Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

GITHUB

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)
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

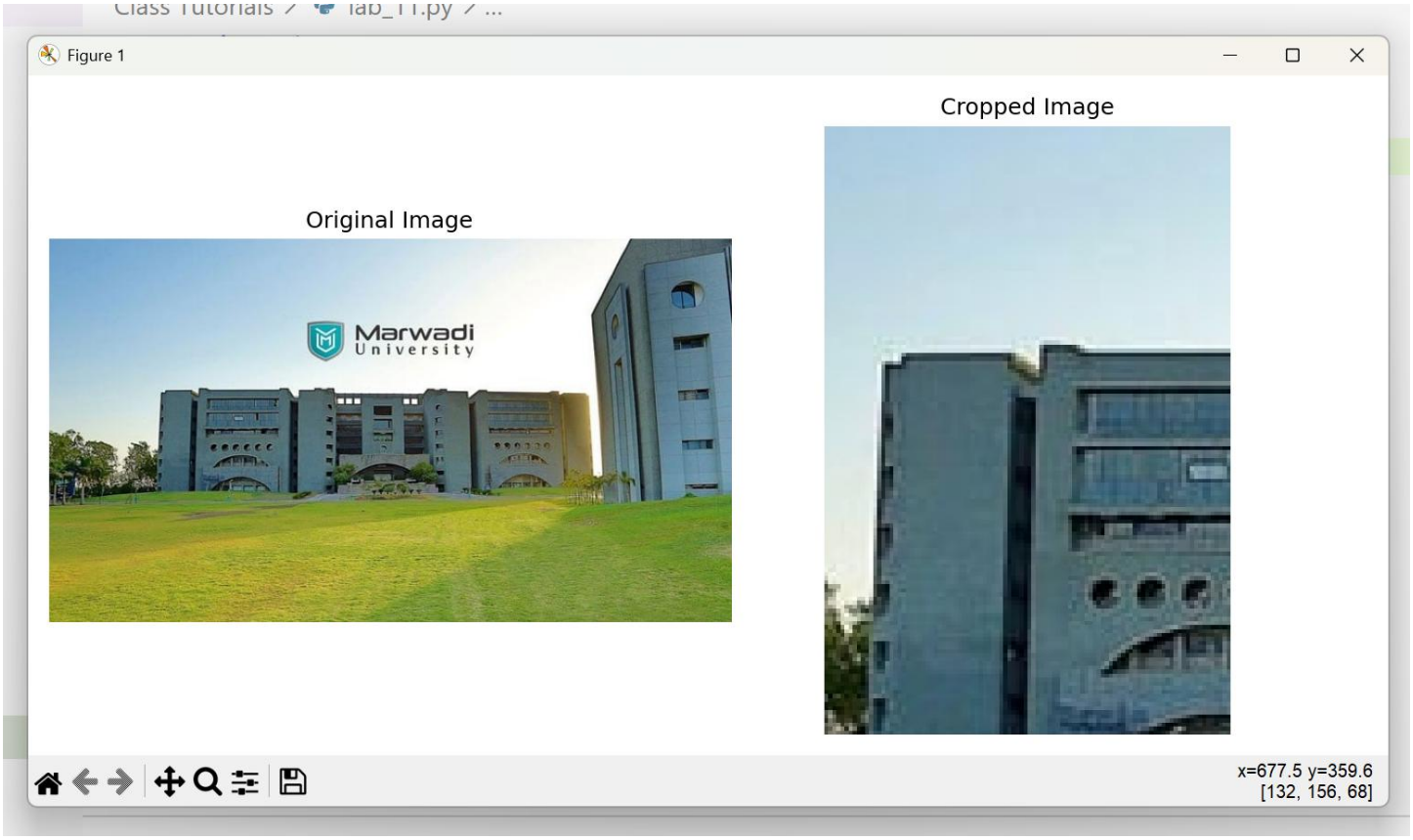
```
plt.imshow(img_array)
plt.title('Original Image')
plt.axis('off')

plt.subplot(1, 2, 2)
plt.imshow(cropped_img)
plt.title('Cropped Image')
plt.axis('off')
plt.tight_layout()
plt.show()
```

Output

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

```python
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
img = Image.open(r"E:\SEM 3\PWP\Class Tutorials\MU.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')
plt.axis('off')

plt.subplot(1, 2, 2)
plt.imshow(cropped_img)
plt.title('Cropped Image')
plt.axis('off')
plt.tight_layout()
plt.show()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



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
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')
```

