 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy	
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>

**Aim:** Practical based on Image Processing with Numpy

**IDE:**

NumPy for Image Processing

NumPy is a robust tool for image processing in Python.

Importing Libraries

The required libraries: PIL, NumPy, and Matplotlib. PIL is used for opening images. NumPy allows for efficient array operations and image processing. Matplotlib is used for visualizing images

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
```

Crop Image

We define coordinates to mark the area we want to crop from the image. The new image contains only the selected part and discards the rest.

Example:

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
print(img_array)
y1, x1 = 100, 100 # Top-left corner of ROI
y2, x2 = 250, 200 # Bottom-right corner of ROI
cropped_img = img_array[y1:y2, x1:x2]
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img_array)
plt.title('Original Image')
```

**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on Image Processing with Numpy

**Experiment No: 11**

**Date:**

**Enrollment No:92400133037**

```
plt.axis('off')
```

```
plt.subplot(1, 2, 2)
```

```
plt.imshow(cropped_img)
```

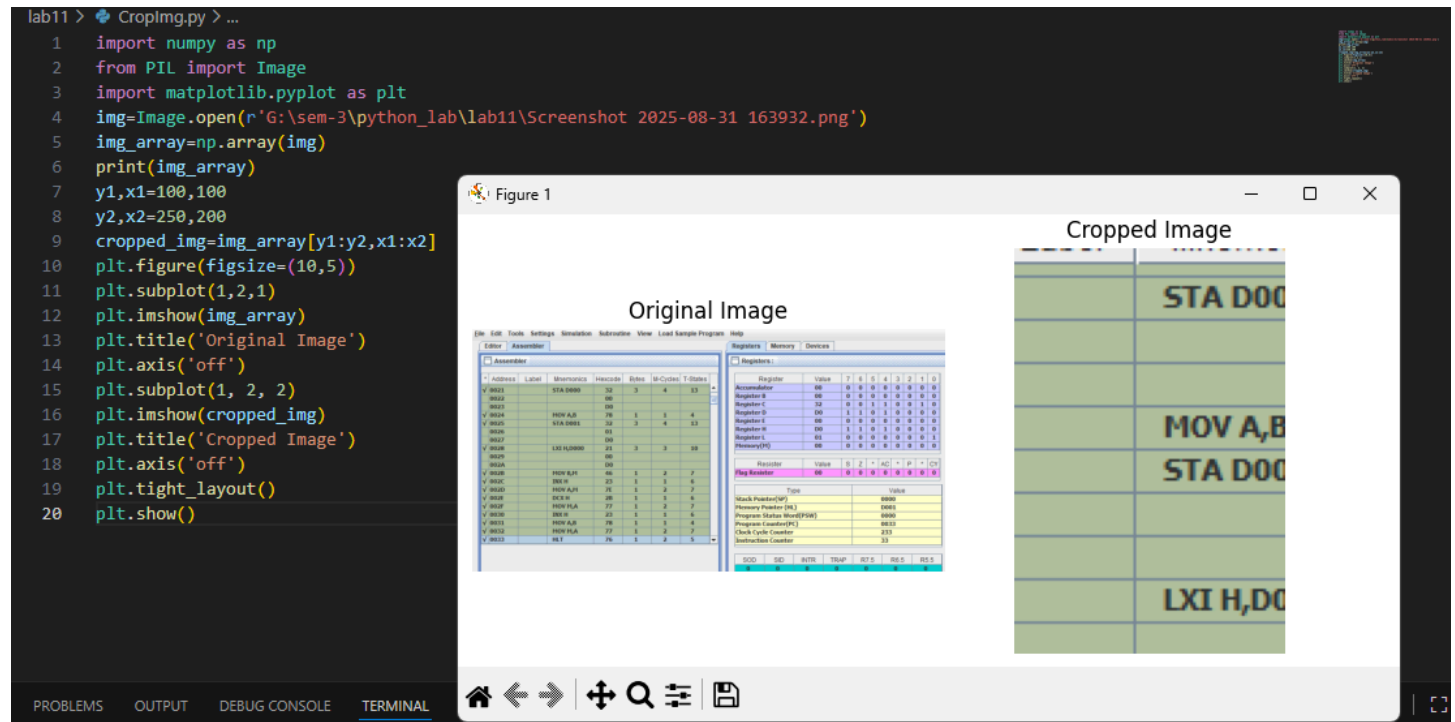
```
plt.title('Cropped Image')
```

```
plt.axis('off')
```

```
plt.tight_layout()
```

```
plt.show()
```

