PRIYANSHU MANOHAR PATIL,BRANCH-EXTC, ROLL.NO-55

```
#1. Pillow is an image manipulation and processing library
#You can use pillow to crop, resize images and to do basic filtering.
#For advanced tasks that require computer vision or machine elarning we have other package
#such as openCV, scikit image and scikit learn.
# to install pillow, pip install Pillow
# to import the package you need to use import PIL
from PIL import Image
import numpy as np   #Use numpy to convert images to arrays
# Read image
img = Image.open("/content/priyanshu1.jpg") #Not a numpy array
print(type(img))
```

```
<class 'PIL.JpegImagePlugin.JpegImageFile'>
```

```
# prints format of image
print(img.format)
```

```
# prints mode of image
print(img.mode)

#PIL is not by default numpy array but can convert PIL image to numpy array.
img1 = np.asarray(img)
print(type(img1))
```

```
RGB
<class 'numpy.ndarray'>
```

```
img1
```

```
array([[[203, 210, 216],
        [203, 210, 216],
        [203, 210, 216],
        ...,
        [187, 192, 198],
        [187, 192, 198],
        [187, 192, 198]],

       [[203, 210, 216],
        [203, 210, 216],
        [203, 210, 216],
        ...,
        [187, 192, 198],
        [187, 192, 198],
        [187, 192, 198]],

       [[203, 210, 216],
        [203, 210, 216],
        [203, 210, 216],
        ...,
        [187, 192, 198],
```

```
       [187, 192, 198],
       [187, 192, 198]],

      ...,

      [[ 46,  35,  17],
       [ 48,  37,  19],
       [ 52,  41,  23],
       ...,
       [157, 147, 148],
       [155, 145, 146],
       [153, 142, 146]],

      [[ 48,  37,  19],
       [ 49,  38,  20],
       [ 49,  38,  20],
       ...,
       [158, 147, 151],
       [155, 144, 148],
       [154, 143, 147]],

      [[ 52,  41,  23],
       [ 50,  39,  21],
       [ 47,  35,  19],
       ...,
       [160, 149, 153],
       [157, 146, 150],
       [155, 144, 148]]], dtype=uint8)
```

```
img1.shape
```

```
(6240, 4160, 3)
```

```
print(img1.shape)
```

```
(6240, 4160, 3)
```

```
img = Image.open("/content/priyanshu1.jpg") #Not a numpy array
print(type(img))
```

```
<class 'PIL.JpegImagePlugin.JpegImageFile'>
```

```
img2 = Image.open("/content/priyanshu2.jpg") #Not a numpy array
print(type(img2))
```

```
<class 'PIL.JpegImagePlugin.JpegImageFile'>
```

```
img2.shape
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-43-90f82793b7d9> in <module>
```

```
######### 2. Using Matplotlib #########
#Matplotlib is a plotting library for the Python programming language
#Pyplot is a Matplotlib module which provides a MATLAB-like interface
#Pyplot is commonly used not just to generate plots and graphs but also to visualize image
#because visualizing images is nothing but plotting data in 2D.
# to install matplotlib, pip install matplotlib
# to import the package you need to use import matplotlib
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
img = mpimg.imread("/content/priyanshu1.jpg")  #this is a numpy array
print(type(img))
#print(img)
print(img.shape)
plt.imshow(img)
plt.colorbar()   #Puts a color bar next to the image.
```

```
<class 'numpy.ndarray'>
(1200, 1920, 3)
<matplotlib.colorbar.Colorbar at 0x7fd34a65ce90>
```


```
######Using scikit image ############
# to install matplotlib, pip install scikit-image
# to import the package you need to use import skimage
#scikit image is an image processing library that includes alforithms for
#segmentation, geometric transformation, color space manipulation, analysis, filtering,
#feature detection, and more.
#A very good package for traditional machine learning, using Random forest or SVM


from skimage import io, img_as_float, img_as_ubyte
import numpy as np
import matplotlib.pyplot as plt


image = img_as_float(io.imread("/content/priyanshu1.jpg"))

image2 = io.imread("/content/priyanshu2.jpg").astype(np.float)
#avoid using astype as it violates assumptions about dtype range.
#for example float should range from 0 to 1 (or -1 to 1) but if you use
```

```
#for example float should range from 0 to 1 (or -1 to 1) but if you use
#astype to convert to float, the values do not lie between 0 and 1.
#print(image.shape)
#plt.imshow(img)

print(image2)

#print(image2)
#image8byte = img_as_ubyte(image)
#print(image8byte)

#End of Skimage
```

