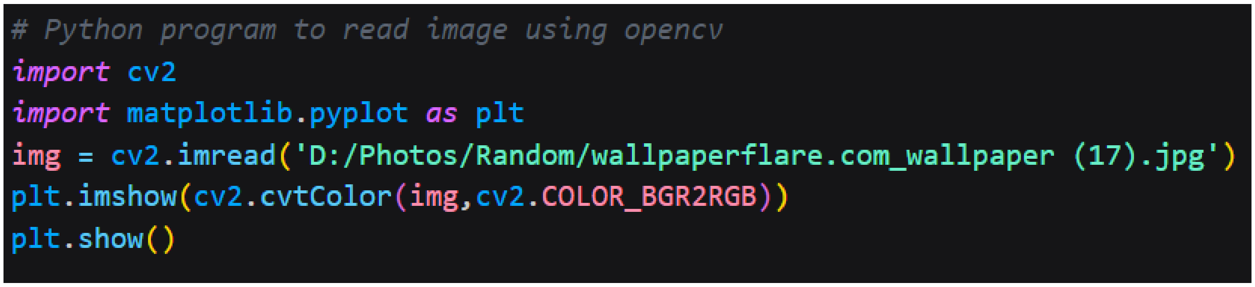
Digital Image Processing
ASSIGNMENT -2

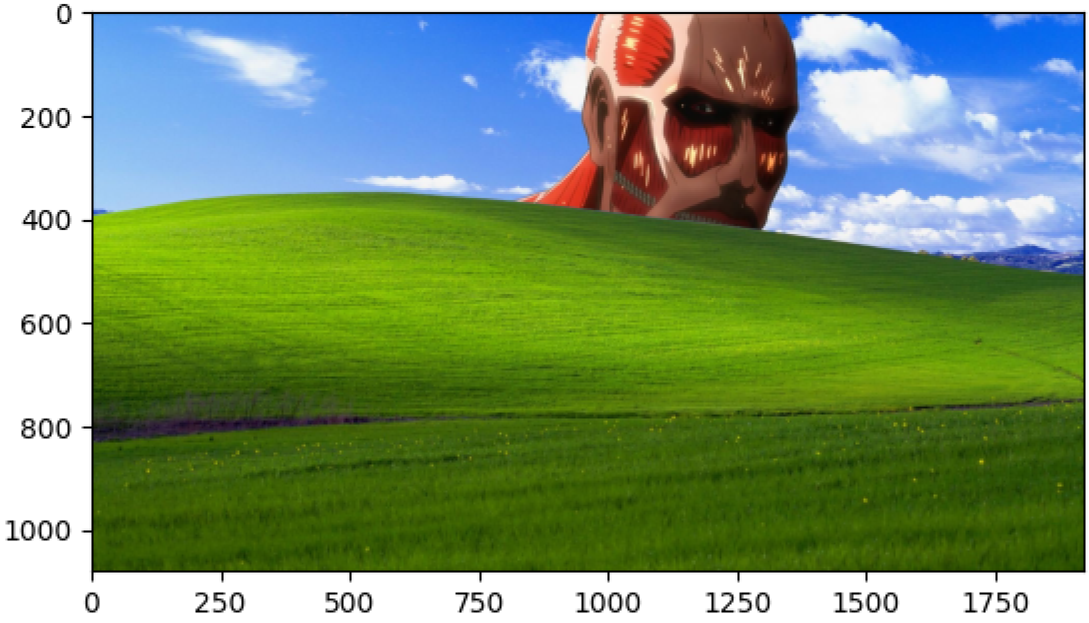
Perumalla Dharan
AP21110010201

1. Perform the following operations using library functions
    a. Python program to read image using opencv

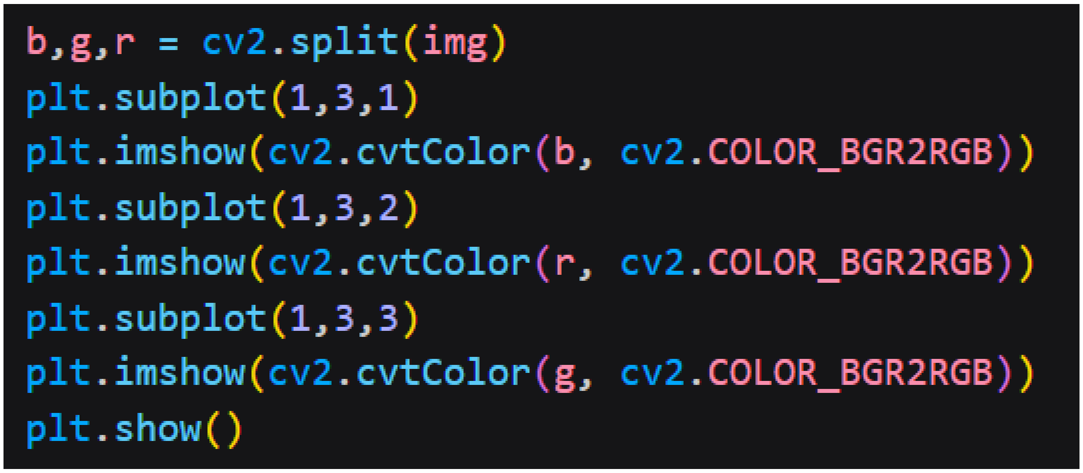


```
# Python program to read image using opencv
import cv2
import matplotlib.pyplot as plt
img = cv2.imread('D:/Photos/Random/wallpaperflare.com_wallpaper (17).jpg')
plt.imshow(cv2.cvtColor(img,cv2.COLOR_BGR2RGB))
plt.show()
```

<u>Output</u>

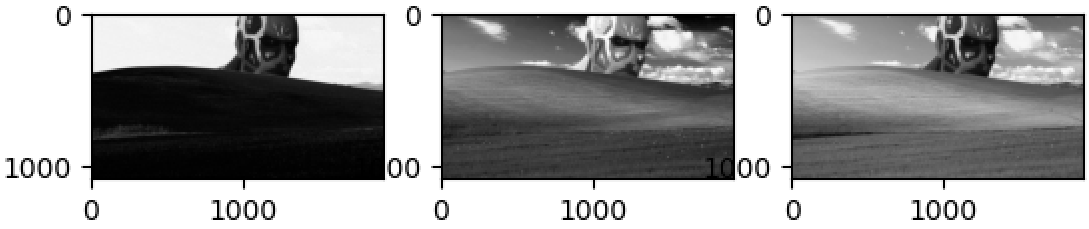


b. Find RED, GREEN and BLUE plane of the color image.



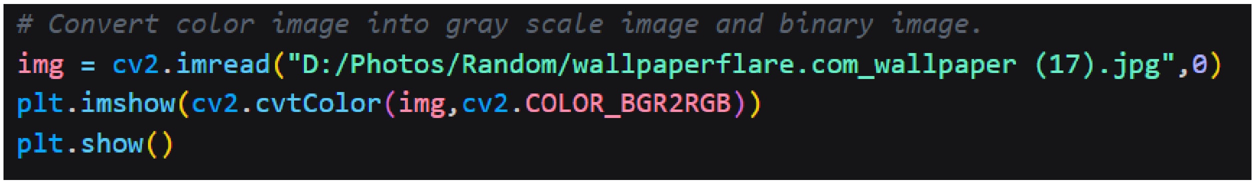
```
b,g,r = cv2.split(img)
plt.subplot(1,3,1)
plt.imshow(cv2.cvtColor(b, cv2.COLOR_BGR2RGB))
plt.subplot(1,3,2)
plt.imshow(cv2.cvtColor(r, cv2.COLOR_BGR2RGB))
plt.subplot(1,3,3)
plt.imshow(cv2.cvtColor(g, cv2.COLOR_BGR2RGB))
plt.show()
```