Output



## Rotate Image

We rotate the image array 90 degrees counterclockwise using NumPy's 'rot90' function.

Example:

```
import numpy as np
```

```
from PIL import Image
```

```
import matplotlib.pyplot as plt
```

**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on Image Processing with Numpy

**Experiment No: 11**

**Date:**

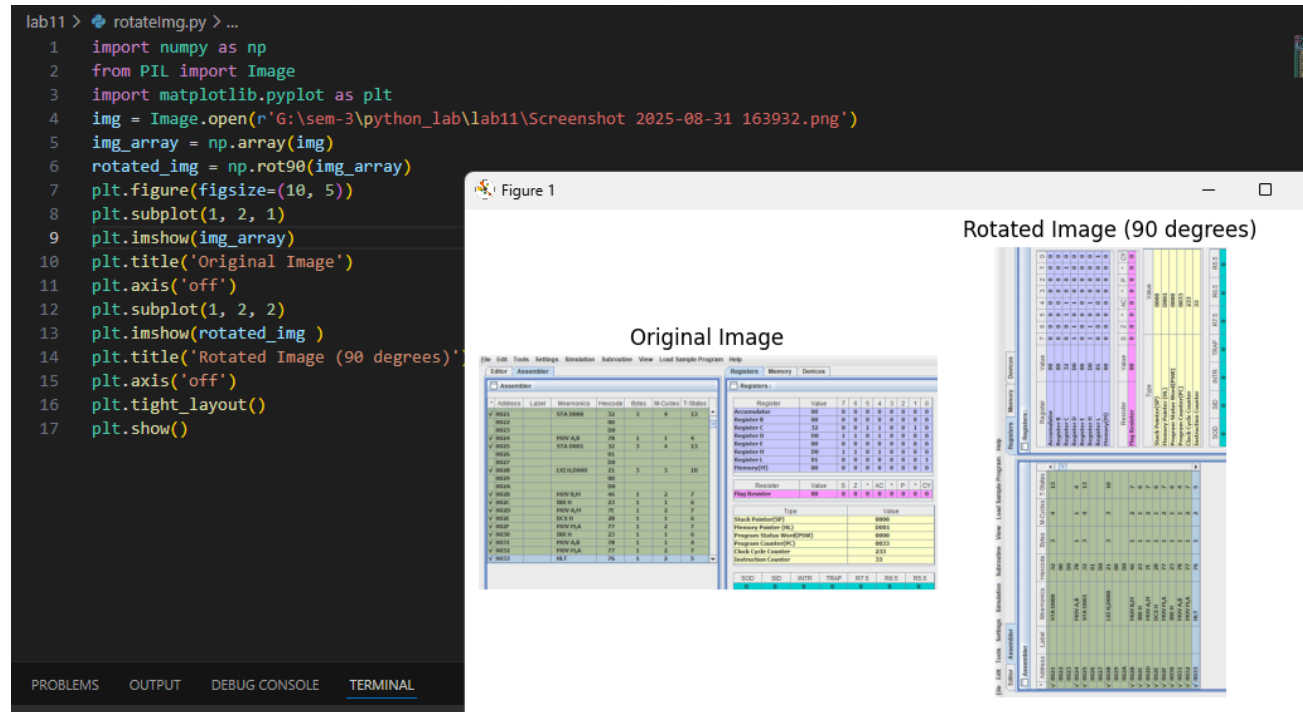
**Enrollment No:92400133037**


```
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
rotated_img = np.rot90(img_array)
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img_array)
plt.title('Original Image')
plt.axis('off')
```

```
plt.subplot(1, 2, 2)
plt.imshow(rotated_img)
plt.title('Rotated Image (90 degrees)')
plt.axis('off')
```

```
plt.tight_layout()
plt.show()
```

Output



 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy	
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>

### Flip Image

We use NumPy's 'fliplr' function to flip the image array horizontally.