	Marwadi University **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

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

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

Output

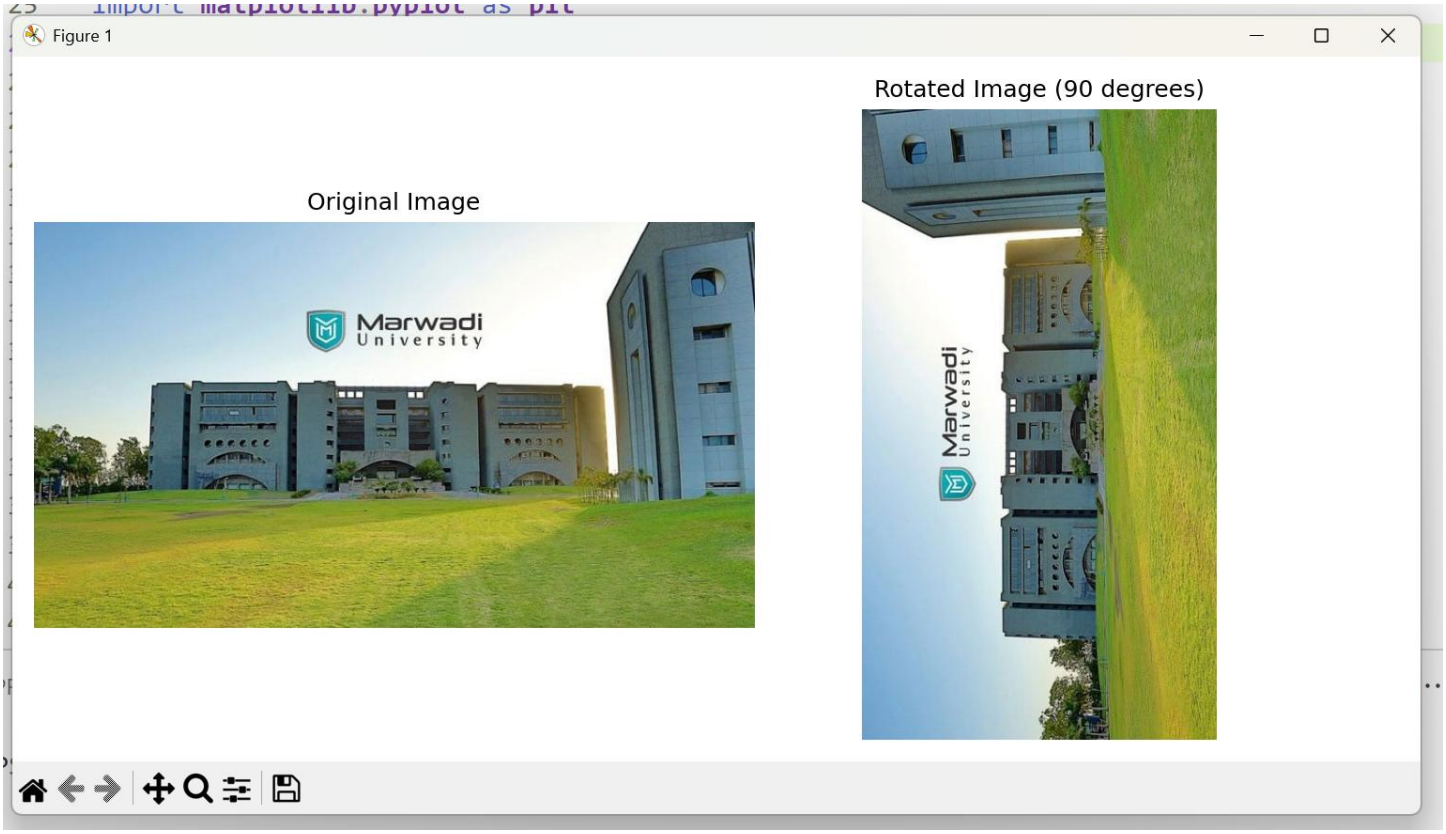
Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

```python
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
img = Image.open(r'E:\SEM 3\PWP\Class Tutorials\MU.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()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_11.py"
```

Marwadi University **Faculty of Engineering & Technology** **Department of Information and Communication Technology**		
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



Flip Image

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

Example:

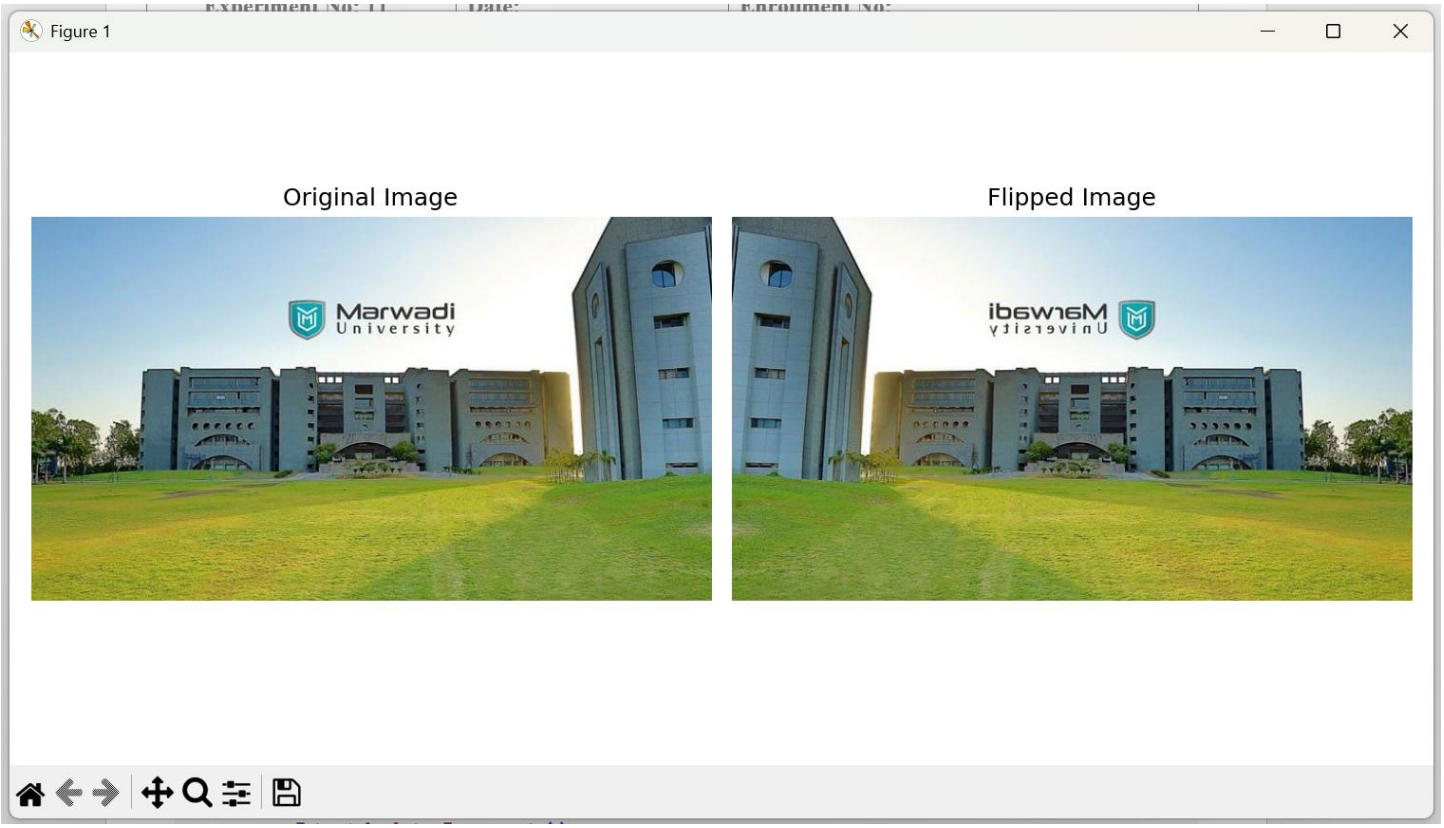
```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
flipped_img = np.fliplr(img_array)
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img_array)
plt.title('Original Image')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

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

plt.title('Flipped Image')
plt.axis('off')
plt.tight_layout()
plt.show()
Output

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
img = Image.open(r'E:\SEM 3\PWP\Class Tutorials\MU.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()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



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:
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

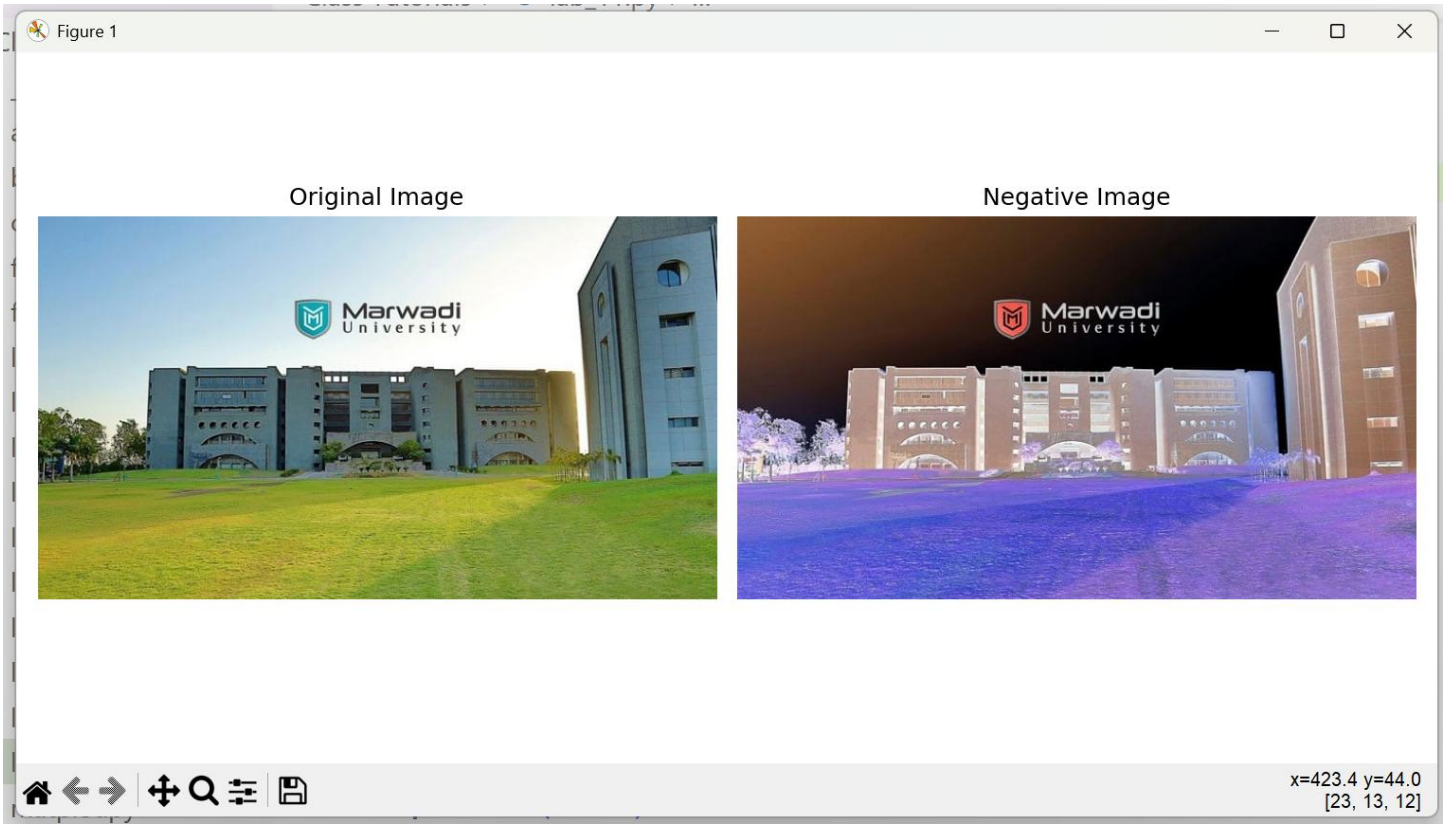
```
        # For grayscale images
        negative_image = 255 - image
    else:
        # For color images (RGB)
        negative_image = 255 - image
    return negative_image
# Create negative of the image
negative_img = create_negative(img_array)
# Display the original and negative images
plt.figure(figsize=(10, 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