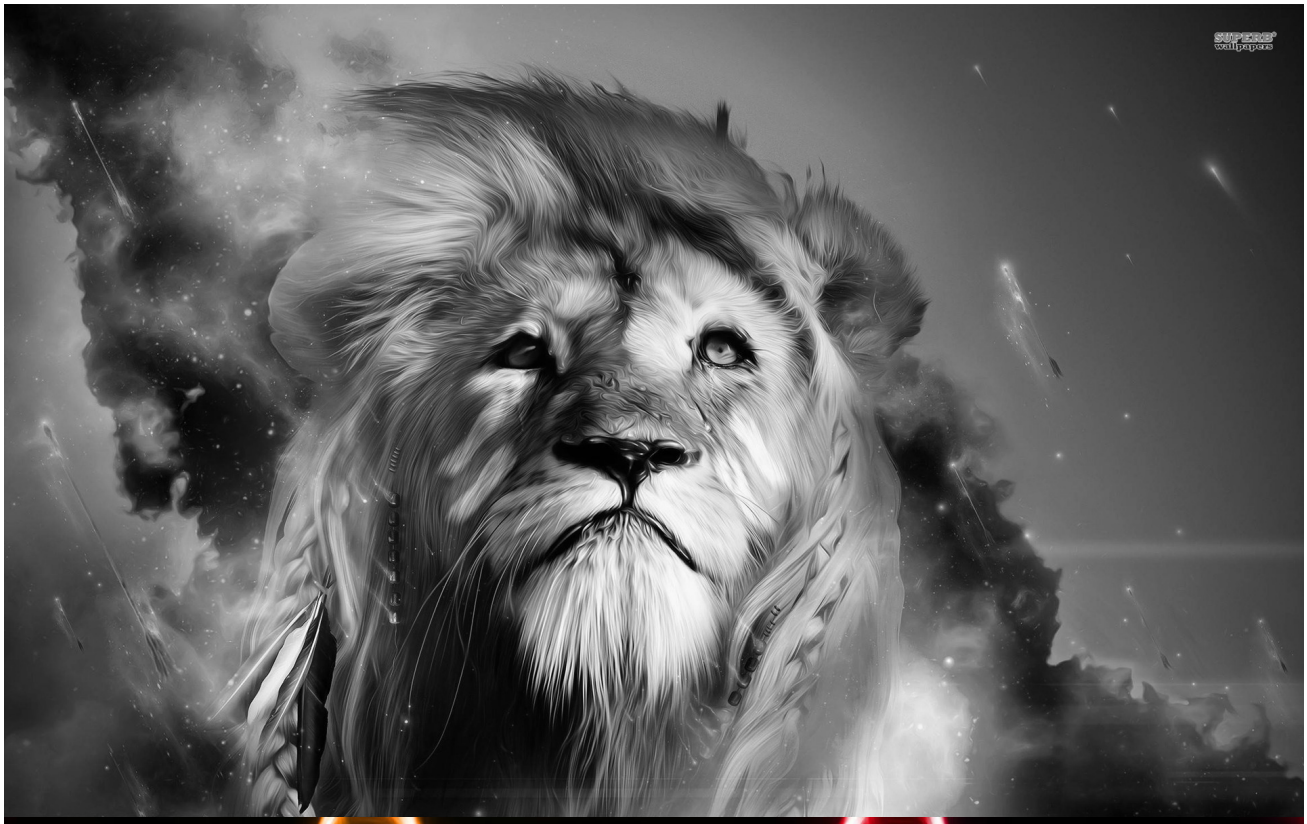
```
[[[27. 71. 20.]
  [34. 78. 27.]
  [19. 63. 12.]
  ...
  [18. 32. 19.]
  [15. 29. 16.]
  [13. 27. 14.]]

 [[32. 74. 24.]
  [41. 83. 33.]
  [33. 77. 26.]
  ...
  [18. 32. 19.]
  [16. 30. 17.]
  [15. 29. 16.]]

 [[30. 72. 22.]
  [35. 77. 27.]
  [39. 81. 31.]
  ...
  [17. 31. 18.]
  [17. 31. 18.]
  [17. 31. 18.]]

 ...

 [[23. 39. 26.]
  [29. 45. 32.]
  [37. 53. 40.]
  ...
  [13. 15. 10.]
  [14. 16. 11.]
  [14. 16. 11.]]

 [[28. 44. 31.]
  [32. 48. 35.]
  [39. 55. 42.]
  ...
  [14. 16. 11.]
  [14. 16. 11.]
  [14. 16. 11.]]

 [[30. 46. 33.]
  [34. 50. 37.]
  [40. 56. 43.]
  ...
```

```
  [14. 16. 11.]
  [14. 16. 11.]
  [15. 17. 12.]]]
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:17: DeprecationWarning:
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/re
```

```
######### Using openCV #########
#to install open CV : pip install opencv-python
#to import the package you need to use import cv2
#openCV is a library of programming functions mainly aimed at computer vision.
#Very good for images and videos, especially real time videos.
#It is used extensively for facial recognition, object recognition, motion tracking,
#optical character recognition, segmentation, and even for artificial neural netwroks.
'''You can import images in color, grey scale or unchanged usingindividual commands
cv2.IMREAD_COLOR : Loads a color image. Any transparency of image will be neglected. It is
cv2.IMREAD_GRAYSCALE : Loads image in grayscale mode
cv2.IMREAD_UNCHANGED : Loads image as such including alpha channel
Instead of these three flags, you can simply pass integers 1, 0 or -1 respectively.
'''
import cv2  # library for Open Source Computer vision library

from google.colab.patches import cv2_imshow

grey_img = cv2.imread("/content/priyanshu1.jpg", 0)
color_img = cv2.imread("/content/priyanshu2.jpg", 1)

#images opened using cv2 are numpy arrays
print(type(grey_img))
print(type(color_img))

# Use the function cv2imshow() to display an image in a window.
# First argument is the window name which is a string. second argument is our image.
```

