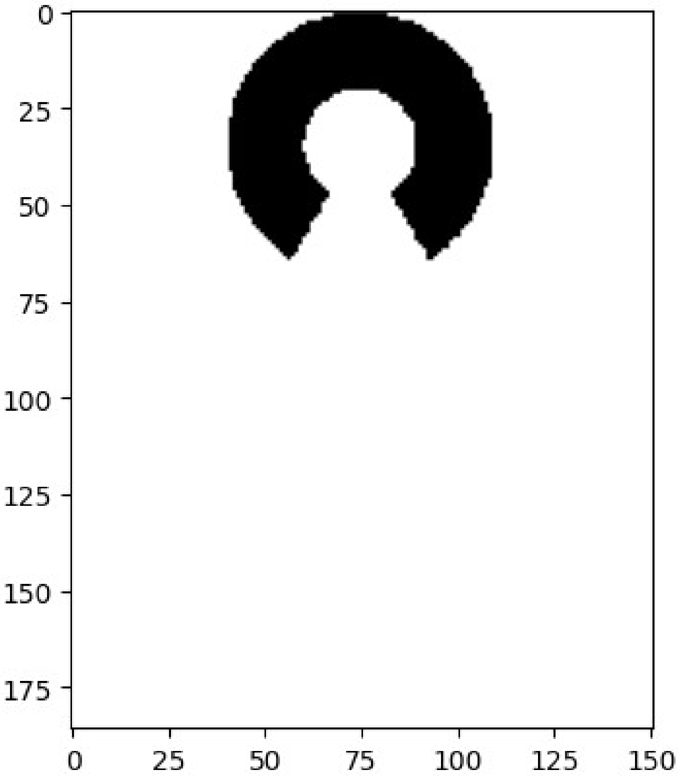
```
img_bgr = cv2.imread('opencv.png',1)
plt.imshow(img_bgr)

newimage = cv2.bitwise_and(img_bgr,img_bgr,mask=opencv_mask)
plt.imshow(newimage)
```

```
<matplotlib.image.AxesImage at 0x16467e7f0b0>
```



Creating an Inverse Mask

```
opencv_mask_inv = cv2.bitwise_not(opencv_mask)
plt.imshow(opencv_mask_inv,cmap='gray')

<matplotlib.image.AxesImage at 0x16466a1ab40>
```



```
newimage1 = cv2.bitwise_and(img_bgr,img_bgr,mask=opencv_mask_inv)
plt.imshow(newimage1)
```