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

```python
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
img = Image.open(r'E:\SEM 3\PWP\Class Tutorials\MU.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:
        # For color images (RGB)
        negative_image = 255 - image
    return negative_image
negative_img = create_negative(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(negative_img)
plt.title('Negative Image')
plt.axis('off')
plt.tight_layout()
plt.show()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



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

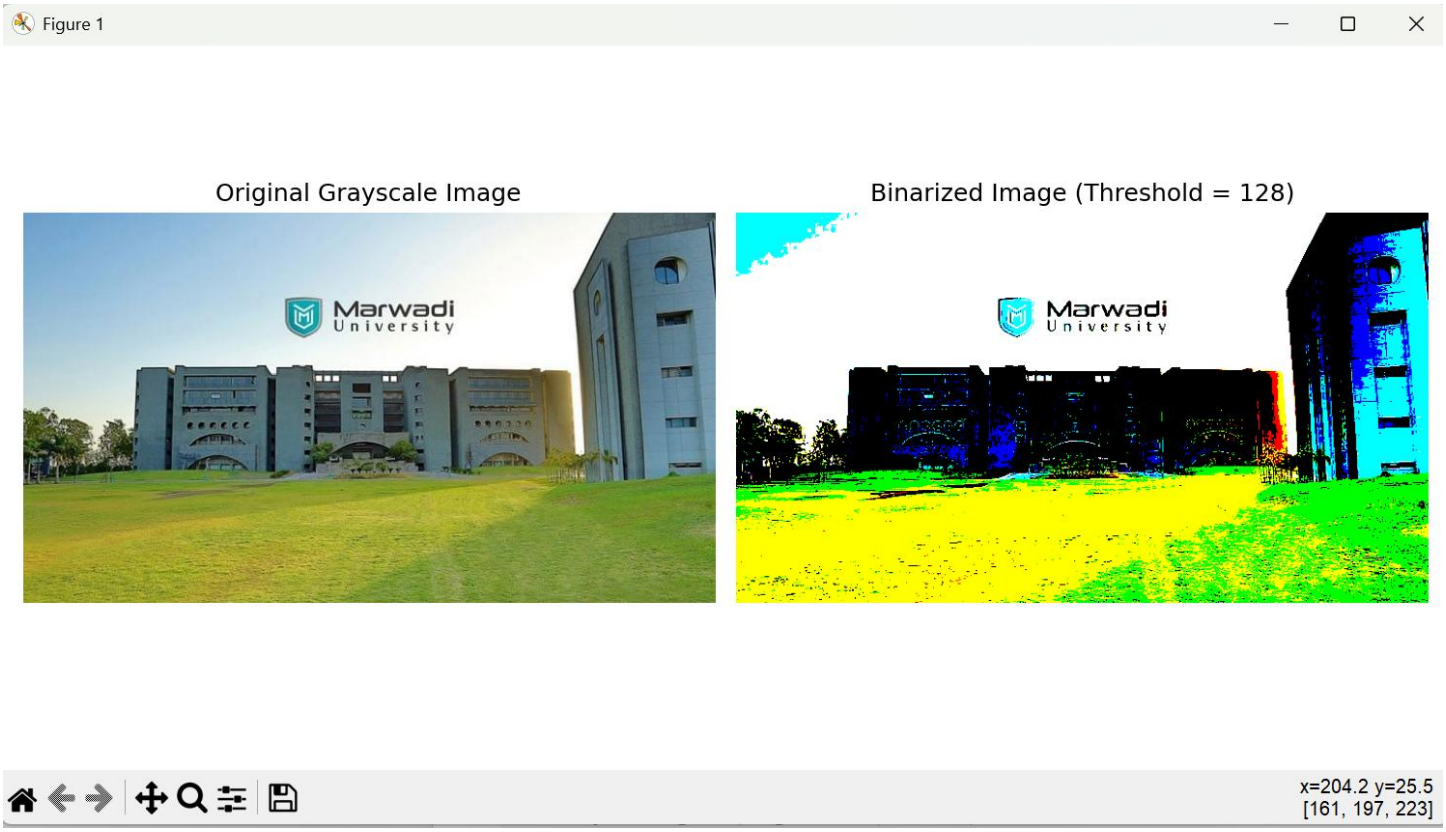
```
import numpy as np
from PIL import Image, ImageOps
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
# Binarize the image using a threshold
threshold = 128
binary_img = np.where(img_array < threshold, 0, 255).astype(np.uint8)
# Display the original and binarized images
plt.figure(figsize= (10, 5))
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

