Worksheet -1

Worksheet - 1

Samprada SHrestha

Installation and Mounting Google Drive

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[ ] # !pip install pillow
    from PIL import Image
    import numpy as np
    import matplotlib.pyplot as plt
```

Exercise - 1:

1. Read and Display Image

```
[ ] colored_image = Image.open("/content/drive/MyDrive/AI and ML Workshop/Week-1/lenna_image.png")
```

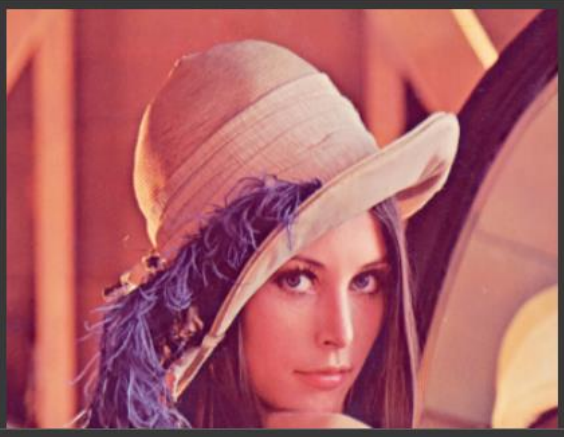
1. Read and Display Image

```
[ ]  colored_image = Image.open("/content/drive/MyDrive/AI and ML Workshop/Week-1/lenna_image.png")
```

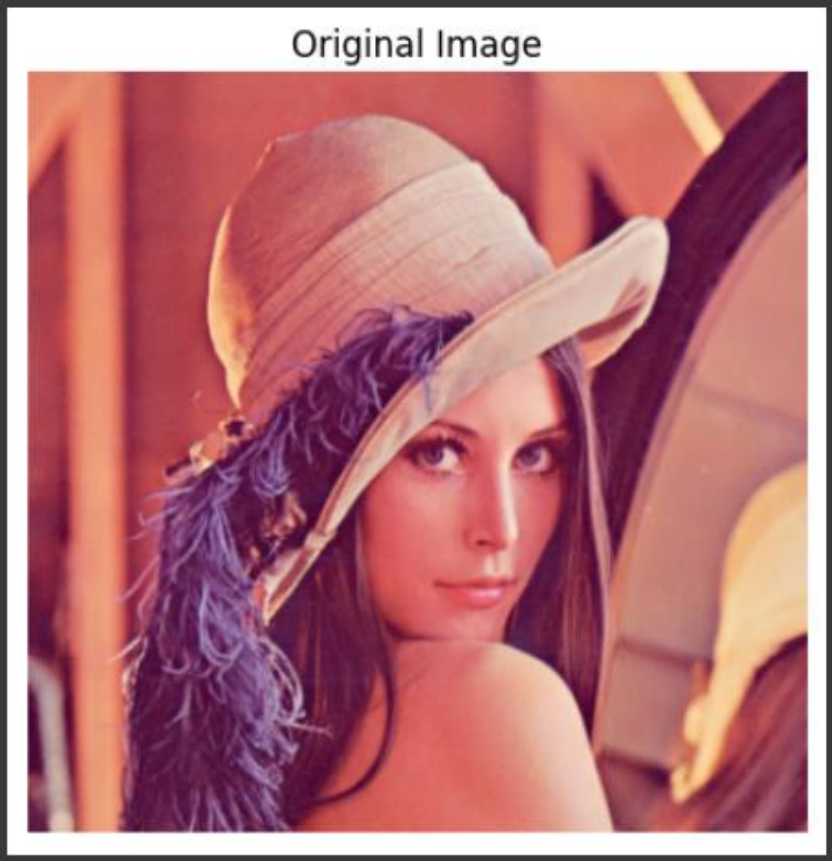
```
[ ]  print("Format: ", colored_image.format)
     print("Mode: ", colored_image.mode)
     print("Size: ", colored_image.size)
```