<u>Output</u>



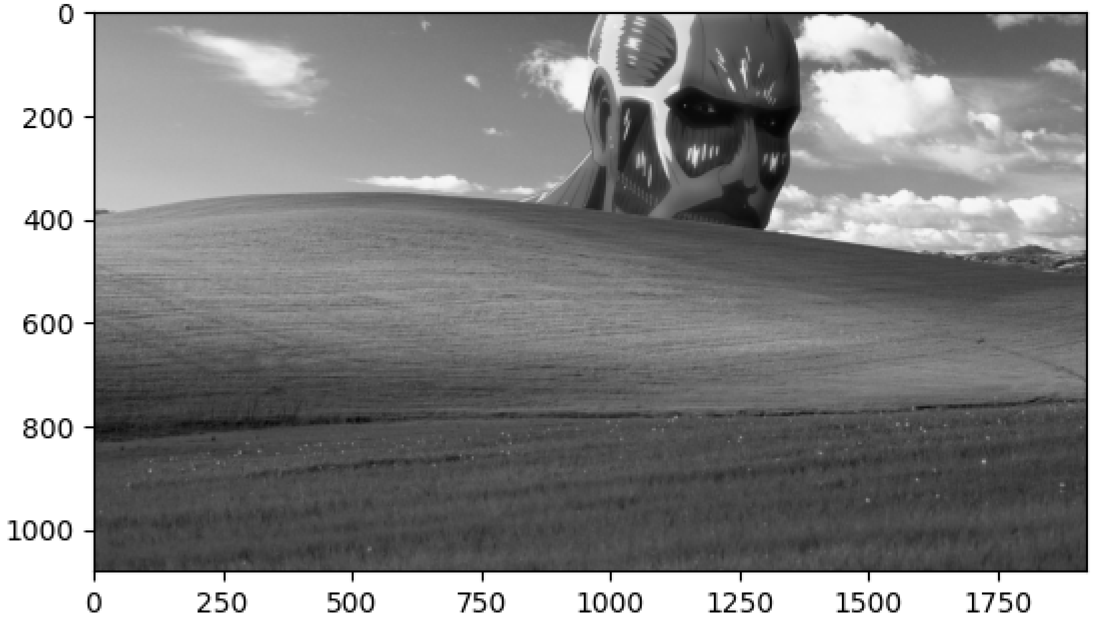
---

c. Convert color image into gray scale image and binary image.



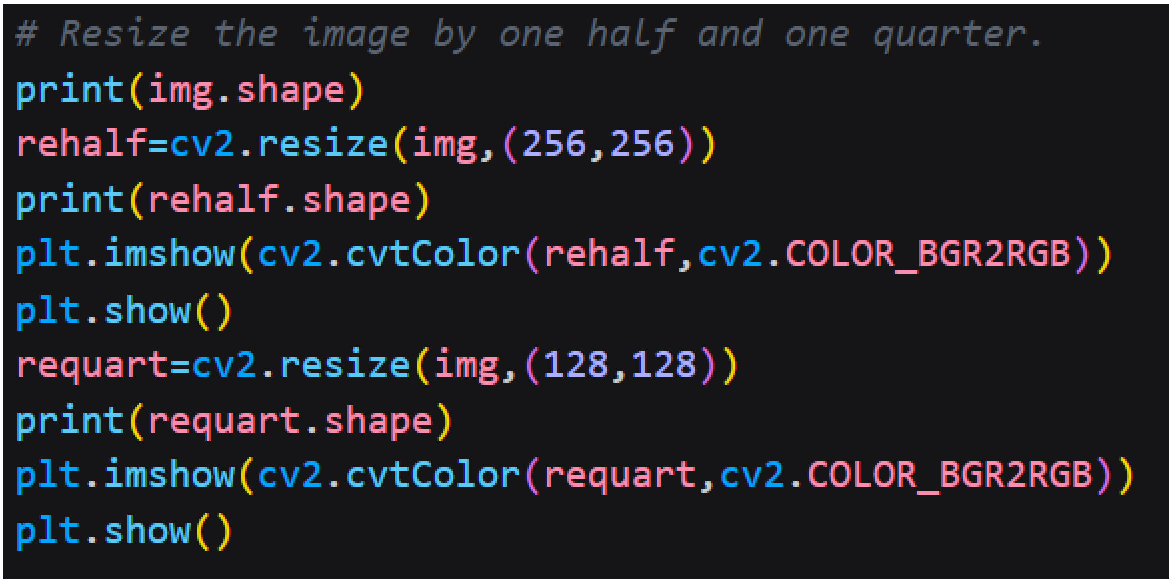
```
# Convert color image into gray scale image and binary image.
img = cv2.imread("D:/Photos/Random/wallpaperflare.com_wallpaper (17).jpg",0)
plt.imshow(cv2.cvtColor(img,cv2.COLOR_BGR2RGB))
plt.show()
```

<u>Output</u>



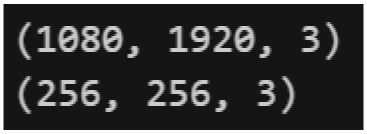
---

d. Resize the image by one half and one quarter.

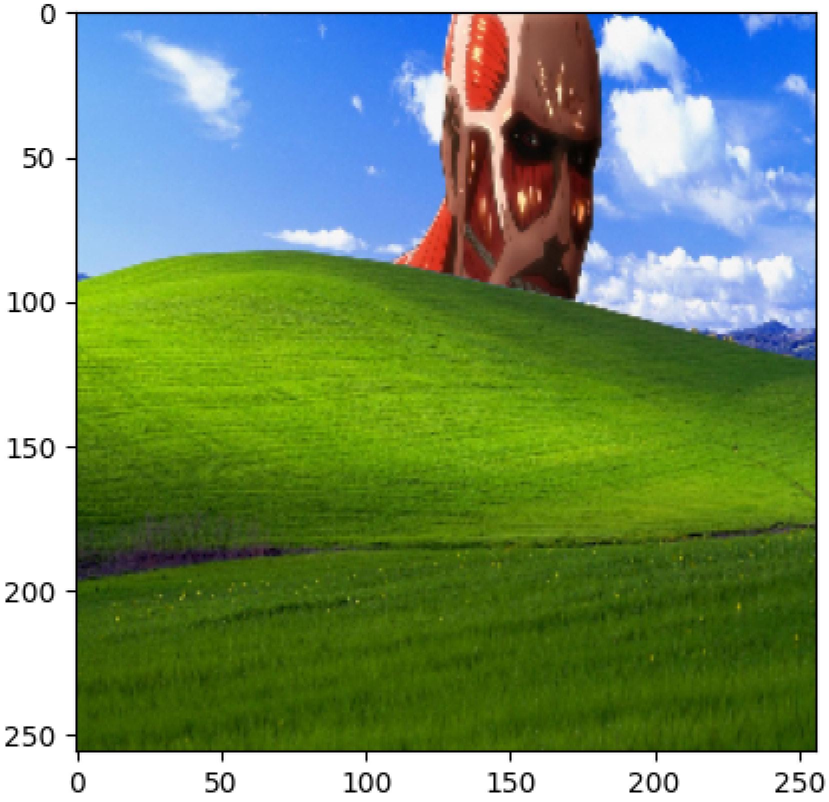


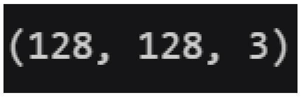
```
# Resize the image by one half and one quarter.
print(img.shape)
rehalf=cv2.resize(img,(256,256))
print(rehalf.shape)
plt.imshow(cv2.cvtColor(rehalf,cv2.COLOR_BGR2RGB))
plt.show()
requart=cv2.resize(img,(128,128))
print(requart.shape)
plt.imshow(cv2.cvtColor(requart,cv2.COLOR_BGR2RGB))
plt.show()
```

