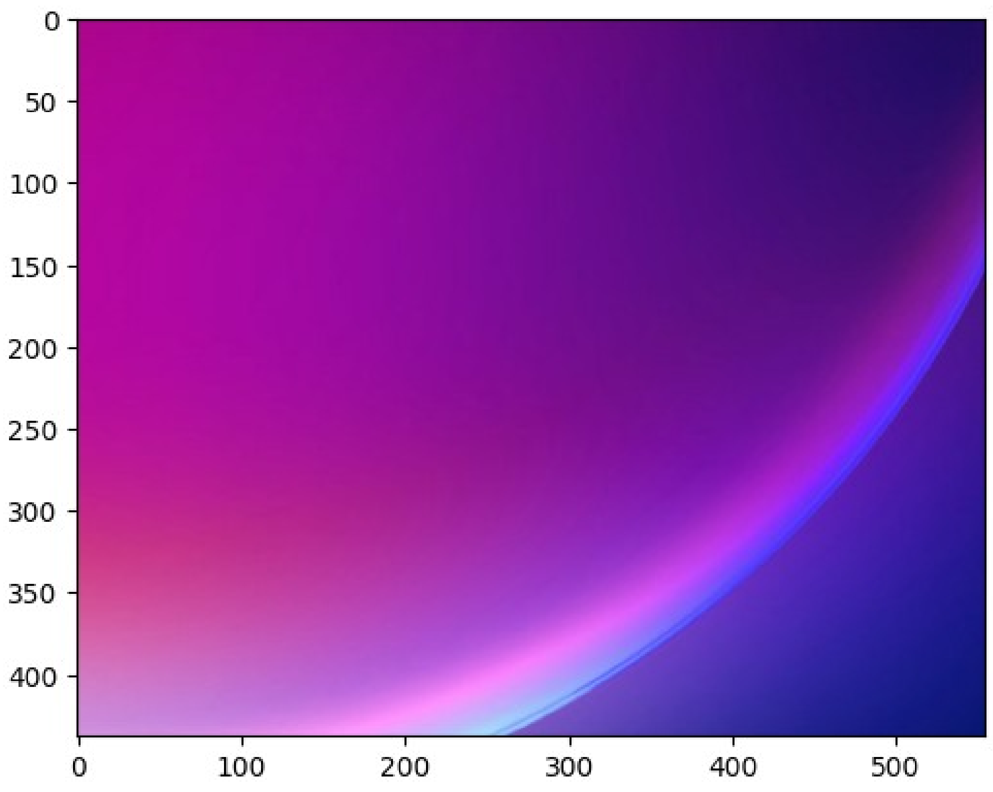
21BAI1300 Lab FAT

Q.1 Convert input colour image to HSI, plot Red, Green, Blue channels and the HSI results.

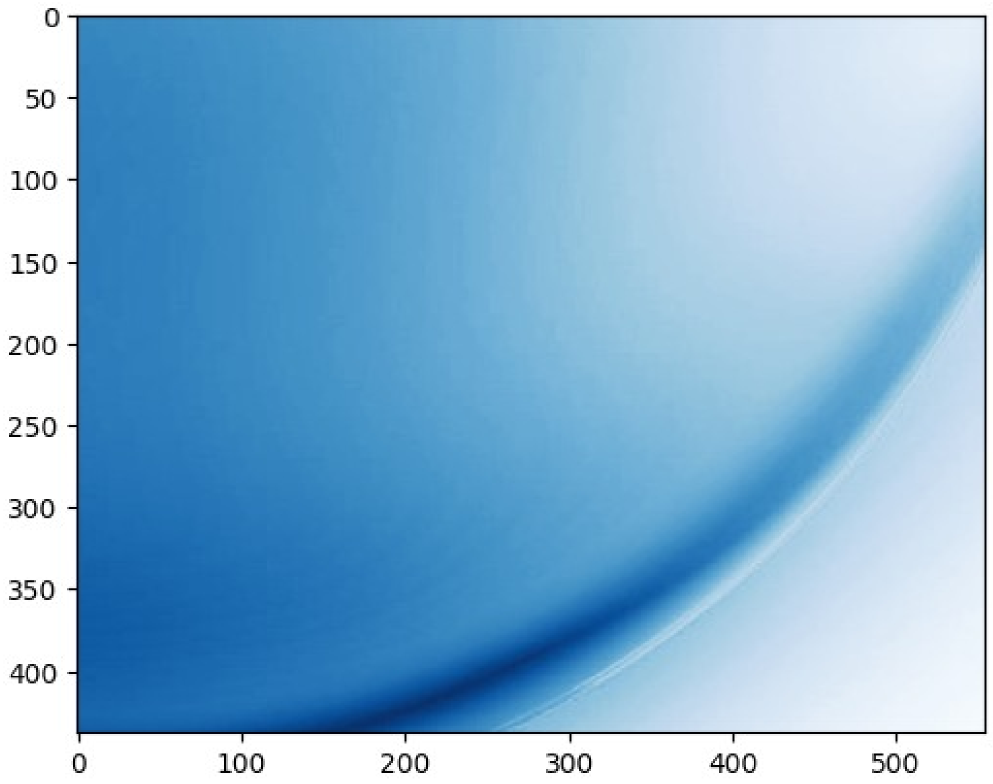
```
import numpy as np
import cv2
from matplotlib import pyplot as plt
from skimage.color import rgb2hsv, hsv2rgb

image = cv2.imread('/content/Screenshot 2024-11-18 081548.png',
cv2.IMREAD_COLOR)

plt.imshow(image)
```

```
<matplotlib.image.AxesImage at 0x7b199ee84a90>
```



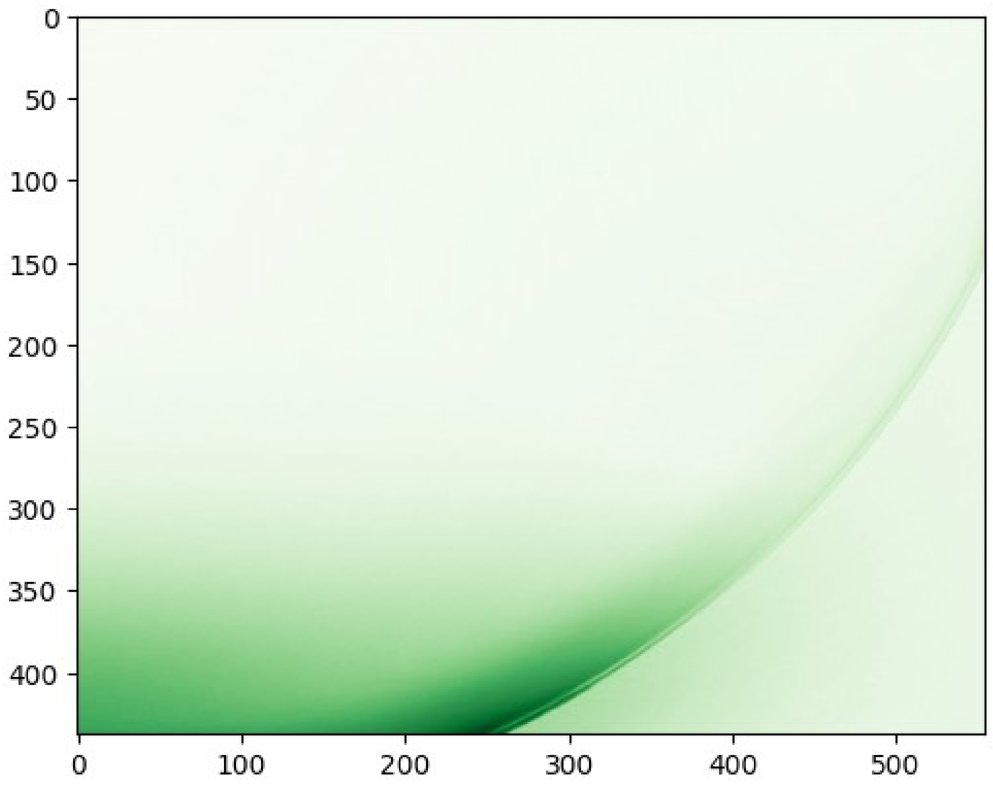
```
b, g, r = cv2.split(image)

plt.imshow(b, cmap='Blues')
```

```