```
Format:  PNG
Mode:  RGBA
Size:  (366, 357)
```

```
     colored_image = colored_image.convert("RGB")
     display(colored_image)
```



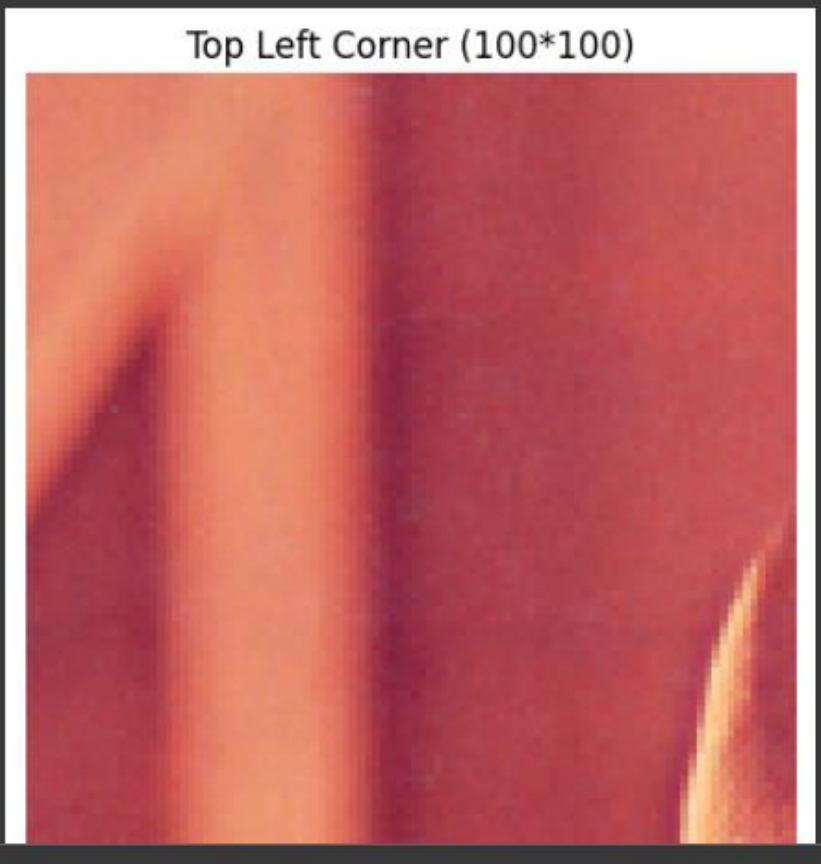
```
image_array = np.array(colored_image)
plt.imshow(image_array)
plt.axis("off")
plt.title("Original Image")
plt.show()
```



2. Display top corner of 100*100 pixels

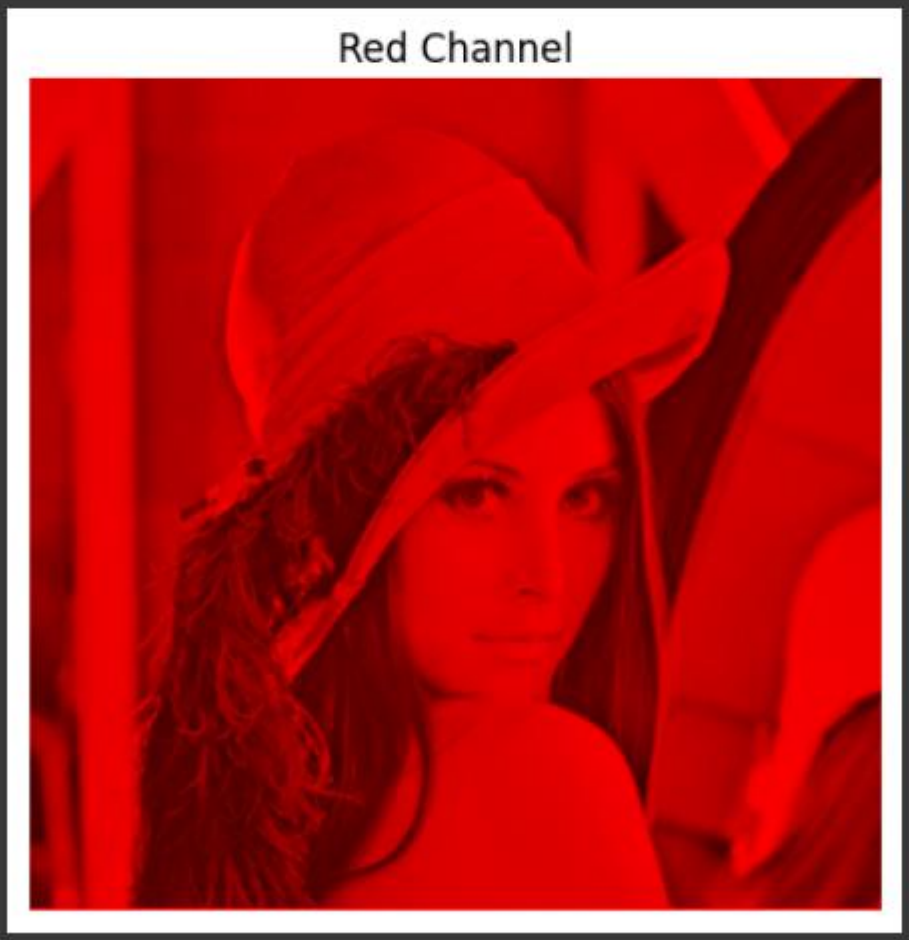
2. Display top corner of 100*100 pixels