<u>Output</u>

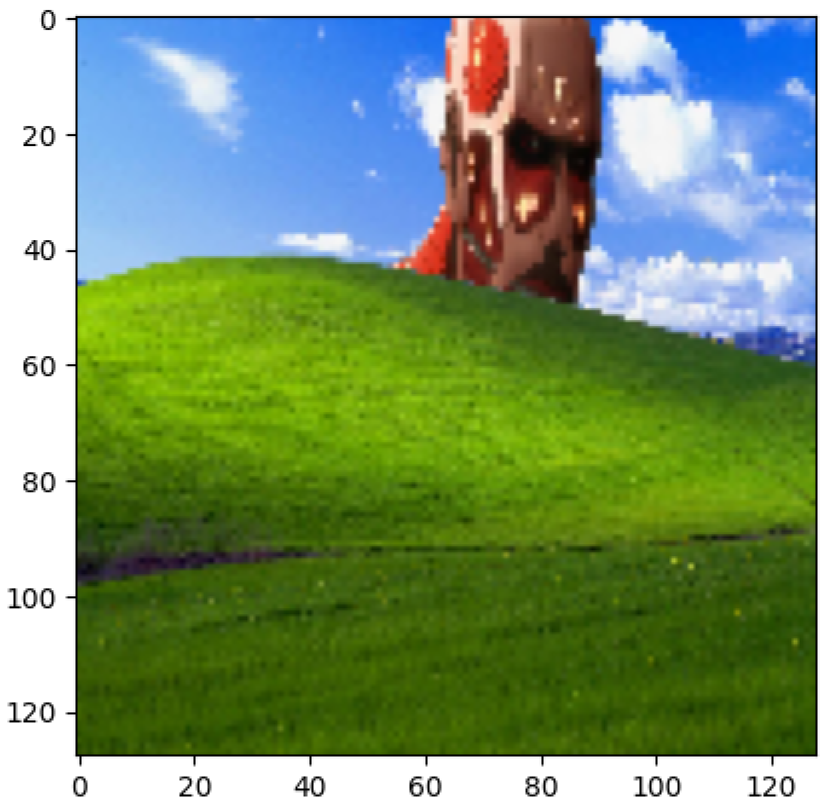


```
(1080, 1920, 3)
(256, 256, 3)
```



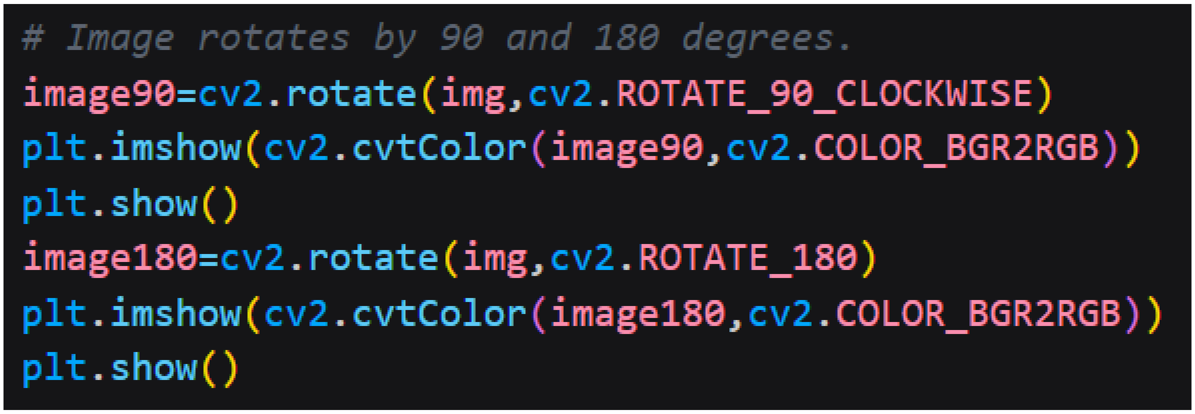


```
(128, 128, 3)
```



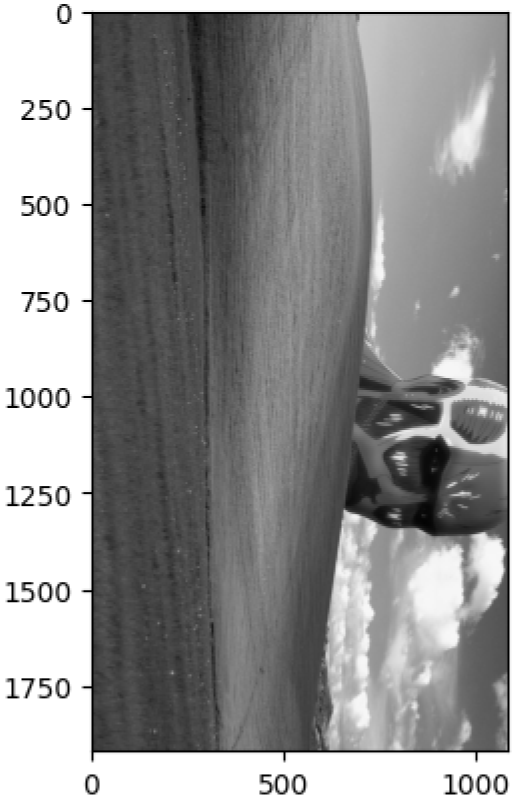
---

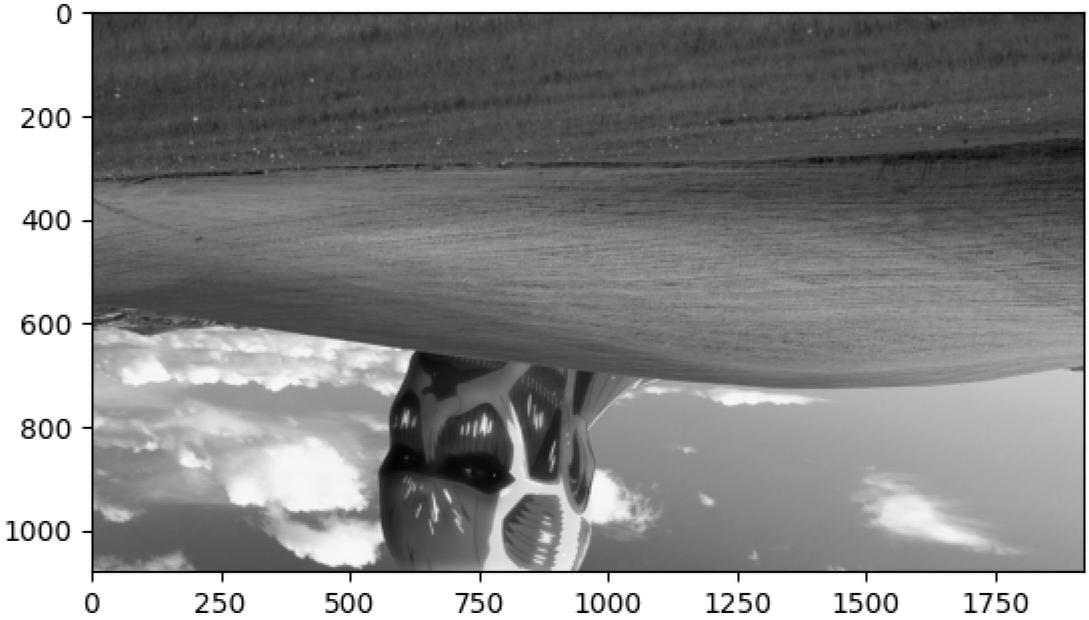
e. Image rotates by 45, 90 and 180 degrees.

