Experiment Title: 6

Student Name: Pratibha **UID:18BCS6093**

Branch:AIML-1 **Section/Group:B**

Semester:5 **Date of Performance:**

Subject Name:DIP LAB **Subject Code: CSF-336**

1. Aim/Overview of the practical:

Write the python program which help us to understand the concept od scikit images and there manipulations

2. The task to be done:

Operation on image using Scikit Module

3. Required libraries or software

```
import numpy as np
from skimage import data
from skimage import io
import matplotlib.pyplot as plt
import cv2 as cv
```

4. Algorithm/Flowchart :

5. Theme/Interests definition(For creative domains):

6. Steps for experiment/practical:

```
# ## Operation  on Image using Scikit module
# In[5]:
# Images are NumPy's arrays
import numpy as np
check = np.zeros((8, 8))
```

```
check[::2, 1::2] = 2
check[1::2, ::2] = 2
import matplotlib.pyplot as plt
plt.imshow(check, cmap='gray', interpolation='nearest')
# cmap() method which returns a matplotlib color map with n colors.
# Interp
# In[6]:
# images using skikit-image
# importing data from skimage
from skimage import data
camera = data.camera()
# An image with 512 rows
# and 512 columns
type(camera)
print(camera.shape)
print(camera.dtype)
# ## Reading and Display of image using Scikit
# In[7]:
# Uploading and Viewing an Image
from skimage import io
img = io.imread('aa.jpg')
io.imshow(img)

# In[4]:
## get shape of image
img.shape
# In[10]:
#Gettingixel Values
from skimage import io
import pandas as pd
#Pandas is used to read, write, and process various file formats.
img = io.imread('aa.jpg')
df = pd.DataFrame(img.flatten())
```

```
# flatten function is used to convert the three dimensions of an RGB image to a single dimension

filepath = 'pixel_values.xlsx'

# DataFrame function converts a one-dimensional array into an Excel-like format, with rows and columns

df.to_excel(filepath, index=True)

# to_excel save that image in an excel file.

print(df)

#print(filepath)

# ## Converion of image into HSV and vice-versa

# In[13]:

from skimage import io

from skimage import data

from pylab import *

from skimage.color import rgb2hsv

#Read image

img1 = io.imread('aa.jpg')

#Convert to HSV

img_hsv = rgb2hsv(img1)

#Convert back to RGB

img_rgb = rgb2hsv(img_hsv)

#Show both figures

figure(0)

io.imshow(img_hsv)

figure(1)

io.imshow(img_rgb)

# ## RGB to XYZ and Vice Versa

# In[19]:

#Import libraries

from skimage import io

from skimage import color

from skimage import data

#Read image

img1 = io.imread('aa.jpg')
```

```
#Convert to XYZ
img_xyz = color.rgb2xyz(img)
#Show  figures
figure(0)
io.imshow(img_xyz)
# In[20]:
#Convert back to RGB
img_rgb = color.xyz2rgb(img_xyz)
# Show figure
figure(1)
io.imshow(img_rgb)
# ## Operation on exposure
# In[23]:
from skimage import exposure
from skimage import io
from pylab import *
img1 = io.imread('aa.jpg')
## increase the exposure
gamma_corrected1 = exposure.adjust_gamma(img, 0.5)
figure(0)
io.imshow(gamma_corrected1)
# In[24]:

## decrease the exposure
gamma_corrected2 = exposure.adjust_gamma(img, 3)
figure(1)
io.imshow(gamma_corrected2)
# ## Operations like rotation , shifting et
# In[25]:
# Rotating,
from skimage import io
from skimage.transform import rotate
img1 = io.imread('aa.jpg')
```