<matplotlib.image.AxesImage at 0x7b1990c2cfa0>
```



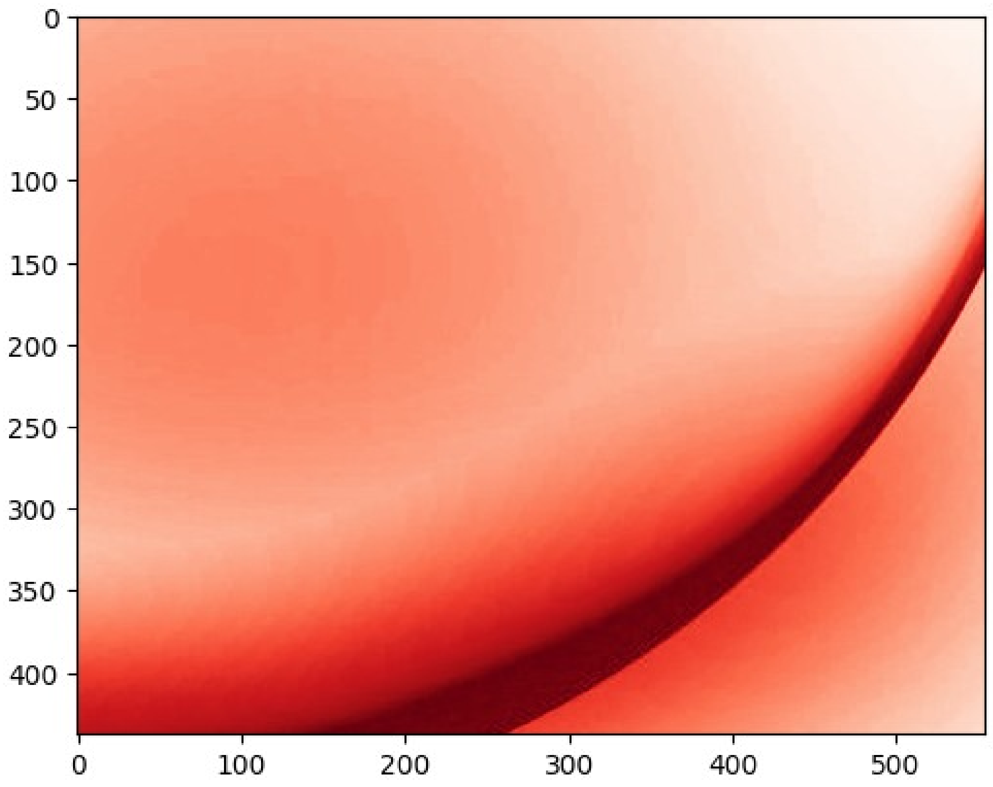
```
plt.imshow(g, cmap='Greens')
```

```
<matplotlib.image.AxesImage at 0x7b1990aba200>
```



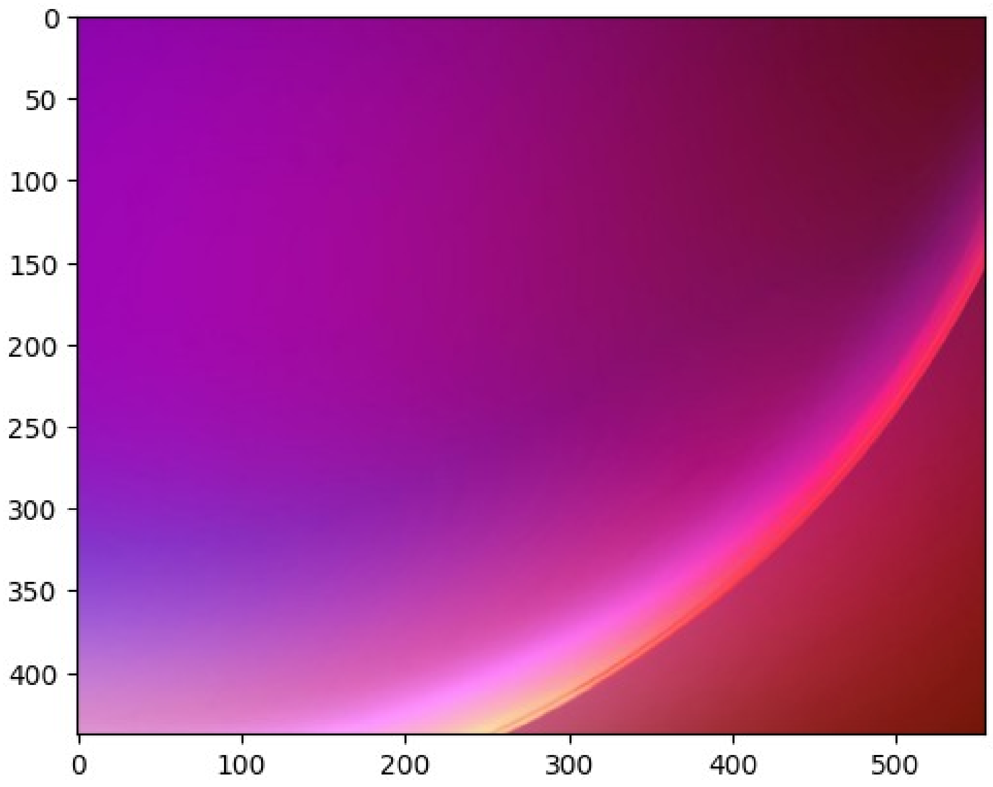
```
plt.imshow(r, cmap='Reds')
```

```
<matplotlib.image.AxesImage at 0x7b19907e3bb0>
```

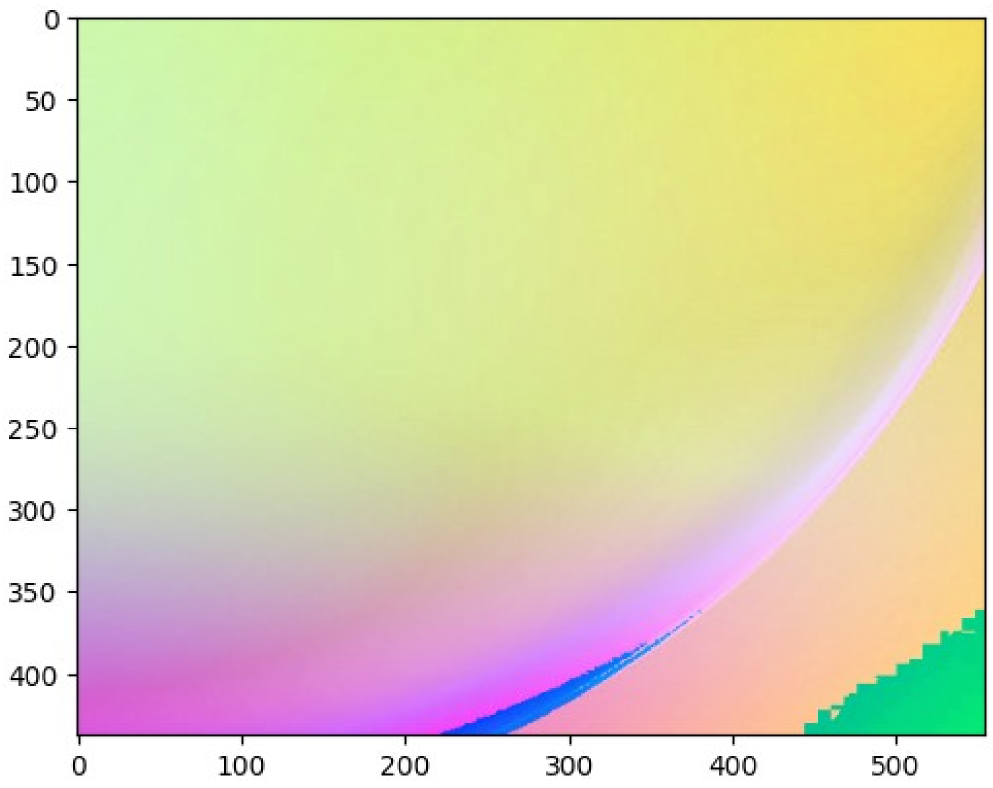


```
# For easier working with image
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.imshow(image_rgb)
```

```
<matplotlib.image.AxesImage at 0x7b199049be80>
```

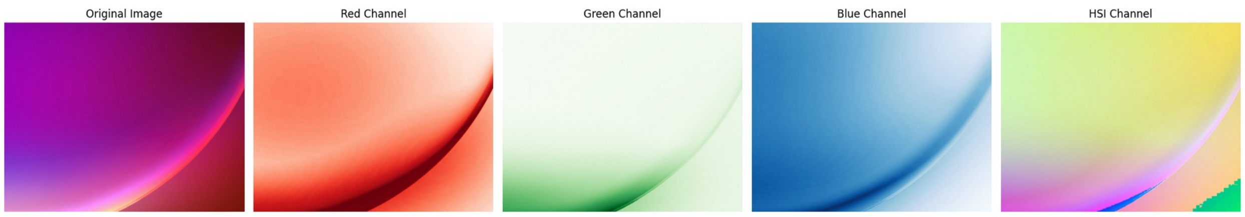


```