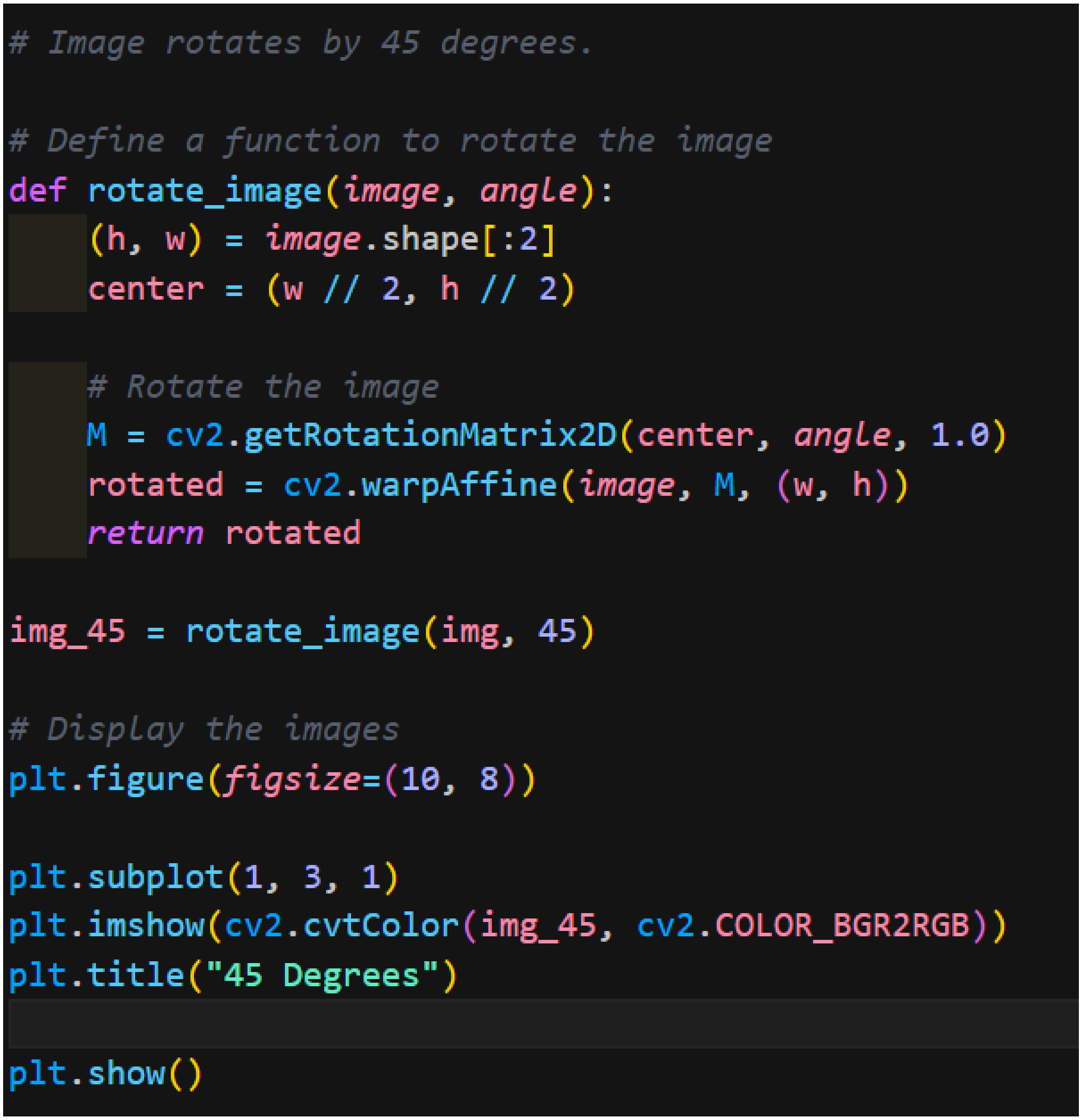


```
# Image rotates by 90 and 180 degrees.
image90=cv2.rotate(img,cv2.ROTATE_90_CLOCKWISE)
plt.imshow(cv2.cvtColor(image90,cv2.COLOR_BGR2RGB))
plt.show()
image180=cv2.rotate(img,cv2.ROTATE_180)
plt.imshow(cv2.cvtColor(image180,cv2.COLOR_BGR2RGB))
plt.show()
```

<u>Output</u>

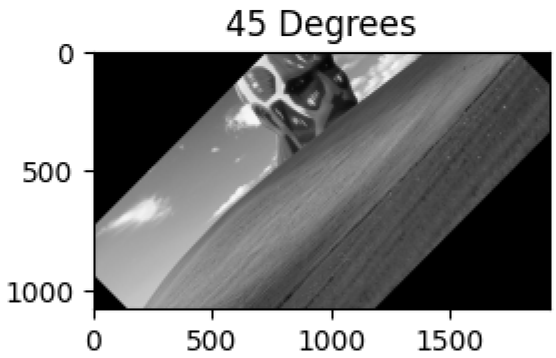

0
250
500
750
1000
1250
1500
1750
0
500
1000


0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750

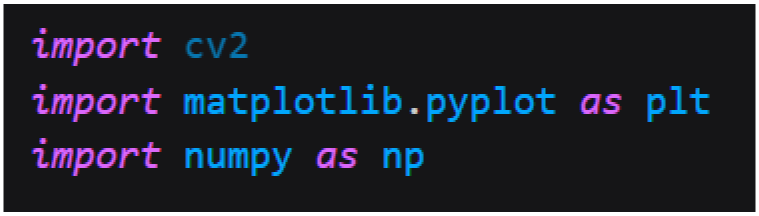


```
# Image rotates by 45 degrees.

# Define a function to rotate the image
def rotate_image(image, angle):
    (h, w) = image.shape[:2]
    center = (w // 2, h // 2)

    # Rotate the image
    M = cv2.getRotationMatrix2D(center, angle, 1.0)
    rotated = cv2.warpAffine(image, M, (w, h))
    return rotated

img_45 = rotate_image(img, 45)

# Display the images
plt.figure(figsize=(10, 8))

plt.subplot(1, 3, 1)
plt.imshow(cv2.cvtColor(img_45, cv2.COLOR_BGR2RGB))
plt.title("45 Degrees")

plt.show()
```

Output

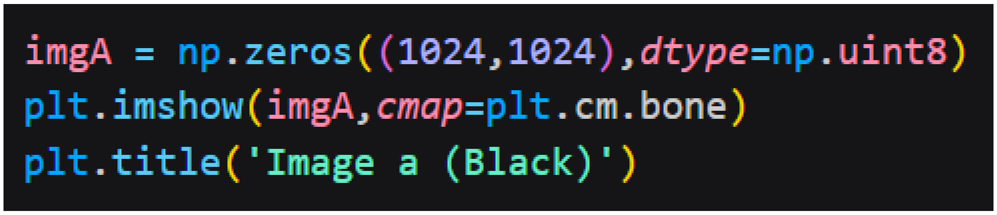


2. Create black and white images (A) of size 1024x1024. Which consists of alternative horizontal lines of black and white? Each line is of size 128. Create black and white images (B) of size 1024x1024. Which consists of alternative vertical lines of black and white? Each line is of size128. Perform the following operations on Image A and Image B.