Example:

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
flipped_img = np.fliplr(img_array)
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img_array)
plt.title('Original Image')
plt.axis('off')
plt.subplot(1, 2, 2)
plt.imshow(flipped_img )
plt.title('Flipped Image')
plt.axis('off')
plt.tight_layout()
plt.show()
```

Output

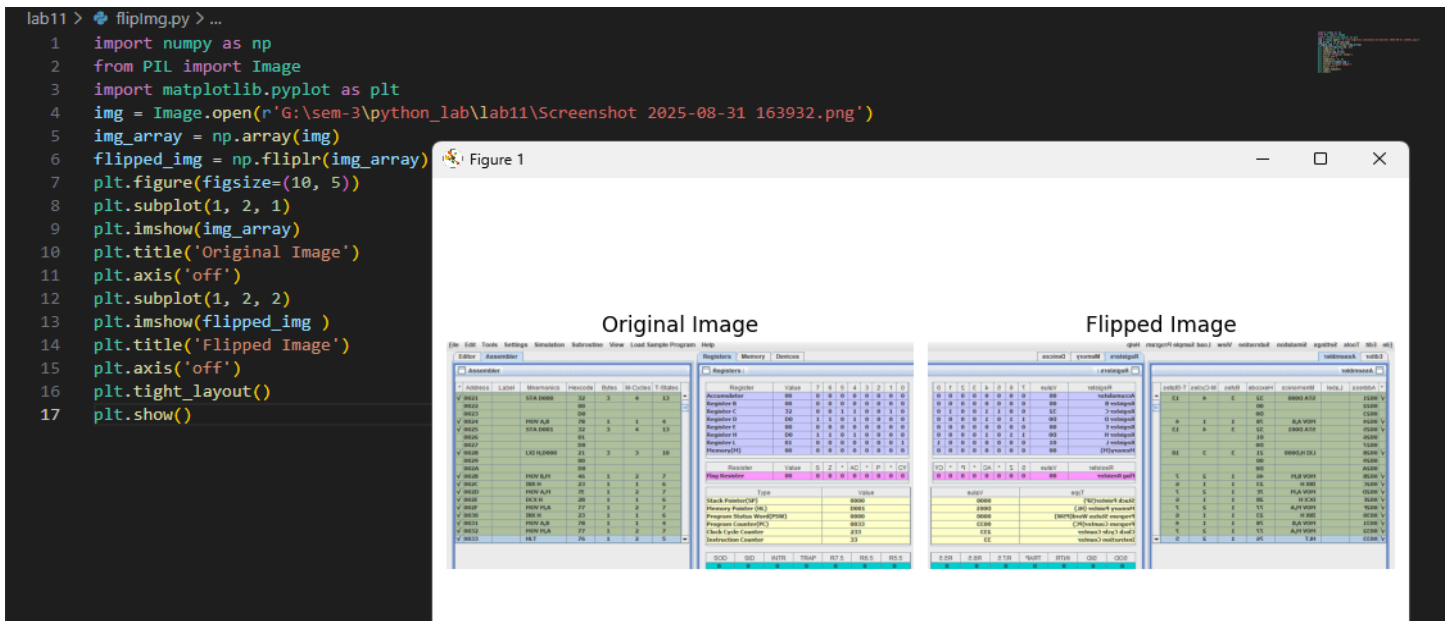
**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on Image Processing with Numpy

**Experiment No: 11**

**Date:**

**Enrollment No:92400133037**



## Negative of an Image


The negative of an image is made by reversing its pixel values. In grayscale images, each pixel's value is subtracted from the maximum (255 for 8-bit images). In color images, this is done separately for each color channel.

Example:

```

import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
is_grayscale = len(img_array.shape) < 3
# Function to create negative of an image
def create_negative(image):
    if is_grayscale:
        # For grayscale images
        negative_image = 255 - image
    else:
  
```

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy	
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>

```

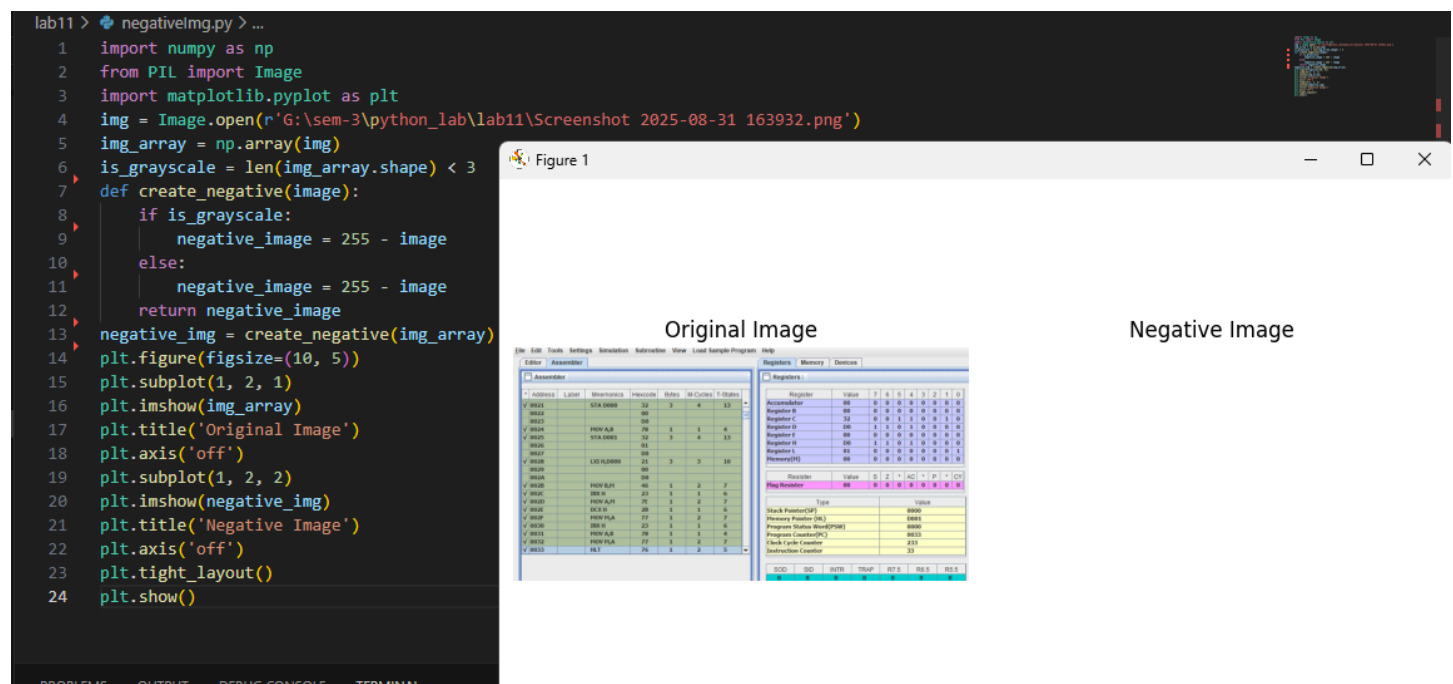
# For color images (RGB)
negative_image = 255 - image
return negative_image


# Create negative of the image
negative_img = create_negative(img_array)

# Display the original and negative images
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img_array)
plt.title('Original Image')
plt.axis('off')
plt.subplot(1, 2, 2)
plt.imshow(negative_img)
plt.title('Negative Image')
plt.axis('off')
plt.tight_layout()
plt.show()

```

Output



 <b>Marwadi University</b> Marwadi Chandarana Group	NAAC <b>A+</b>	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy		
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>	

### Binarize Image

Binarizing an image converts it to black and white. Each pixel is marked black or white based on a threshold value. Pixels that are less than the threshold become 0 (black) and above those above it become 255 (white).

### Example