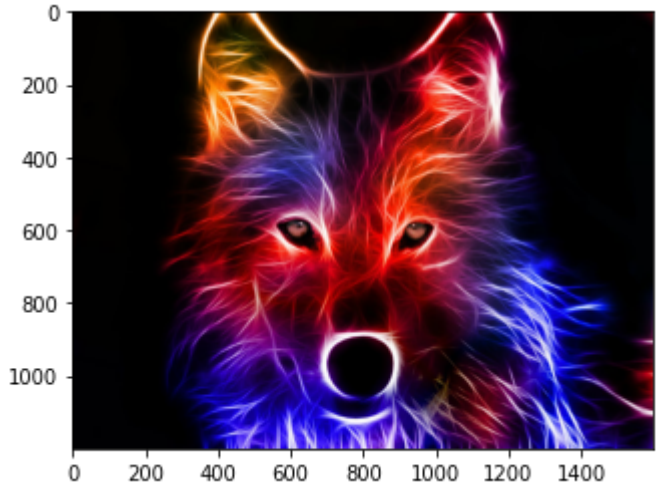
```
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
```

```
cv2_imshow(grey_img)
cv2_imshow(color_img)
```



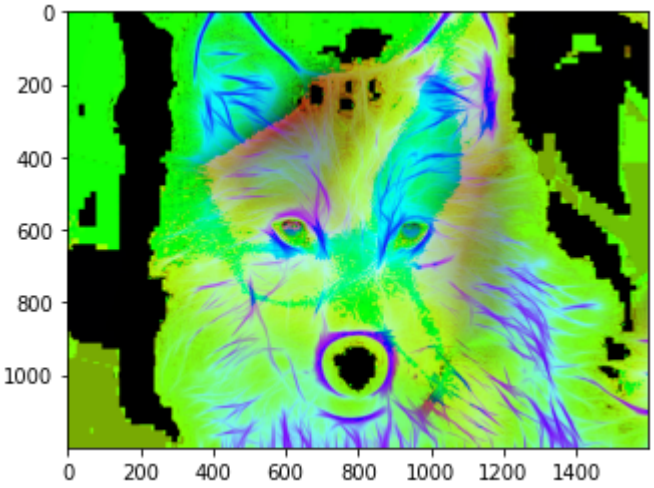
```
#OpenCV represents RGB images as multi-dimensional NumPy arrays, but as BGR.
#we can convert the images from BGR to RGB
plt.imshow(cv2.cvtColor(color_img, cv2.COLOR_BGR2RGB))
```

```
<matplotlib.image.AxesImage at 0x7fd33c501f10>
```



```
#We can also change color spaces from RGB to HSV..
plt.imshow(cv2.cvtColor(color_img, cv2.COLOR_BGR2HSV))
```

```
<matplotlib.image.AxesImage at 0x7fd33c0d29d0>
```




```
#Expt=1
```

```
import numpy as np       # library used for working with arrays
import matplotlib.pylab as plt  # library used for ploting, graph
from io import BytesIO
import cv2  # library for Open Source Computer vision library
from PIL import Image    # Python Imaging Library for load, display, save etc
from google.colab.patches import cv2_imshow
from google.colab import files
uploaded = files.upload()
```

Choose Files 2 files

- **priyanshu1.jpg**(image/jpeg) - 622500 bytes, last modified: 10/17/2022 - 100% done
- **priyanshu2.jpg**(image/jpeg) - 215840 bytes, last modified: 10/17/2022 - 100% done

```
Saving priyanshu1.jpg to priyanshu1 (1).jpg
Saving priyanshu2.jpg to priyanshu2 (1).jpg
```

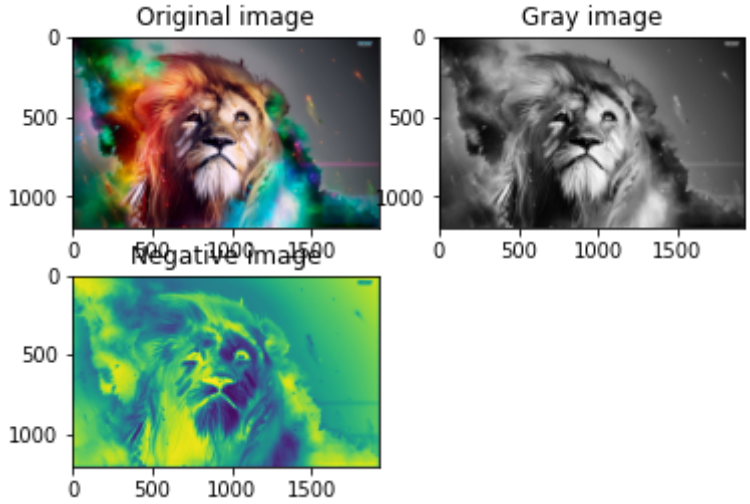
```
img1=Image.open(BytesIO(uploaded['priyanshu1.jpg']))
img2 = cv2.imread("priyanshu1.jpg")
```

```
#plt.imshow(img2)
gray_img = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
```

```
#plt.imshow(gray_img,cmap=plt.cm.gray)

### Negative of the image
[rows,cols] = gray_img.shape;
neg_img = np.zeros((rows,cols));
for r in range(0, rows-1):
  for c in range(0, cols-1):
    neg_img[r,c] = 255-gray_img[r,c]; #Calculate negative image
#plt.imshow(neg_img, cmap=plt.cm.gray)
```