from skimage import color

hsi_image = color.rgb2hsv(image_rgb)

plt.imshow(hsi_image)

<matplotlib.image.AxesImage at 0x7b198fe52620>
```



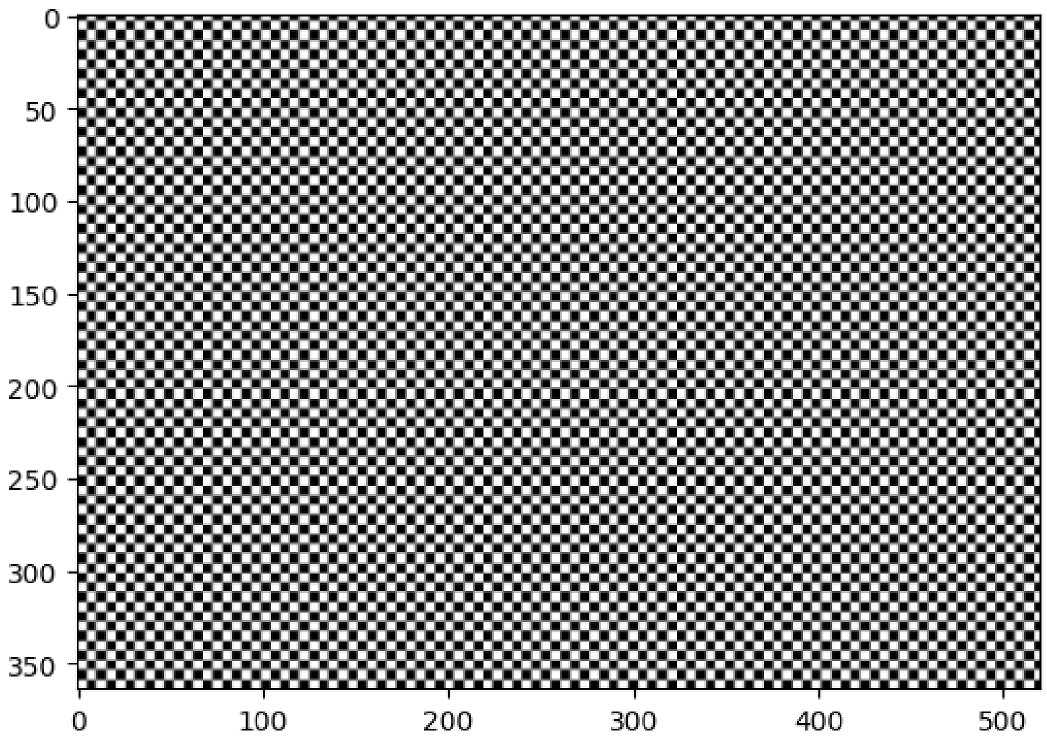
Hence, now we will plot the Red, Green, Blue channels and the HSI converted image with neatness and title

```python
# Display each channel separately
plt.figure(figsize=(20, 20))

plt.subplot(1, 5, 1)
plt.imshow(image_rgb)
plt.title('Original Image')
plt.axis('off')

plt.subplot(1, 5, 2)
plt.imshow(r, cmap='Reds')
plt.title('Red Channel')
plt.axis('off')

plt.subplot(1, 5, 3)
plt.imshow(g, cmap='Greens')