```
image_array = np.array(colored_image)

top_left = image_array[:100, :100]

plt.axis("off")
plt.title("Top Left Corner (100*100)")
plt.imshow(top_left)
plt.show()
```

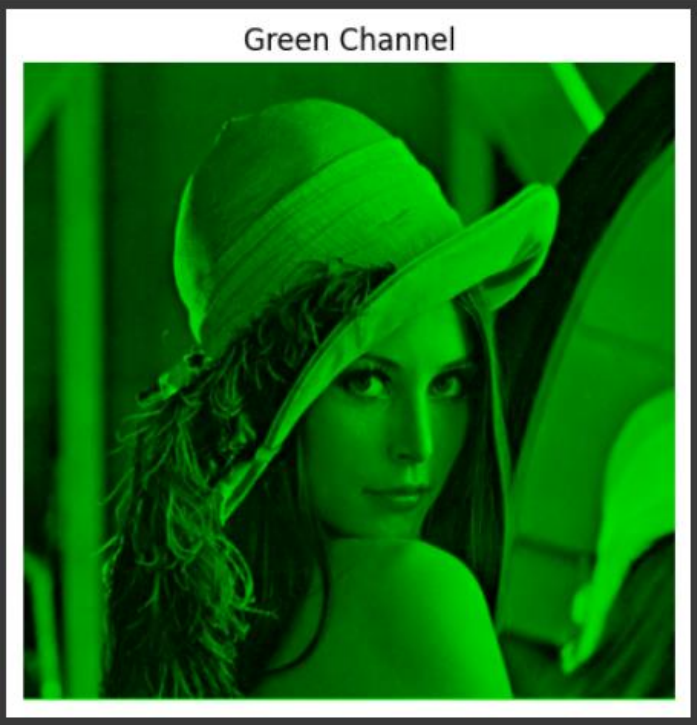


3. Displaying Color Channels

```
red_channel = image_array.copy()
red_channel[:, :, 1] = 0
red_channel[:, :, 2] = 0
plt.axis("off")
plt.title("Red Channel")
plt.imshow(red_channel)
plt.show()
```



```
green_channel = image_array.copy()
green_channel[:, :, 0] = 0
green_channel[:, :, 2] = 0
plt.axis("off")
plt.title("Green Channel")
plt.imshow(green_channel)
plt.show()
```



```
blue_channel = image_array.copy()
blue_channel[:, :, 1] = 0
blue_channel[:, :, 0] = 0
plt.axis("off")
plt.title("Blue Channel")
plt.imshow(blue_channel)
plt.show()
```

4. Top 100 * 100 modification

```
image_modified = image_array.copy()
image_modified[:100, :100] = 210

plt.axis("off")
plt.title("Top 100 * 100 to 210 - Modified Image")
plt.imshow(image_modified)
plt.show()
```

Exercise - 2:

1. Load and Display Grayscale Image

```
[ ] image_grayscale = Image.open("/content/drive/MyDrive/AI and ML Workshop/Week-1/camera_man.jpg").convert("L")
```

```
image_array = np.array(image_grayscale)

plt.title("Original Grayscale Image")
plt.axis("off")
plt.imshow(image_array, cmap="gray")
plt.show()
```

2. Extract Middle 150 pixels of Image

```
width, height = image_array.shape

col_start = (width // 2)
row_start = (height // 2 )
mid_section = image_array[row_start:row_start+150, col_start:col_start+150]


plt.title("Middle 150 * 150 pixels")
plt.axis("off")
plt.imshow(mid_section, cmap="gray")
plt.show()
```

3.Apply a simple threshold to Image

```
binary_image = np.zeros_like(image_array, dtype=np.uint8)

height, width = image_array.shape

for i in range(height):
    for j in range(width):
        if image_array[i, j] < 100:
            binary_image[i, j] = 0
        else:
            binary_image[i, j] = 255

plt.title("Thresholded Image")
plt.axis("off")
plt.imshow(binary_image, cmap="gray")
plt.show()
```

4. Rotate Image 90 deg