```
import numpy as np
from PIL import Image, ImageOps
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
# Binarize the image using a threshold
threshold = 128
binary_img = np.where(img_array < threshold, 0, 255).astype(np.uint8)
# Display the original and binarized images
plt.figure(figsize= (10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img_array, cmap='gray')
plt.title('Original Grayscale Image')
plt.axis('off')
plt.subplot(1, 2, 2)
plt.imshow(binary_img, cmap='gray')
plt.title('Binarized Image (Threshold = 128)')
plt.axis('off')
plt.tight_layout()
plt.show()
```

Output

**Subject: Programming With Python (01CT1309)**

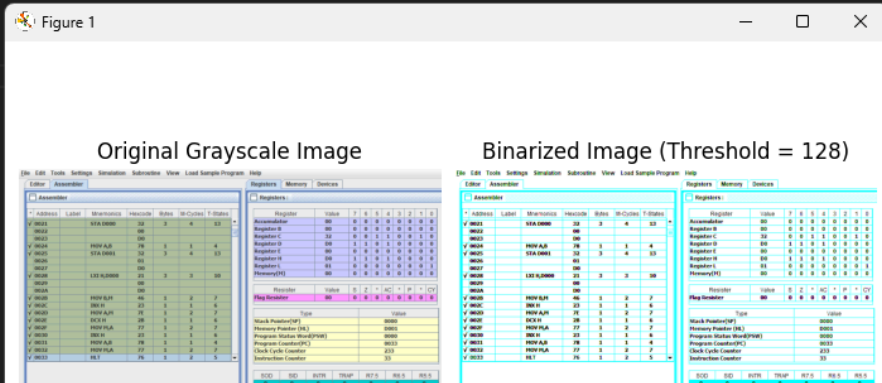
**Aim:** Practical based on Image Processing with Numpy

**Experiment No: 11**

**Date:**

**Enrollment No:92400133037**

```
lab11 > BinariseImg.py > ...
1 import numpy as np
2 from PIL import Image, ImageOps
3 import matplotlib.pyplot as plt
4 img=Image.open(r'G:\sem-3\python_lab\lab11\Screenshot 2025-08-31 163932.png')
5 img_array = np.array(img)
6 threshold = 128
7 binary_img = np.where(img_array < threshold, 0, 255).astype(np.uint8)
8 plt.figure(figsize= (10, 5))
9 plt.subplot(1, 2, 1)
10 plt.imshow(img_array, cmap='gray')
11 plt.title('Original Grayscale Image')
12 plt.axis('off')
13 plt.subplot(1, 2, 2)
14 plt.imshow(binary_img, cmap='gray')
15 plt.title('Binarized Image (Threshold = 128)')
16 plt.axis('off')
17 plt.tight_layout()
18 plt.show()
```



## Color Space Conversion

Color space conversion changes an image from one color model to another. This is done by changing the array of pixel values. We use a weighted sum of the RGB channels to convert a color image to a grayscale.

### Example

```
import numpy as np
from PIL import Image, ImageOps
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
# Grayscale conversion formula: Y = 0.299*R + 0.587*G + 0.114*B
gray_img = np.dot (img_array[...,:3], [0.299, 0.587, 0.114])
# Display the original RGB image
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img_array)
plt.title('Original RGB Image')
plt.axis('off')
```



**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on Image Processing with Numpy

**Experiment No: 11**

**Date:**

**Enrollment No:92400133037**

# Display the converted grayscale image