plt.title('Green Channel')
plt.axis('off')

plt.subplot(1, 5, 4)
plt.imshow(b, cmap='Blues')
plt.title('Blue Channel')
plt.axis('off')
plt.subplot(1, 5, 5)
plt.imshow(hsi_image)
plt.title('HSI Channel')
plt.axis('off')
plt.tight_layout()
plt.show()
```



Q.2 Process of computing gray level Co occurence matrix (GLCM). Compute Contrast, Energy, Homogeneity, Correlation.

```
image2 = cv2.imread("/content/Screenshot 2024-11-18
081131.png",cv2.IMREAD_GRAYSCALE) # Since needed in thee format of
gray GLCM

plt.imshow(image2, cmap='gray')

<matplotlib.image.AxesImage at 0x7b198ef86590>
```

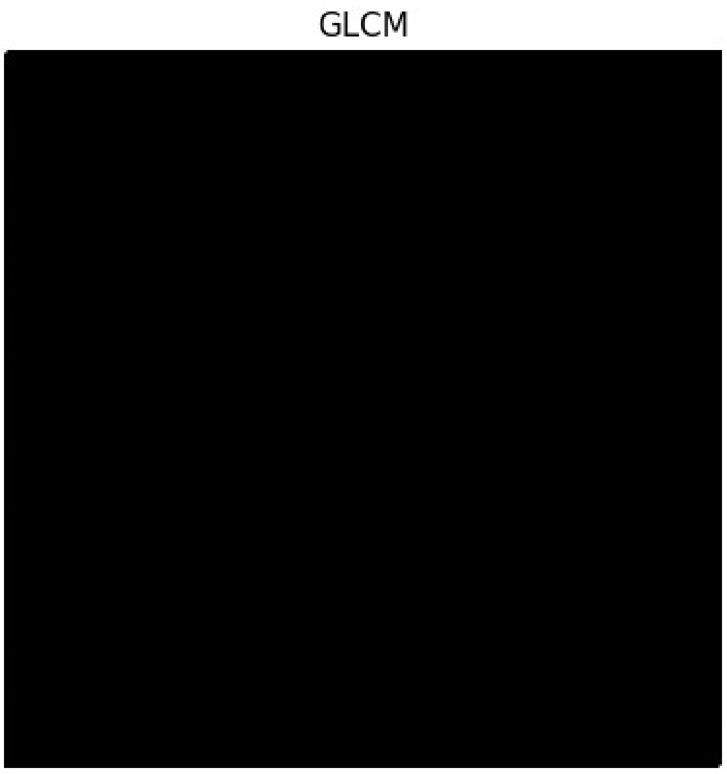


```python
# Function made for "Gray-Level Co-Occurrence Matrix (GLCM)"
Calculation
def calculate_glcm(image, d, theta):
    glcm = np.zeros((256, 256))
    for i in range(image.shape[0] - d):
        for j in range(image.shape[1] - d):
            if theta == 0:
                x = image[i, j]
                y = image[i + d, j]
            elif theta == 45:
                x = image[i, j]
                y = image[i + d, j + d]
            elif theta == 90:
                x = image[i, j]
                y = image[i, j + d]