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

```
70  import numpy as np
71  from PIL import Image, ImageOps
72  import matplotlib.pyplot as plt
73  img = Image.open(r'E:\SEM 3\PWP\Class Tutorials\MU.jpg')
74  img_array = np.array(img)
75  # Binarize the image using a threshold
76  threshold = 128
77  binary_img = np.where(img_array < threshold, 0, 255).astype(np.uint8)
78  # Display the original and binarized images
79  plt.figure(figsize= (10, 5))
80  plt.subplot(1, 2, 1)
81  plt.imshow(img_array, cmap='gray')
82  plt.title('Original Grayscale Image')
83  plt.axis('off')
84  plt.subplot(1, 2, 2)
85  plt.imshow(binary_img, cmap='gray')
86  plt.title('Binarized Image (Threshold = 128)')
87  plt.axis('off')
88  plt.tight_layout()
89  plt.show()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



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

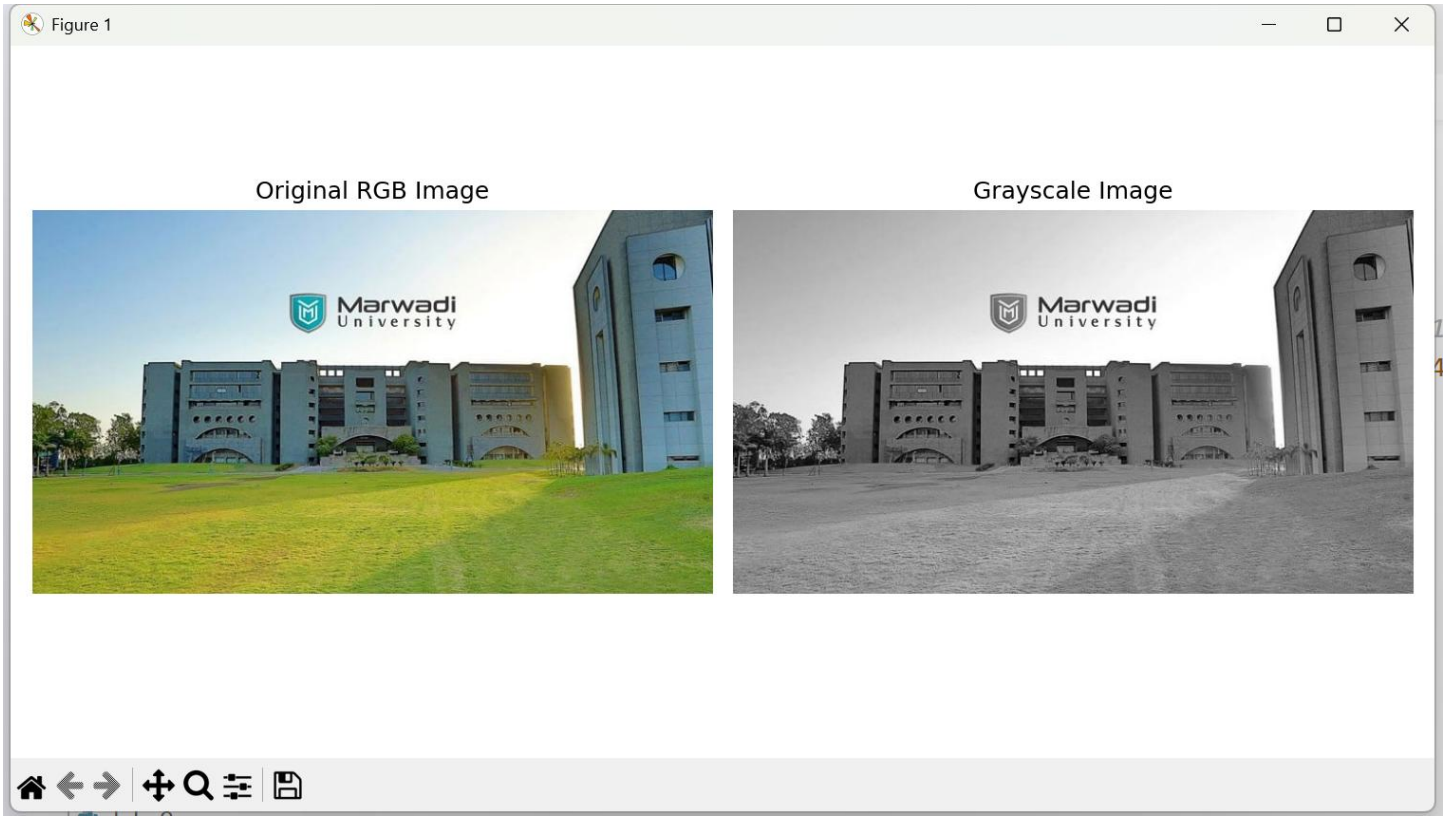
```
import numpy as np
from PIL import Image, ImageOps
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
# Grayscale conversion formula: Y = 0.299*R + 0.587*G + 0.114*B
gray_img = np.dot (img_array[..., :3], [0.299, 0.587, 0.114])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

