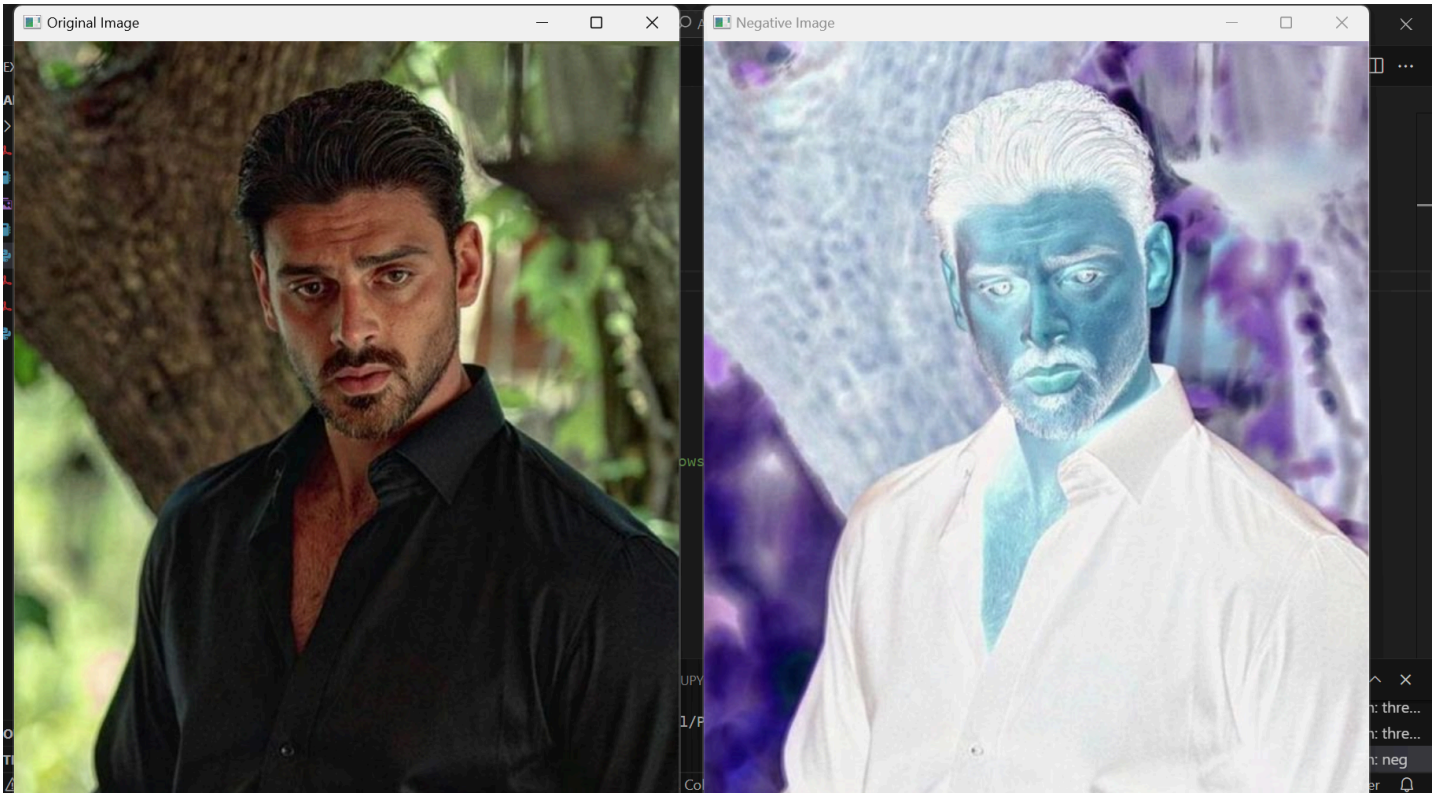
**Practical- 1:**
**Write a Program to display the negative of a digital image.**