            elif theta == 135:
                x = image[i, j]
                y = image[i - d, j + d]
            glcm[x, y] += 1
    return glcm

def calculate_glcm_properties(glcm):
    contrast = np.sum(glcm * (glcm.shape[0] -
np.arange(glcm.shape[0])) ** 2)
    homogeneity = np.sum(glcm / (1 + (glcm.shape[0] -
np.arange(glcm.shape[0])) ** 2))
```

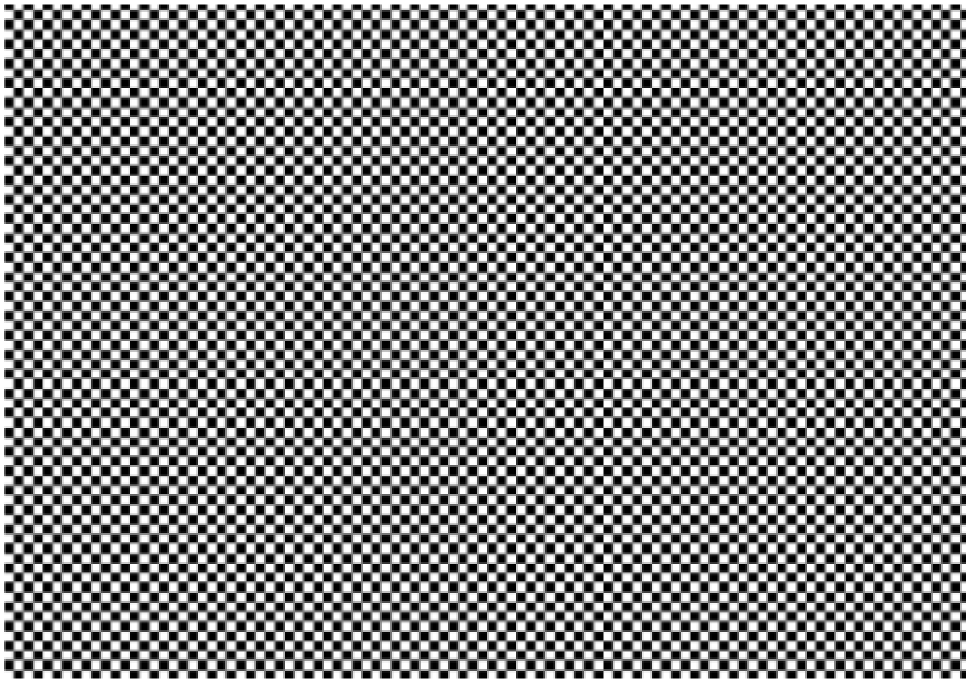
```
    energy = np.sum(glcm ** 2)
    correlation = np.sum(glcm * np.arange(glcm.shape[0]) *
np.arange(glcm.shape[1]))
    return contrast, homogeneity, energy, correlation

glcm = calculate_glcm(image2, 1, 0)
contrast, homogeneity, energy, correlation =
calculate_glcm_properties(glcm)

print("Contrast:", contrast)
print("Homogeneity:", homogeneity)
print("Energy:", energy)
print("Correlation:", correlation)
```

```
Contrast: 5261473580.0
Homogeneity: 23971.04967845306
Energy: 2445290899.0
Correlation: 5192250156.0
```

```
plt.imshow(glcm, cmap='gray')
plt.title('GLCM')
plt.axis('off')
plt.show()
```



```
plt.imshow(image2, cmap='gray')
plt.axis('off')
plt.show()
```



#*Hence, both the questions. Question 1 and question 2 has been successfully solved*