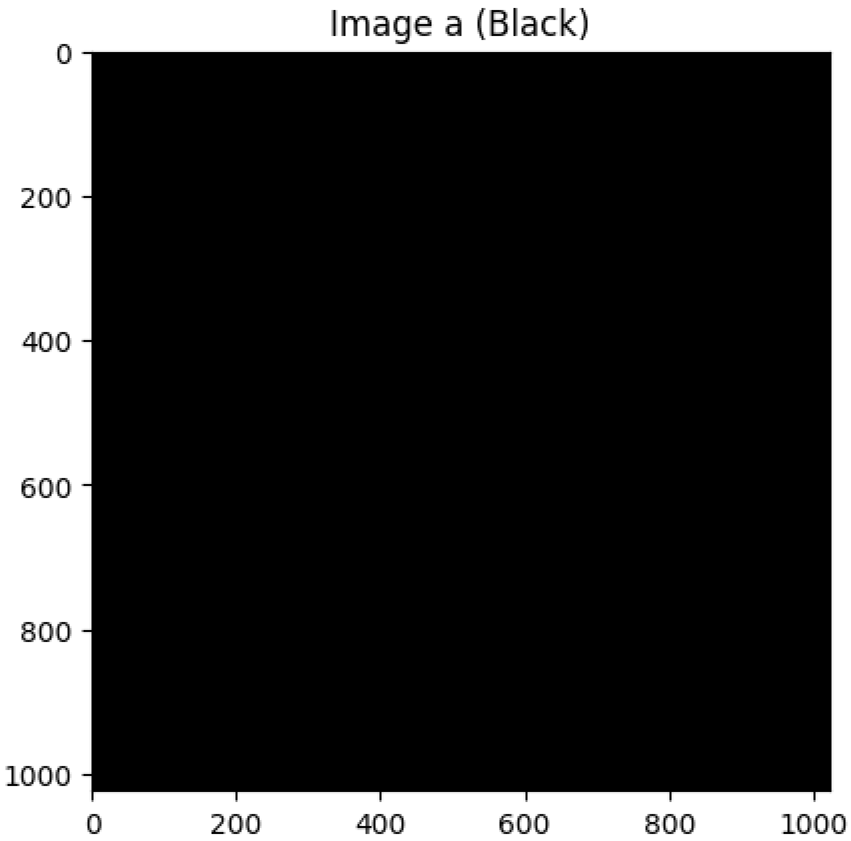


```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

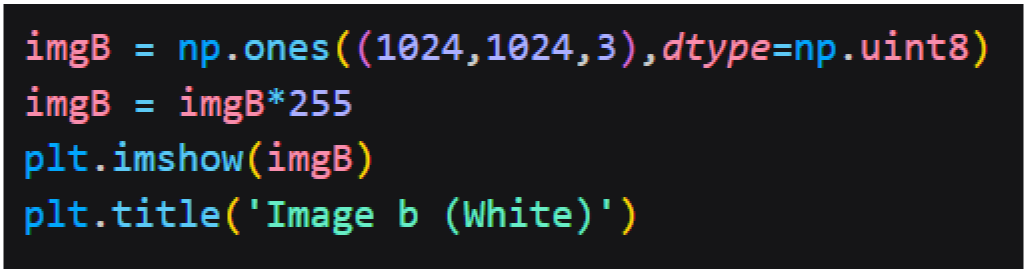
Code to create a black image

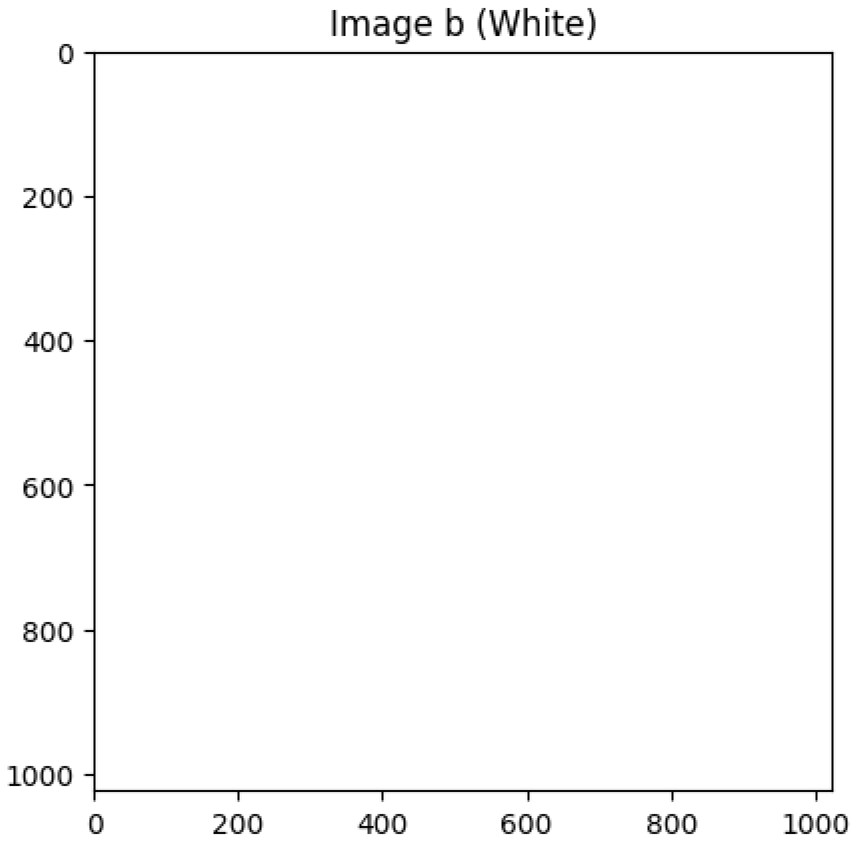


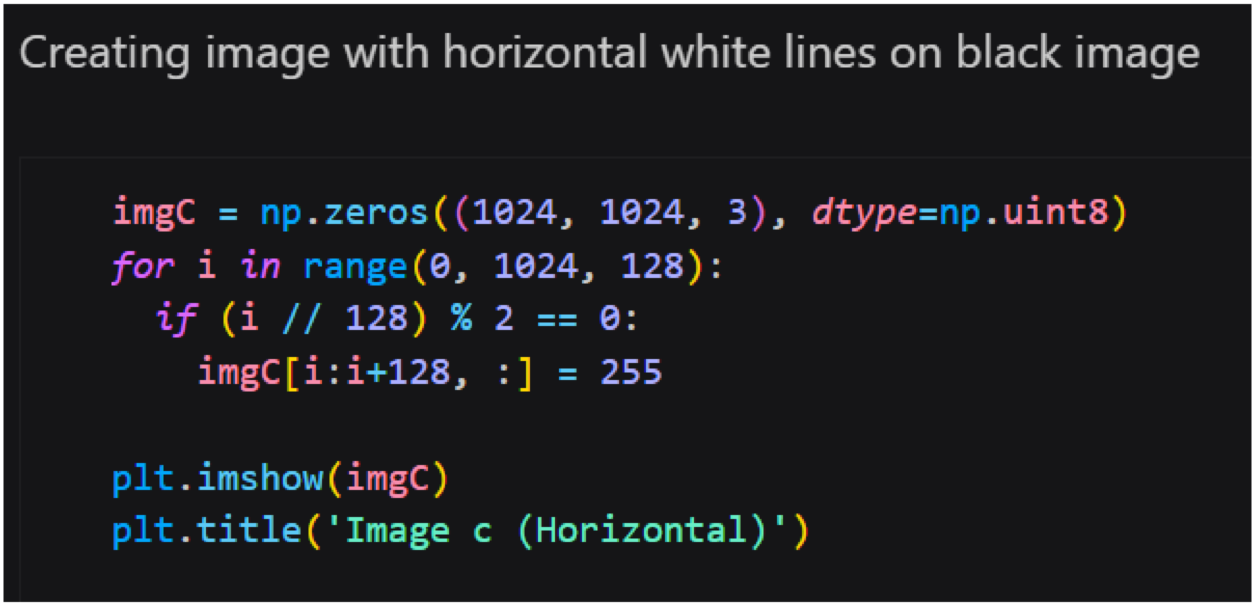
```
imgA = np.zeros((1024,1024),dtype=np.uint8)
plt.imshow(imgA,cmap=plt.cm.bone)
plt.title('Image a (Black)')
```



Code to create a white image


```
imgB = np.ones((1024,1024,3),dtype=np.uint8)
imgB = imgB*255
plt.imshow(imgB)
plt.title('Image b (White)')
```




Creating image with horizontal white lines on black image

```
imgC = np.zeros((1024, 1024, 3), dtype=np.uint8)
for i in range(0, 1024, 128):
  if (i // 128) % 2 == 0:
    imgC[i:i+128, :] = 255