```
plt.subplot(1, 2, 2)
```

```
plt.imshow(gray_img, cmap='gray')
```

```
plt.title('Grayscale Image')
```

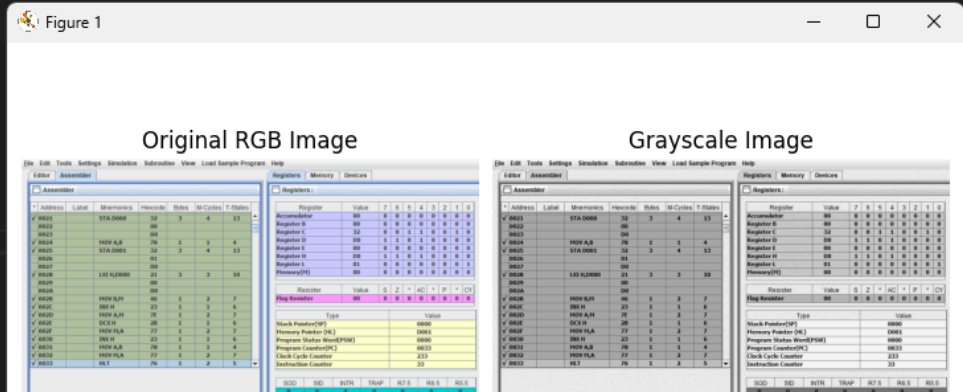
```
plt.axis('off')
```

```
plt.tight_layout()
```

```
plt.show()
```

Output

```
lab11 > ColorSpace.py > ...
1 import numpy as np
2 from PIL import Image, ImageOps
3 import matplotlib.pyplot as plt
4 img=Image.open(r'G:\sem-3\python_lab\lab11\Screenshot 2025-08-31 163932.png')
5 img_array = np.array(img)
6 gray_img = np.dot (img_array[... , :3], [0.299, 0.587, 0.114])
7 plt.figure(figsize=(10, 5))
8 plt.subplot(1, 2, 1)
9 plt.imshow(img_array)
10 plt.title('Original RGB Image')
11 plt.axis('off')
12 plt.subplot(1, 2, 2)
13 plt.imshow(gray_img, cmap='gray')
14 plt.title('Grayscale Image')
15 plt.axis('off')
16 plt.tight_layout()
17 plt.show()
```



**Pixel Intensity Histogram**

The histogram shows the distribution of pixel values in an image. The image is flattened into a one-dimensional array to compute the histogram.


Example:

```
import numpy as np
```

```
from PIL import Image, ImageOps
```

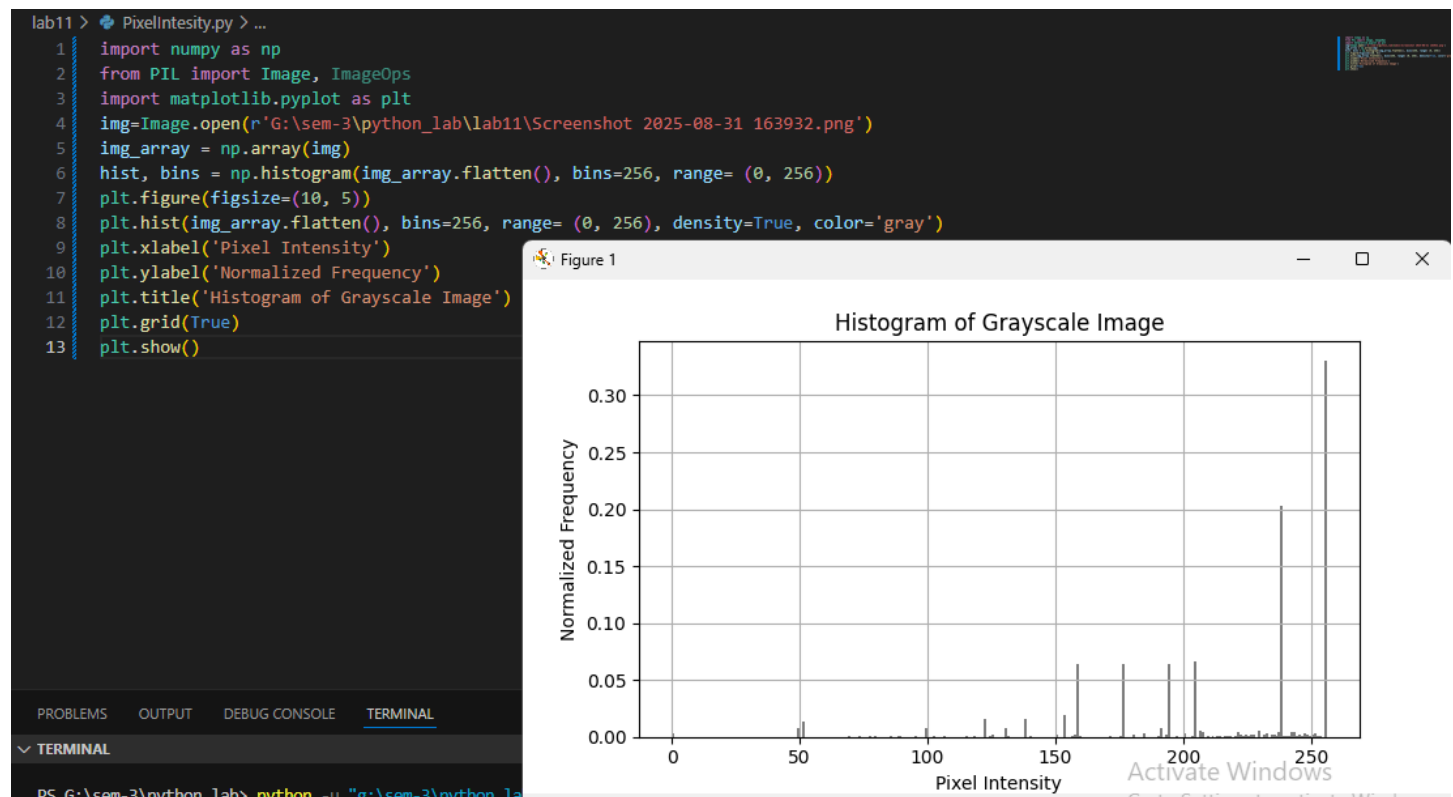
```
import matplotlib.pyplot as plt
```


```
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
```

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy	
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>

```
img_array = np.array(img)
# Compute the histogram of the image
hist, bins = np.histogram(img_array.flatten(), bins=256, range= (0, 256))
# Plot the histogram
plt.figure(figsize=(10, 5))
plt.hist(img_array.flatten(), bins=256, range= (0, 256), density=True, color='gray')
plt.xlabel('Pixel Intensity')
plt.ylabel('Normalized Frequency')
plt.title('Histogram of Grayscale Image')
plt.grid(True)
plt.show()
```

Output



 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy	
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>

### Post Lab Exercise:

- Write a Python program to display details of an image (dimension of an image, shape of an image, min pixel value at channel B).
- Write a Python program to padding black spaces
- Write a Python program to visualize RGB channels

More Practice


Reference :

<https://www.analyticsvidhya.com/blog/2021/05/image-processing-using-numpy-with-practical-implementation-and-code/>

```

lab11 > PostLab.py > ...
1  import numpy as np
2  from PIL import Image
3  import matplotlib.pyplot as plt
4  img = Image.open("G:\sem-3\python_lab\lab11\Screenshot 2025-08-31 163932.png")
5  img_array = np.array(img)
6
7  print("Dimensions (H x W x C):", img_array.shape)
8  print("Shape of image:", img_array.shape)
9
10 min_blue = img_array[:, :, 2].min()
11 print("Minimum pixel value in Blue channel:", min_blue)
12
13
14 img = Image.open("G:\sem-3\python_lab\lab11\Screenshot 2025-08-31 163932.png").convert("RGB")
15 img_array = np.array(img)
16
17 padded_array = np.pad(
18     img_array,
19     pad_width=((100,100),(100,100),(0,0)), # (rows, cols, channels)
20     mode='constant',
21     constant_values=0 # black
22 )
23
24 padded_img = Image.fromarray(padded_array)
25

```

 <b>Marwadi University</b> Marwadi Chandarana Group	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy	
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>