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

plt.axis('off')
# 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

```
import numpy as np
from PIL import Image, ImageOps
import matplotlib.pyplot as plt
img = Image.open(r'E:\SEM 3\PWP\Class Tutorials\MU.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')
# 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()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



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:

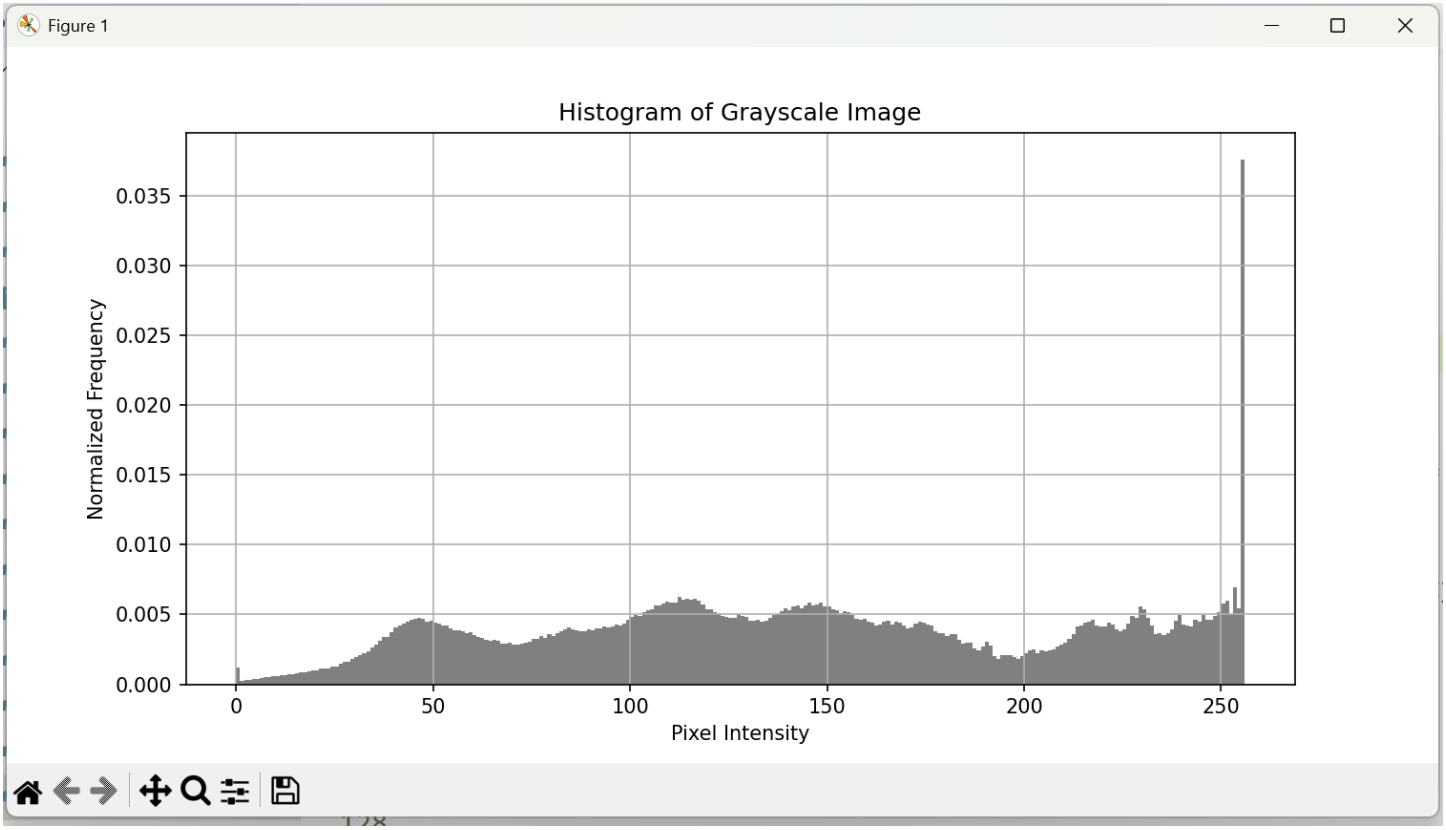
```
import numpy as np
from PIL import Image, ImageOps
import matplotlib.pyplot as plt
img = Image.open(r'C:\Users\Mitesh\OneDrive\Desktop\images.jpg')
img_array = np.array(img)
# Compute the histogram of the image
hist, bins = np.histogram(img_array.flatten(), bins=256, range= (0, 256))
# Plot the histogram
plt.figure(figsize=(10, 5))
plt.hist(img_array.flatten(), bins=256, range= (0, 256), density=True, color='gray')
plt.xlabel('Pixel Intensity')
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