plt.imshow(imgC)
plt.title('Image c (Horizontal)')
```

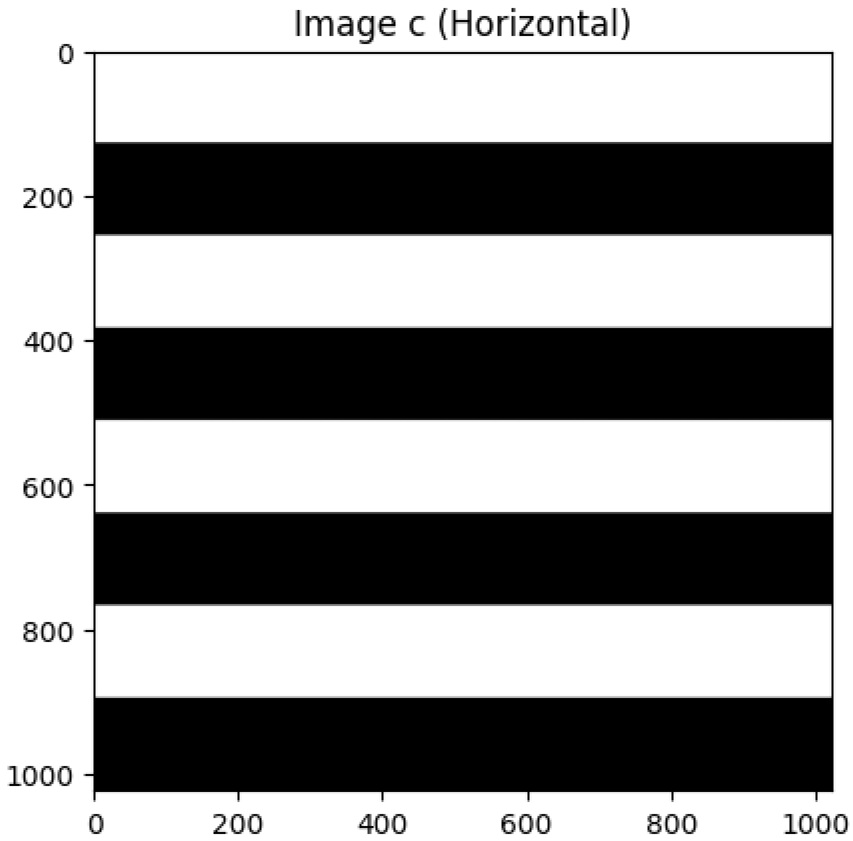
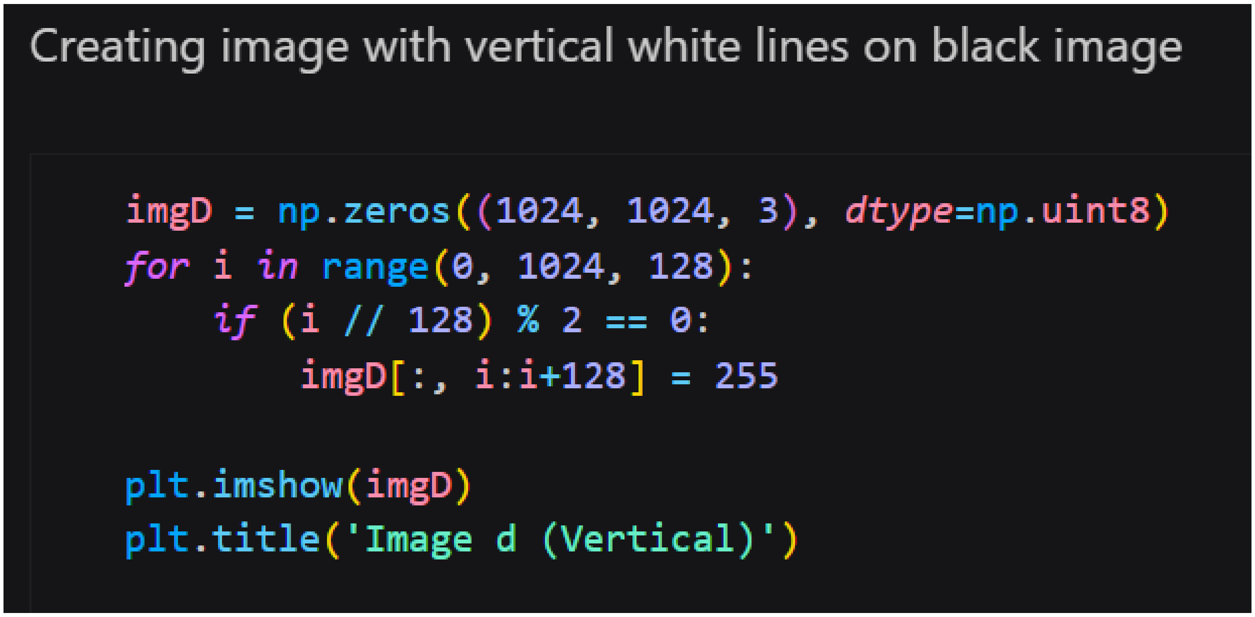


Image c (Horizontal)



Creating image with vertical white lines on black image

```
imgD = np.zeros((1024, 1024, 3), dtype=np.uint8)
for i in range(0, 1024, 128):
    if (i // 128) % 2 == 0:
        imgD[:, i:i+128] = 255

plt.imshow(imgD)
plt.title('Image d (Vertical)')
```

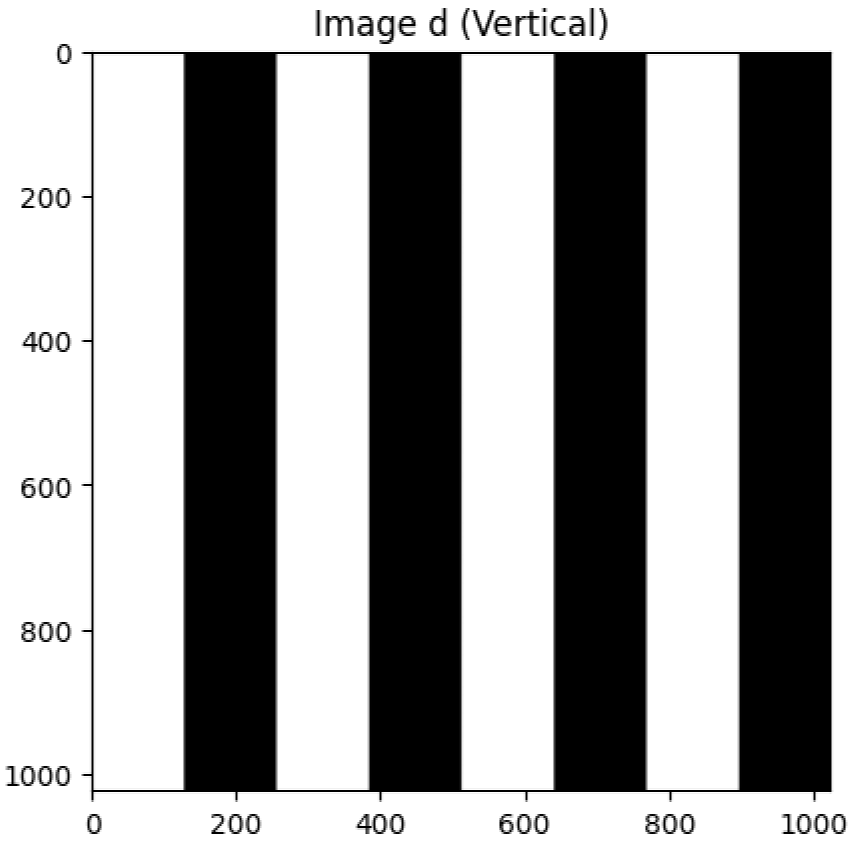
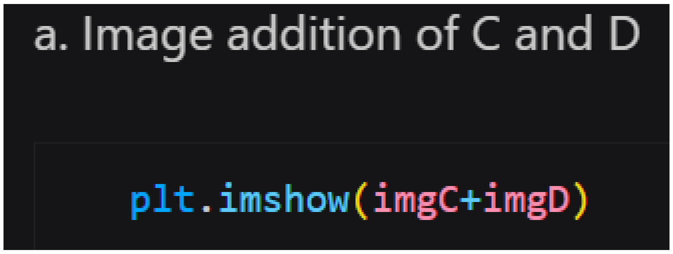


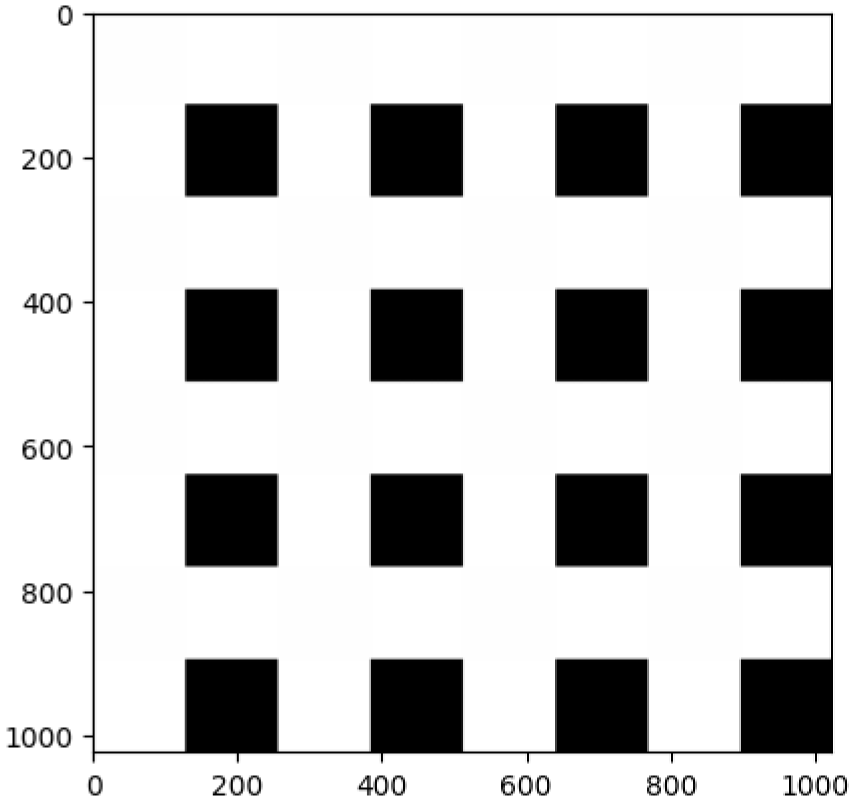
Image d (Vertical)

---

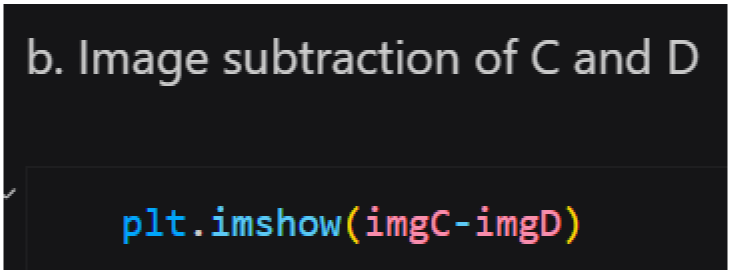
a.


a. Image addition of C and D