**Code:**

```
import cv2
import numpy as np

# Load the image
img = cv2.imread('image.jpg')

# Convert the image to negative
negative_img = cv2.bitwise_not(img)

# Display the original image
cv2.imshow('Original Image', img)

# Display the negative image
cv2.imshow('Negative Image', negative_img)

# Wait for a key press and then close all windows
cv2.waitKey(0)
cv2.destroyAllWindows()
```

**Output:**

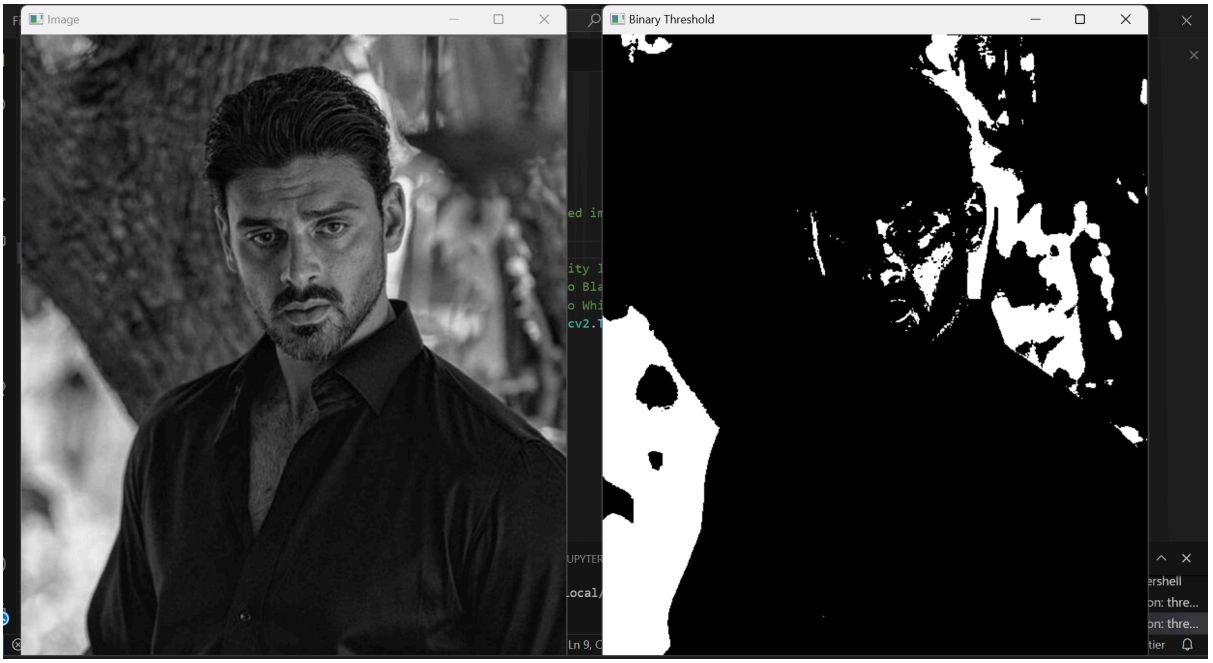


**Practical- 2**
**Write a Program to perform thresholding on an input image.**

**Code:**

```
import cv2

# Loading the image named test.jpg
img = cv2.imread("image.jpg")

# Converting color mode to Grayscale
# as thresholding requires a single channeled image
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imshow('Image', img)
# Thresholding the image placing 127 intensity level as threshold
# Pixel values below 127 would be changed to Black
# Pixel values above 127 would be changed to White (255)
ret, thresh = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)