plt.ylabel('Normalized Frequency')
plt.title('Histogram of Grayscale Image')
plt.grid(True)
plt.show()
Output

```
113  import numpy as np
114  from PIL import Image, ImageOps
115  import matplotlib.pyplot as plt
116  img = Image.open(r'E:\SEM 3\PWP\Class Tutorials\MU.jpg')
117  img_array = np.array(img)
118  # Compute the histogram of the image
119  hist, bins = np.histogram(img_array.flatten(), bins=256, range= (0, 256))
120  # Plot the histogram
121  plt.figure(figsize=(10, 5))
122  plt.hist(img_array.flatten(), bins=256, range= (0, 256), density=True, color='gray')
123  plt.xlabel('Pixel Intensity')
124  plt.ylabel('Normalized Frequency')
125  plt.title('Histogram of Grayscale Image')
126  plt.grid(True)
127  plt.show()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



Post Lab Exercise:

a. Write a Python program to display details of an image (dimension of an image, shape of an image, min pixel value at channel B).

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

```python
#Question 1
import numpy as np
from PIL import Image

img = Image.open(r"E:\SEM 3\PWP\Class Tutorials\MU.jpg")
img_array = np.array(img)
height, width, channels = img_array.shape
min_blue = img_array[:, :, 2].min()
print(f"Image Dimensions: {img_array.shape}")
print(f"Height: {height}, Width: {width}, Channels: {channels}")
print(f"Minimum pixel value in Blue channel: {min_blue}")
```

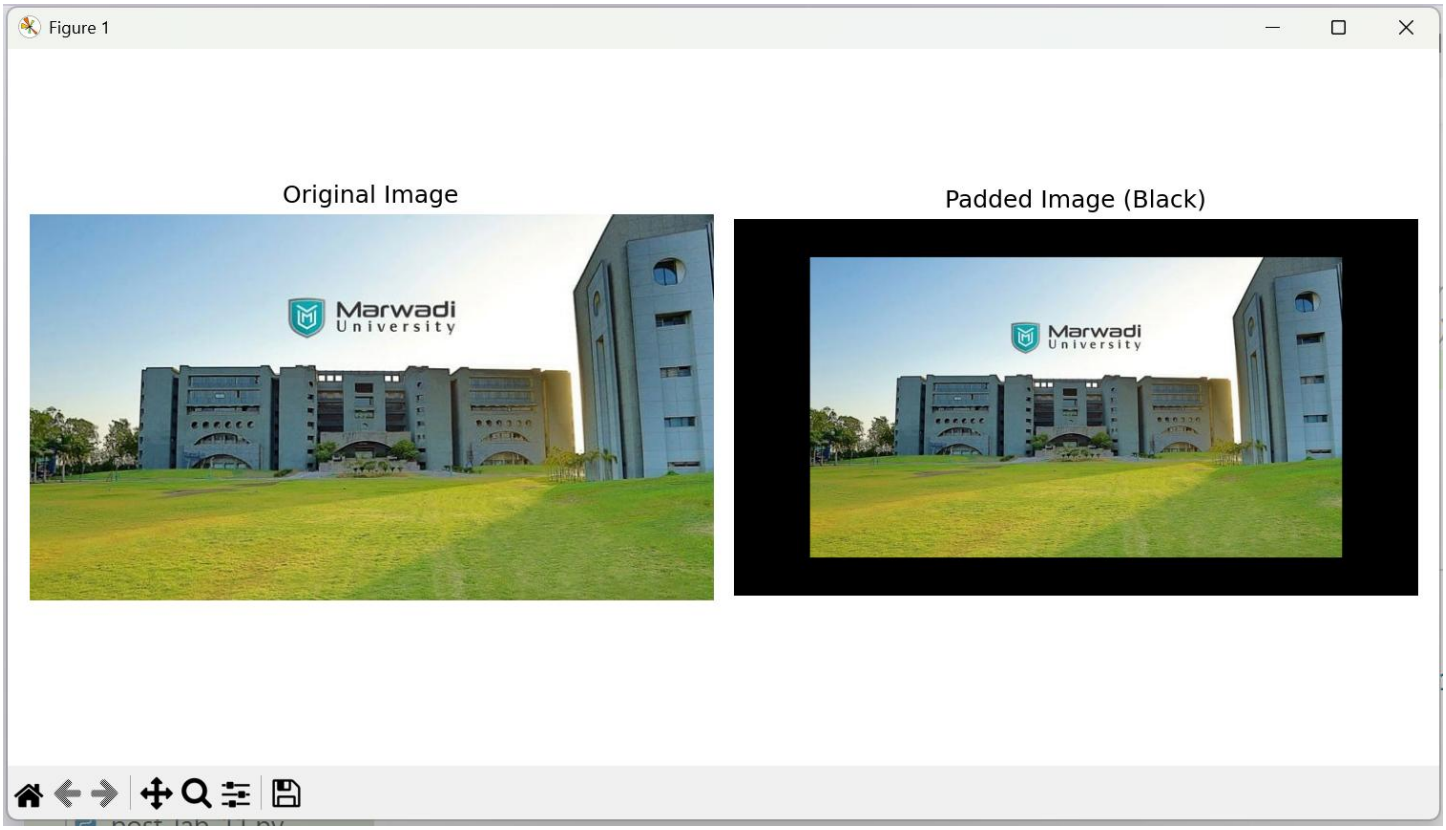
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Image Dimensions: (394, 700, 3)
Height: 394, Width: 700, Channels: 3
Minimum pixel value in Blue channel: 0
PS E:\SEM 3\PWP>
```