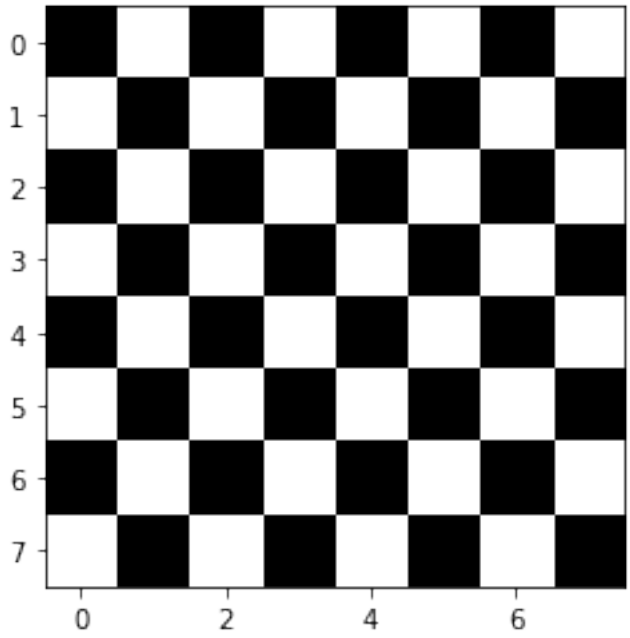
```
img_rot = rotate(img, 30)
io.imshow(img_rot)
# In[26]
from skimage import io
from skimage.transform import resize
img1 = io.imread('aa.jpg')
img_res = resize(img, (100,100))
io.imshow(img_res)
io.imsave("pp.png", img_res)
```

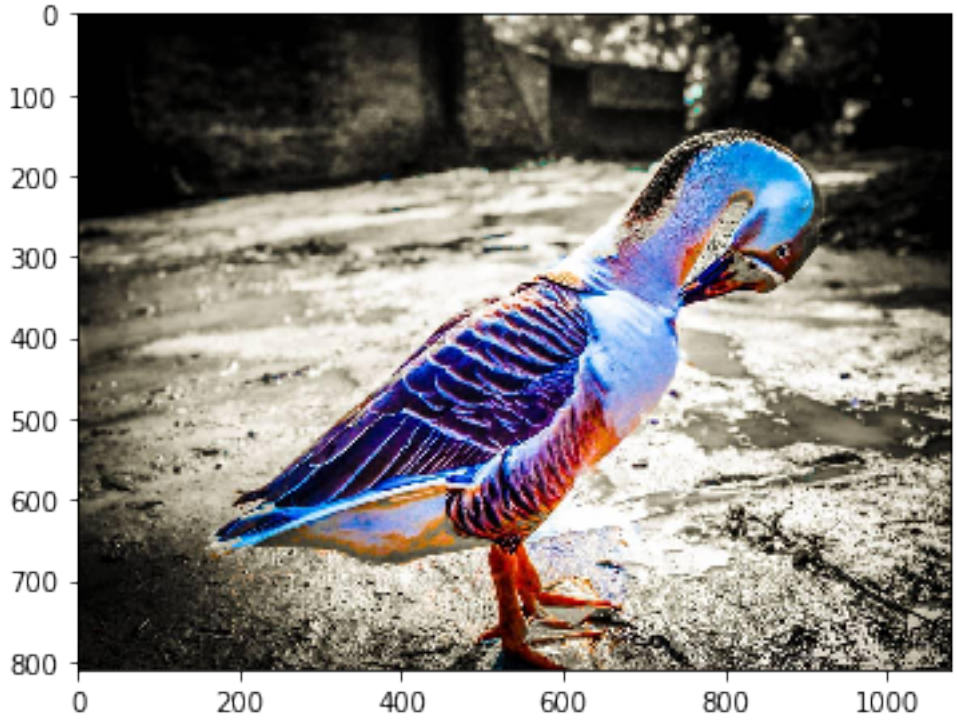
7. Observations/Discussions(For applied/experimental sciences/materials based labs):