# Displaying the output image
cv2.imshow('Binary Threshold', thresh)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

**Output:**

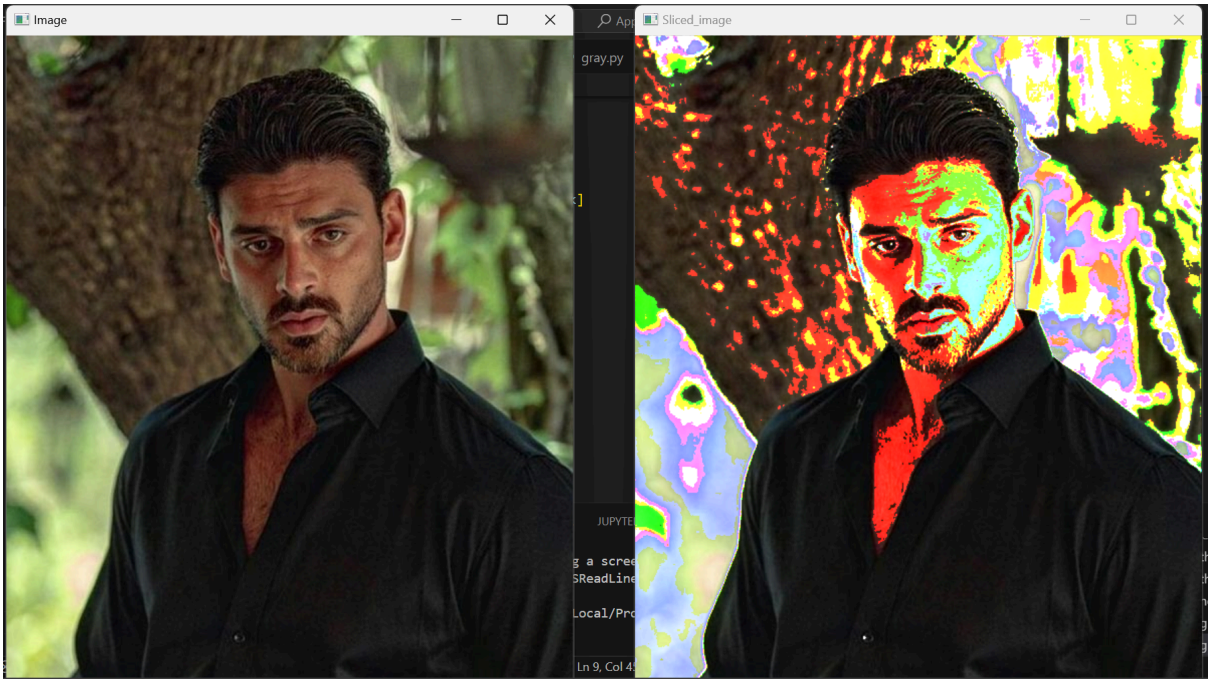


**Practical- 3**
**Write a Program to perform gray level slicing without background.**

**Code:**

```
import numpy as np
import cv2
img = cv2.imread('eyes.jpg')
row, column, layer = img.shape
img1 = np.zeros((row,column, layer), dtype = 'uint8')
min_range = 80
max_range = 140
for i in range(row):
    for j in range(column):
        for k in range(layer):
            if img[i][j][k] in range(min_range, max_range):
                img1[i][j][k] = 255
            else:
                img1[i][j][k] = img[i][j][k]
cv2.imshow("Image",img)
cv2.imshow("Sliced_image",img1)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