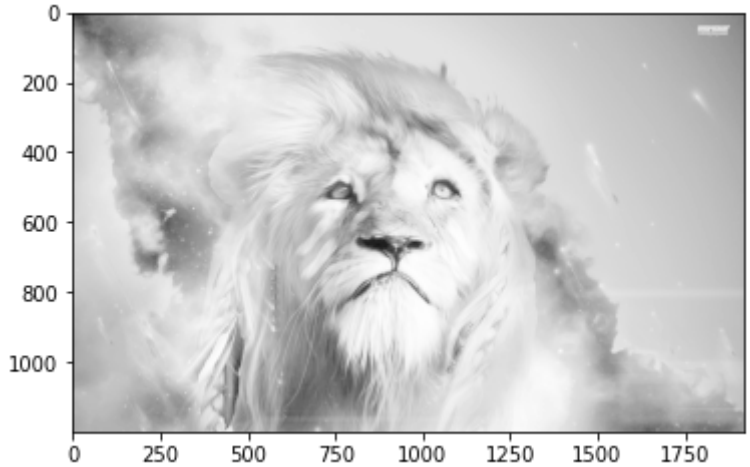
```
#plt.imshow(neg_img, cmap=plt.cm.gray)
fig = plt.figure()
ax1 = fig.add_subplot(2,2,1)
ax1.imshow(img1)
ax1.set_title('Original image')
ax2 = fig.add_subplot(2,2,2)
ax2.imshow(gray_img, cmap=plt.cm.gray)
ax2.set_title('Gray image')
ax3 = fig.add_subplot(2,2,3)
ax3.imshow(neg_img)
ax3.set_title('Negative image')
```

```
Text(0.5, 1.0, 'Negative image')
```




```
# c = 255/(log (1 + m)), where m is the maximum pixel value in the image.
#c = 0.1
c = 255/(np.log(1 + np.max(gray_img)))
log_transformed = c * np.log(1 + gray_img)
log_transformed1 = np.array(log_transformed, dtype = np.uint8)
plt.imshow(log_transformed1, cmap=plt.cm.gray)
```

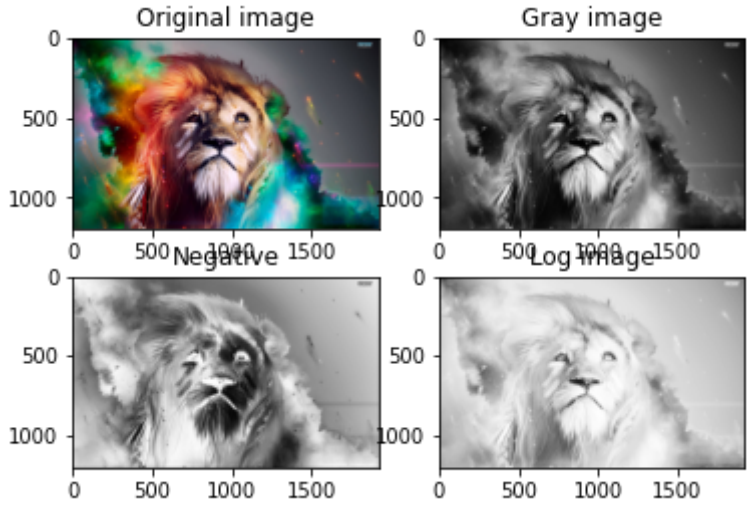
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: RuntimeWarning: divic
  after removing the cwd from sys.path.
<matplotlib.image.AxesImage at 0x7fd33c280590>
```




```
fig = plt.figure()
ax1 = fig.add_subplot(2,2,1)
ax1.imshow(img1)
```

```
ax1.set_title('Original image')
ax2 = fig.add_subplot(2,2,2)
ax2.imshow(gray_img, cmap=plt.cm.gray)
ax2.set_title('Gray image')
ax3 = fig.add_subplot(2,2,3)
ax3.imshow(neg_img, cmap=plt.cm.gray)
ax3.set_title('Negative')
ax4 = fig.add_subplot(2,2,4)
ax4.imshow(log_transformed1, cmap=plt.cm.gray)
ax4.set_title('Log image')
```

```
Text(0.5, 1.0, 'Log image')
```



✓ 9s completed at 11:07 PM