```
plt.imshow(imgC+imgD)
```

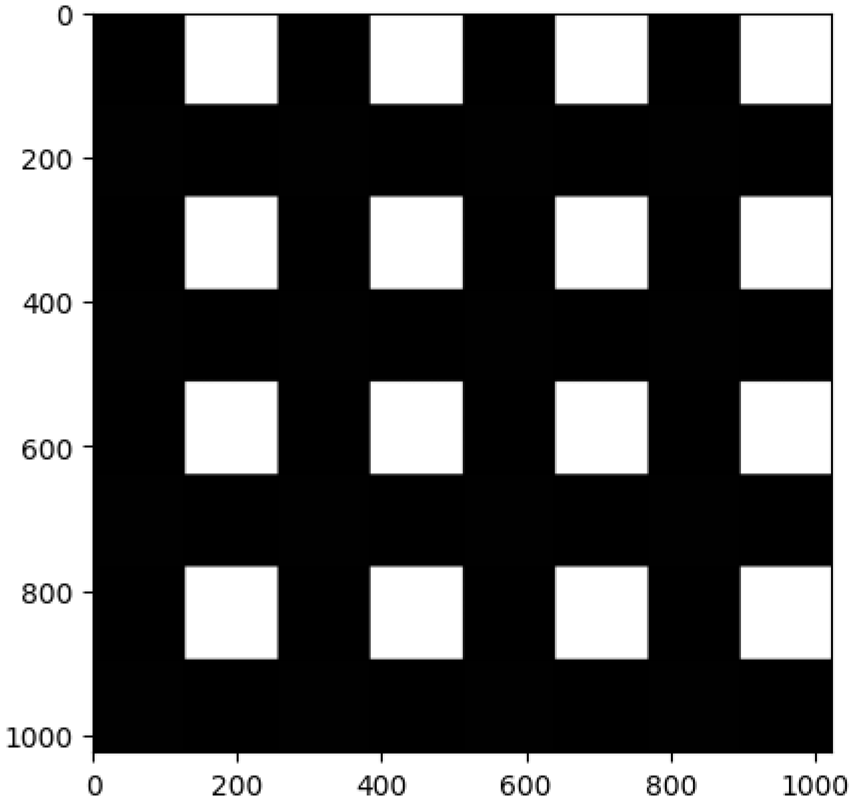


b.

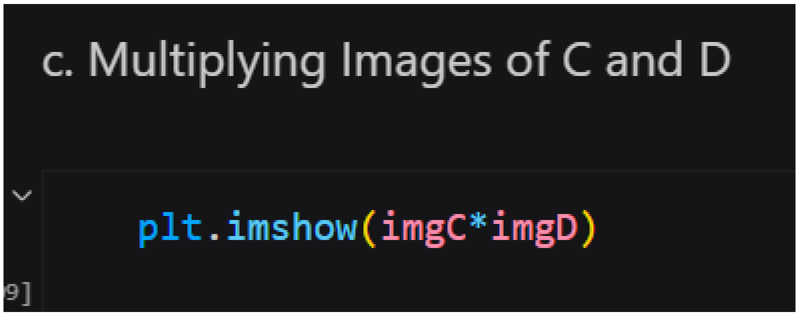


b. Image subtraction of C and D