**Output:**

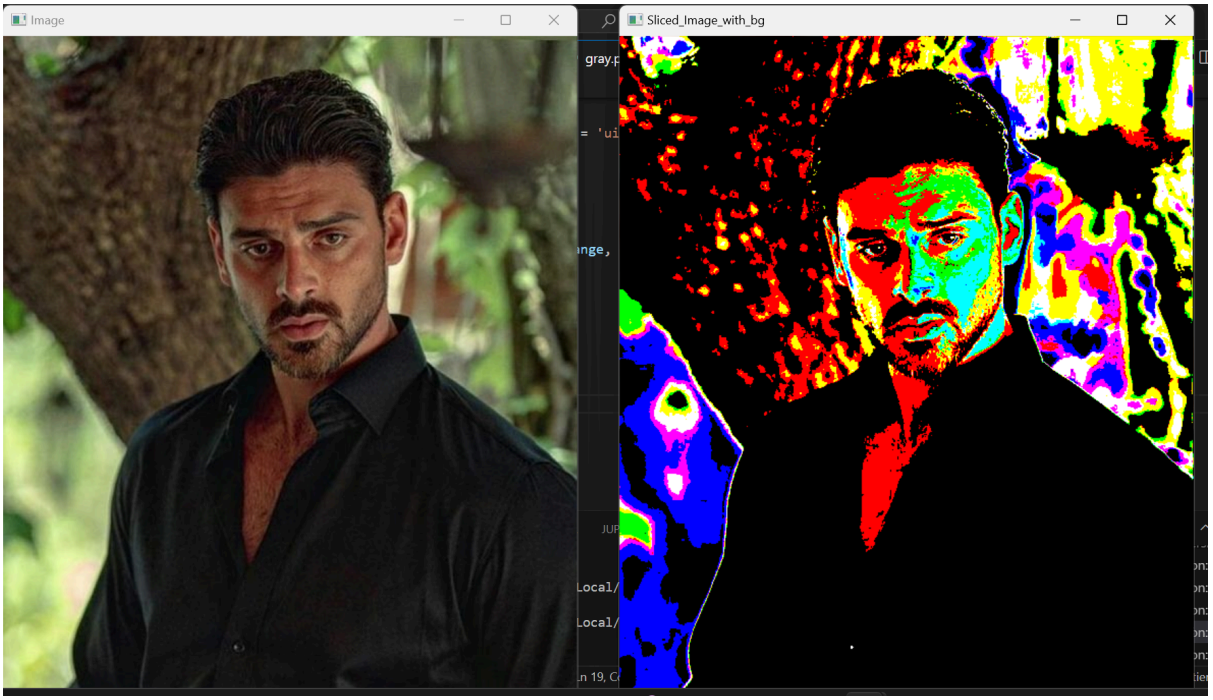


**Practical- 4**
**Write a Program to perform gray level slicing with the background.**

**Code:**

```
import numpy as np
import cv2
img = cv2.imread('image.jpg')
row, column, layer = img.shape
img1 = np.zeros((row,column, layer), dtype = 'uint8')
min_range= 80
max_range = 140
for i in range(row):
    for j in range(column):
        for k in range(layer):
            if img[i][j][k] in range(min_range, max_range):
                img1[i][j][k] = 255

            else:
                img1[i][j][k]=0

cv2.imshow("Image",img)
cv2.imshow("Sliced_Image_with_bg",img1)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

**Output:**

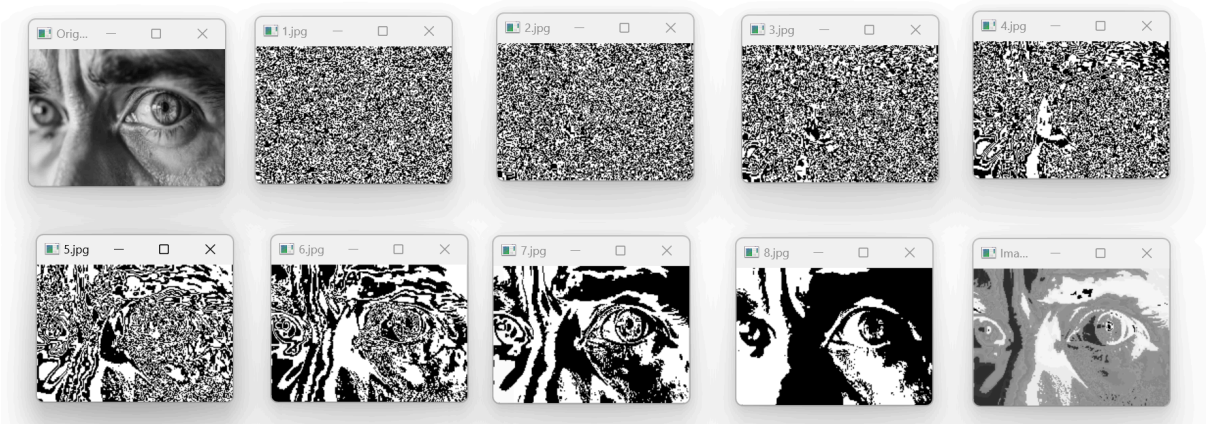


**Practical- 5**
**Write a Program to perform bit-plane slicing**

**Code:**