b. Write a Python program to padding black spaces

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

```python
#Question 2
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

img = Image.open(r"E:\SEM 3\PWP\Class Tutorials\MU.jpg")
img_array = np.array(img)
padded_img = np.pad(img_array, ((50, 50), (100, 100), (0, 0)), mode='constant', constant_values=0)
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(padded_img)
plt.title("Padded Image (Black)")
plt.axis("off")
plt.tight_layout()
plt.show()
```

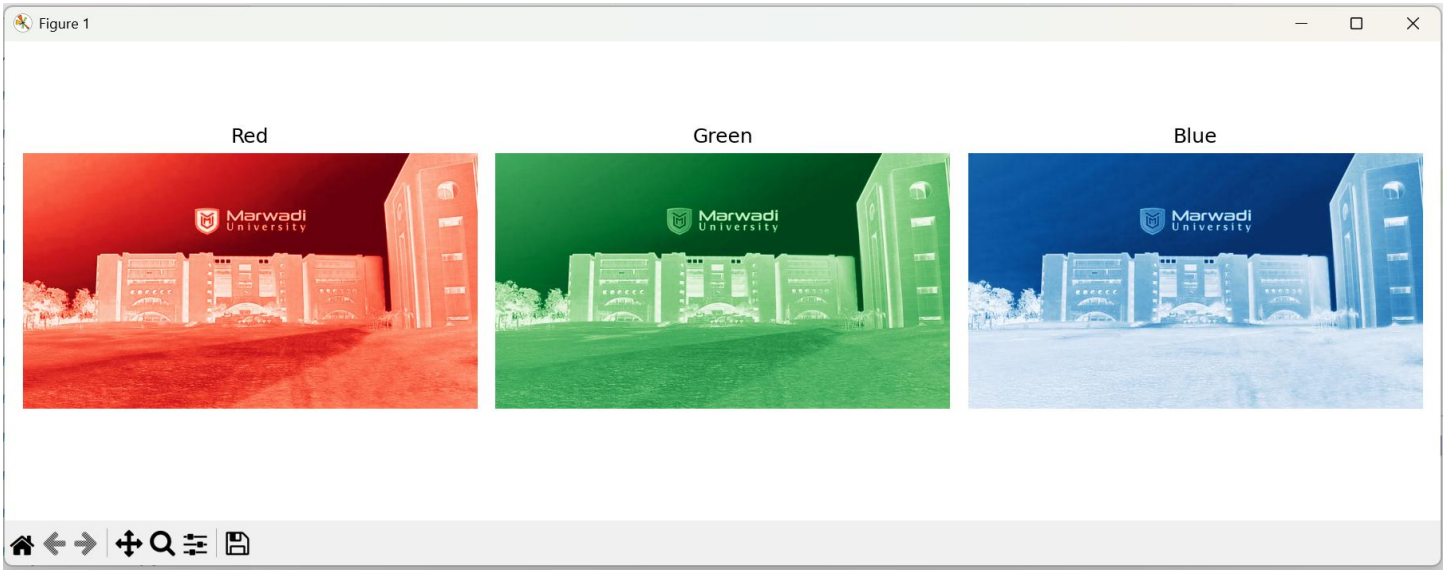


Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**

c. Write a Python program to visualize RGB channels

```
34  #Question 3
35  import numpy as np
36  from PIL import Image
37  import matplotlib.pyplot as plt
38
39  img = Image.open(r"E:\SEM 3\PWP\Class Tutorials\MU.jpg")
40  img_array = np.array(img)
41  R = img_array[:, :, 0]
42  G = img_array[:, :, 1]
43  B = img_array[:, :, 2]
44
45  plt.figure(figsize=(12,4))
46  plt.subplot(1,3,1)
47  plt.imshow(R, cmap="Reds")
48  plt.title("Red")
49  plt.axis("off")
50  plt.subplot(1,3,2)
51  plt.imshow(G, cmap="Greens")
52  plt.title("Green")
53  plt.axis("off")
54  plt.subplot(1,3,3)
55  plt.imshow(B, cmap="Blues")
56  plt.title("Blue")
57  plt.axis("off")
58  plt.tight_layout()
59  plt.show()
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Image Processing with Numpy	
Experiment No: 11	**Date:**	**Enrollment No: 92400133055**



More Practice

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