```

26 plt.imshow(padded_img)
27 plt.title("With Black Padding (np.pad)")
28 plt.axis("off")
29 plt.show()
30
31
32
33 img = Image.open("G:\sem-3\python_lab\lab11\Screenshot 2025-08-31 163932.png")
34 img_array = np.array(img)
35
36 R = img_array[:, :, 0]
37 G = img_array[:, :, 1]
38 B = img_array[:, :, 2]
39
40 plt.figure(figsize=(12,4))
41
42 plt.subplot(1,3,1)
43 plt.title("Red Channel")
44 plt.imshow(R, cmap="Reds")
45 plt.axis("off")
46
47 plt.subplot(1,3,2)
48 plt.title("Green Channel")
49 plt.imshow(G, cmap="Greens")
50 plt.axis("off")
51
52 plt.subplot(1,3,3)
53 plt.title("Blue Channel")
54 plt.imshow(B, cmap="Blues")
55 plt.axis("off")
56
57 plt.show()

```

**Subject: Programming With Python (01CT1309)**

**Aim:** Practical based on Image Processing with Numpy

**Experiment No: 11**

**Date:**

**Enrollment No:92400133037**

Figure 1

With Black Padding (np.pad)

Assembler						
* Address	Label	Mnemonics	Hexcode	Bytes	M-Cycles	T-States
✓ 0021	STA D000	32	3	4	13	
0022		00				
0023		00				
✓ 0024	MOV A,B	78	1	1	4	
✓ 0025	STA D001	32	3	4	13	
0026		01				
0027		00				
✓ 0028	LXI H,D000	21	3	3	10	
0029		00				
002A		00				
✓ 002B	MOV B,H	46	1	2	7	
✓ 002C	INX H	23	1	1	6	
✓ 002D	MOV A,H	7E	1	2	7	
✓ 002E	DCX H	2B	1	1	6	
✓ 002F	MOV PLA	77	1	2	7	
✓ 0030	INX H	23	1	1	6	
✓ 0031	MOV A,B	78	1	1	4	
✓ 0032	MOV PLA	77	1	2	7	
✓ 0033	HLT	76	1	2	5	

Register	Value	7	6	5	4	3	2	1	0
Accumulator	00	0	0	0	0	0	0	0	0
Register B	00	0	0	0	0	0	0	0	0
Register C	32	0	0	1	1	0	0	1	0
Register D	00	1	1	0	1	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	00	1	1	0	1	0	0	0	0
Register L	01	0	0	0	0	0	0	0	1
Memory(H)	00	0	0	0	0	0	0	0	0

Register	Value	S	Z	*	AC	*	P	*	CY
Flag Register	00	0	0	0	0	0	0	0	0

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	0001
Program Status Word(PSW)	0000
Program Counter(PC)	0033
Clock Cycle Counter	213
Instruction Counter	33

SOD	SID	INTR	TRAP	RT 5	RS 5	RS 5
0	0	0	0	0	0	0



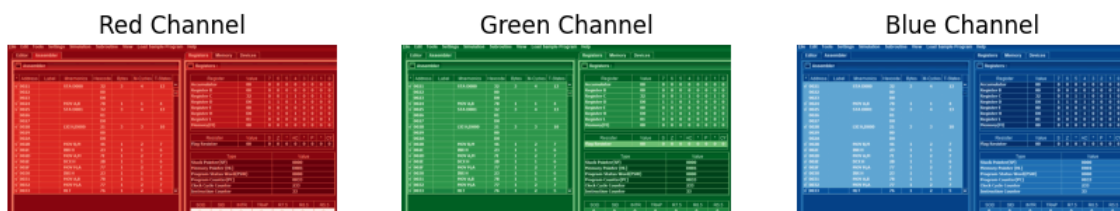
 <b>Marwadi University</b> Marwadi Chandarana Group 	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Practical based on Image Processing with Numpy	
<b>Experiment No: 11</b>	<b>Date:</b>	<b>Enrollment No:92400133037</b>

Figure 1



**GITHUB LINK:**

[https://github.com/Heer972005/Python\\_Lab](https://github.com/Heer972005/Python_Lab)