```
rotate_image = image_grayscale.rotate(-90, expand = True)

plt.title("Rotated Image Clockwise 90 deg")
plt.axis("off")
plt.imshow(rotate_image, cmap="gray")
plt.show()
```

5. Convert Grayscale to RGB image

5. Convert Grayscale to RGB image

```
image_colored = Image.merge("RGB", (image_grayscale, image_grayscale, image_grayscale))

plt.title("Grayscale to RGB")
plt.axis("off")
plt.imshow(image_colored, cmap="gray")
plt.show()
```

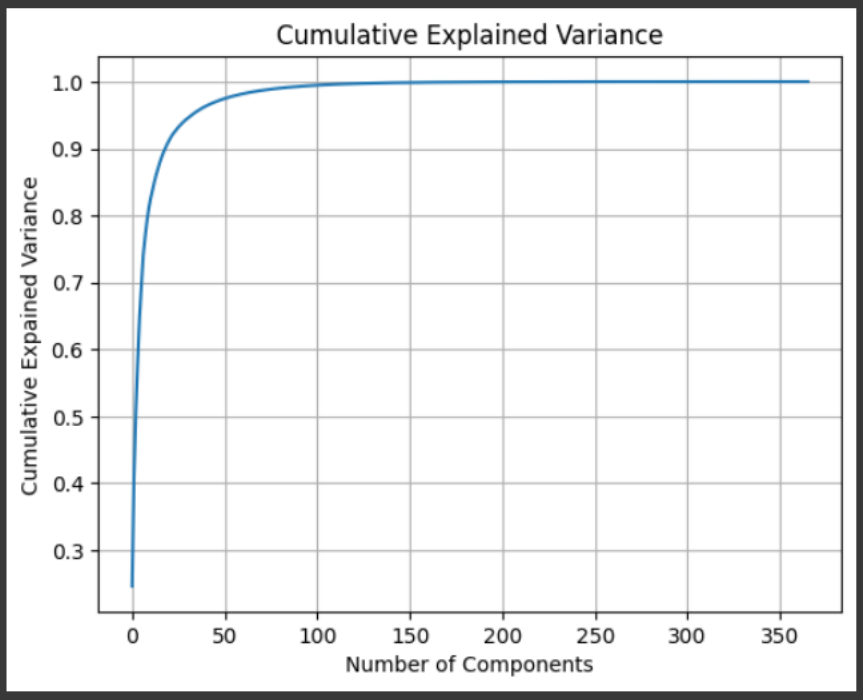
1. Load and Prepare Data

```
[ ]  image = Image.open("/content/drive/MyDrive/AI and ML Workshop/Week-1/lenna_image.png").convert("L")
```

```
image_array = np.array(image)
print(image_array.shape)

height, width = image_array.shape

data = image_array.copy()

plt.title("Original Image")
plt.axis("off")
plt.imshow(image_array, cmap='gray')
plt.show()
```

```
(357, 366)
```

```
explained_variance_ratio = eigenvalues / np.sum(eigenvalues)
plt.plot(np.cumsum(explained_variance_ratio))
plt.title("Cumulative Explained Variance")
plt.xlabel("Number of Components")
plt.ylabel("Cumulative Expained Variance")
plt.grid(True)
plt.show()
```



```
[ ]  decompressed_data1 = np.dot(compressed_data1, components1.T) + mean
     decompressed_data2 = np.dot(compressed_data2, components2.T) + mean
     decompressed_data3 = np.dot(compressed_data3, components3.T) + mean
     decompressed_data4 = np.dot(compressed_data4, components4.T) + mean
     decompressed_data5 = np.dot(compressed_data5, components5.T) + mean
```

```
plt.figure(figsize=(15, 8))

plt.subplot(2, 3, 1)
plt.imshow(image_array, cmap="gray")
plt.title("Original Image")
plt.axis("off")

plt.subplot(2, 3, 2)
plt.imshow(decompressed_data1, cmap="gray")
plt.title("10 Components Image")
plt.axis("off")

plt.subplot(2, 3, 3)
plt.imshow(decompressed_data2, cmap="gray")
plt.title("20 Components Image")
plt.axis("off")

plt.subplot(2, 3, 4)
plt.imshow(decompressed_data3, cmap="gray")
plt.title("50 Components Image")
plt.axis("off")

plt.subplot(2, 3, 5)
plt.imshow(decompressed_data4, cmap="gray")
plt.title("100 Components Image")
plt.axis("off")
```