```
plt.imshow(imgC-imgD)
```

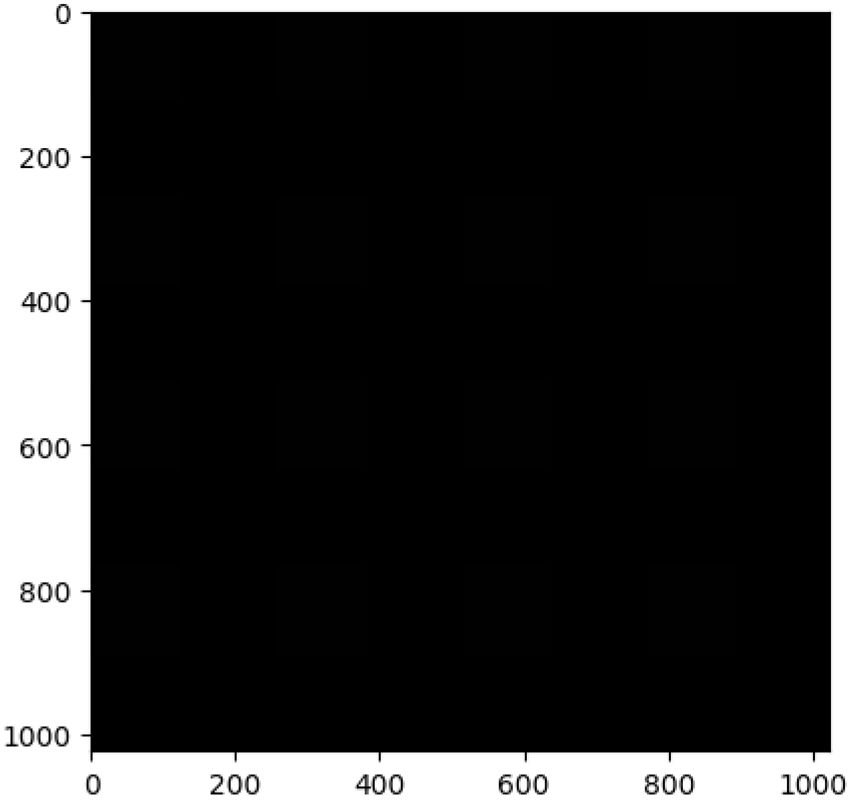


c.

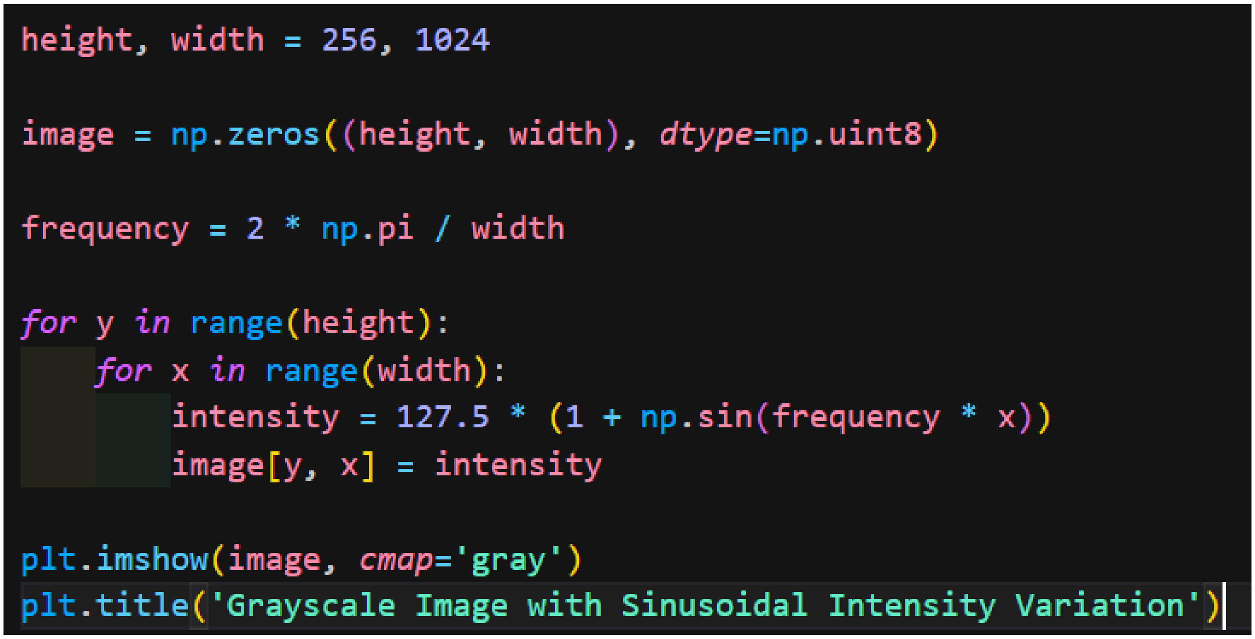


c. Multiplying Images of C and D

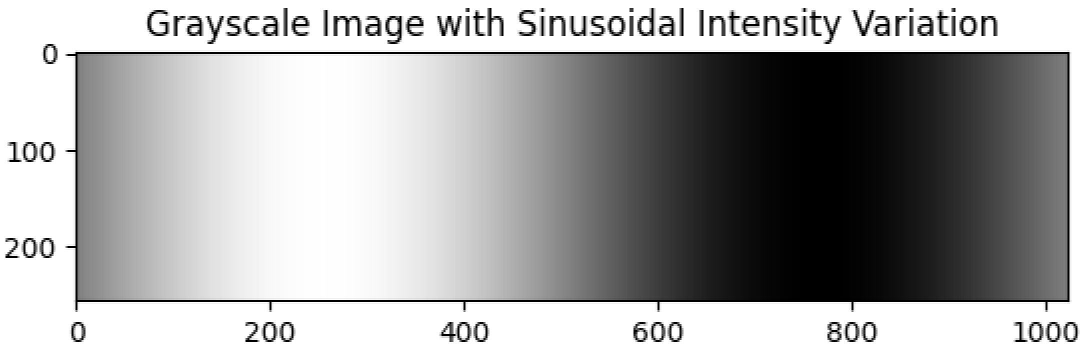
```
plt.imshow(imgC*imgD)
```

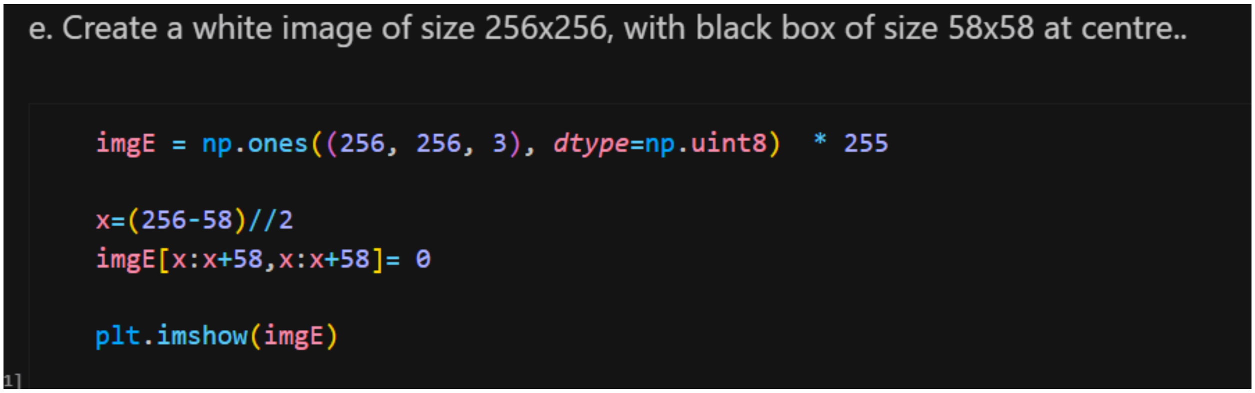


d. Create a grayscale image of size 256 x 1024. Intensity of image should vary sinusoidal.



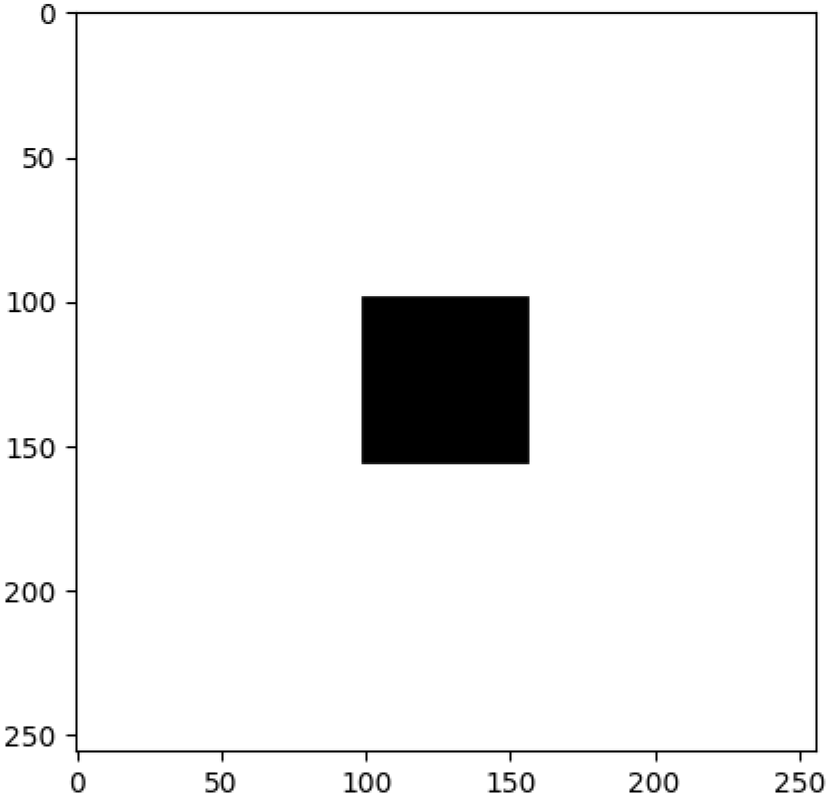
```
height, width = 256, 1024

image = np.zeros((height, width), dtype=np.uint8)

frequency = 2 * np.pi / width

for y in range(height):
    for x in range(width):
        intensity = 127.5 * (1 + np.sin(frequency * x))
        image[y, x] = intensity

plt.imshow(image, cmap='gray')
plt.title('Grayscale Image with Sinusoidal Intensity Variation')
```





e. Create a white image of size 256x256, with black box of size 58x58 at centre..

```
imgE = np.ones((256, 256, 3), dtype=np.uint8)  * 255

x=(256-58)//2
imgE[x:x+58,x:x+58]= 0

plt.imshow(imgE)
```


0
50
100
150
200
250
0
50
100
150
200
250