```
#bit plane slicing
import cv2 as cv

img=cv.imread('eyes.jpg')
img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
img=cv.resize(img, (200,140))
imgs=[255*((img & (1<<i))>>i) for i in range(8)]
cv.imshow('1.jpg',imgs[0])
cv.imshow('2.jpg',imgs[1])
cv.imshow('3.jpg',imgs[2])
cv.imshow('4.jpg',imgs[3])
cv.imshow('5.jpg',imgs[4])
cv.imshow('6.jpg',imgs[5])
cv.imshow('7.jpg',imgs[6])
cv.imshow('8.jpg',imgs[7])
new_img=(imgs[7]*100)+(imgs[6]*100)+(imgs[5]*10)+(imgs[4]*10)
cv.imshow('Image using 8,7,6,5th bits',new_img)
cv.imshow('Original', img)
cv.waitKey(0)
```

**Output:**

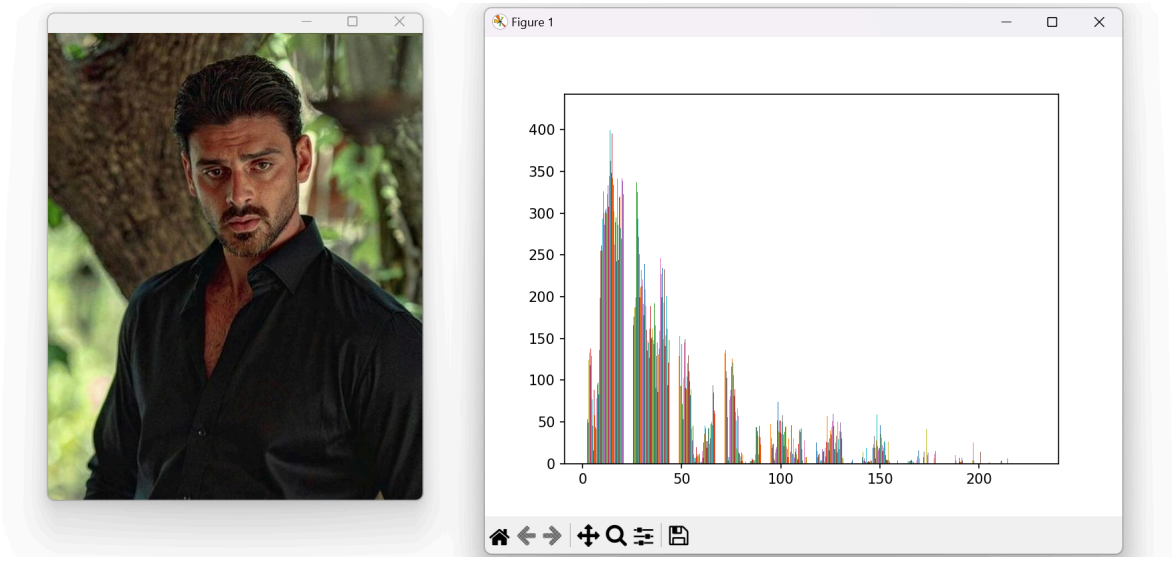


**Practical- 6**
**Write a Program to display a Histogram of an image.**

**Code:**

```
import matplotlib.pyplot as plt #importing matplotlib
import numpy as np
import cv2
img = cv2.imread('image.jpg')
row, column, layer = img.shape
print(row,column)
cv2.imshow("Original", img)
a=[[0 for i in range(column)] for i in range(row)]
for i in range(row):
    for j in range(column):
        a[i][j]=img[i][j][0]
plt.hist(a)
plt.show()
cv2.waitKey(0)
```

**Output:**



**Practical- 7**
**Write a Program to perform a Log Transformation of an image.**

**Code:**

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
# Read an image
image = cv2.imread('GFG.png')
# Apply log transformation method
c = 255 / np.log(1 + np.max(image))
log_image = c * (np.log(image + 1))
# Specify the data type so that
# float value will be converted to int
log_image = np.array(log_image, dtype = np.uint8)
# Display both images
plt.imshow(image)
plt.show()
plt.imshow(log_image)
plt.show()
```

**Output:**