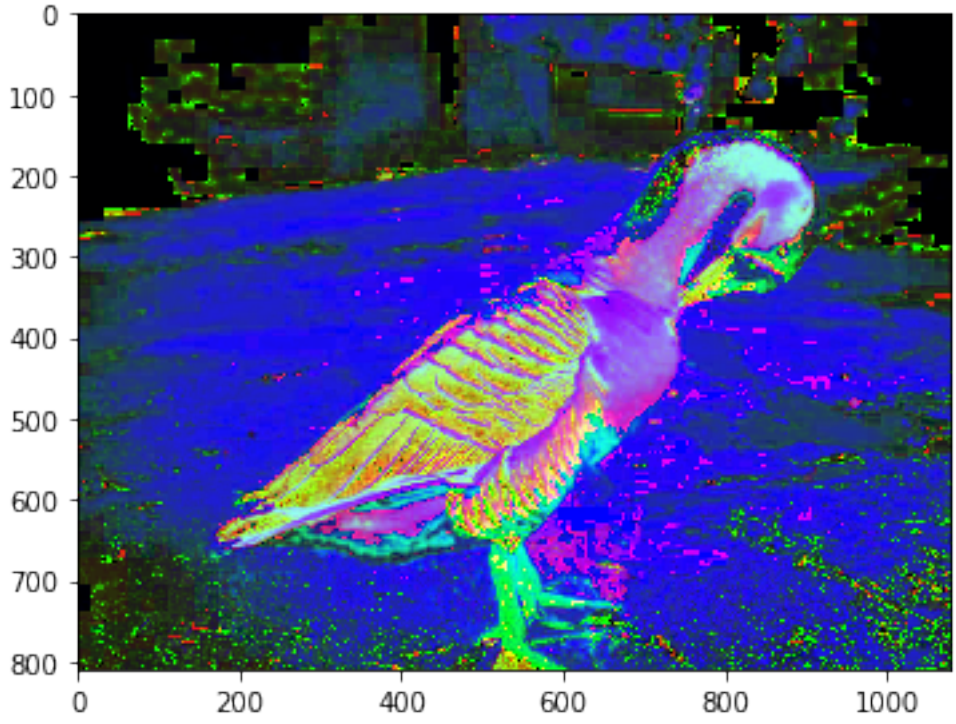
8. Percentage error (if any or applicable):

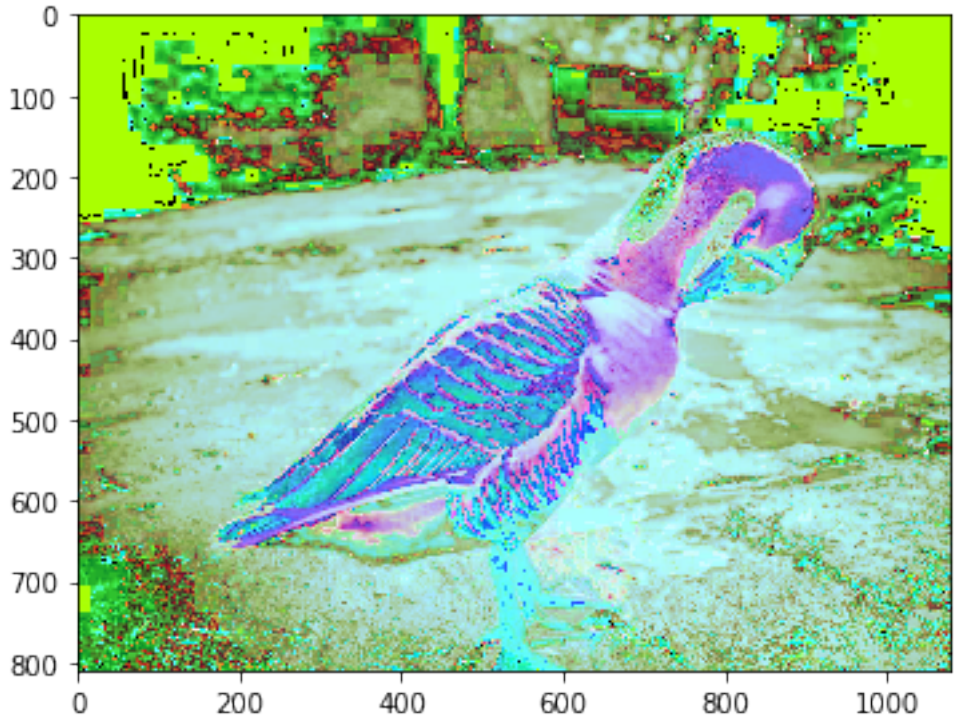
9. The command that we have learned today in the program :

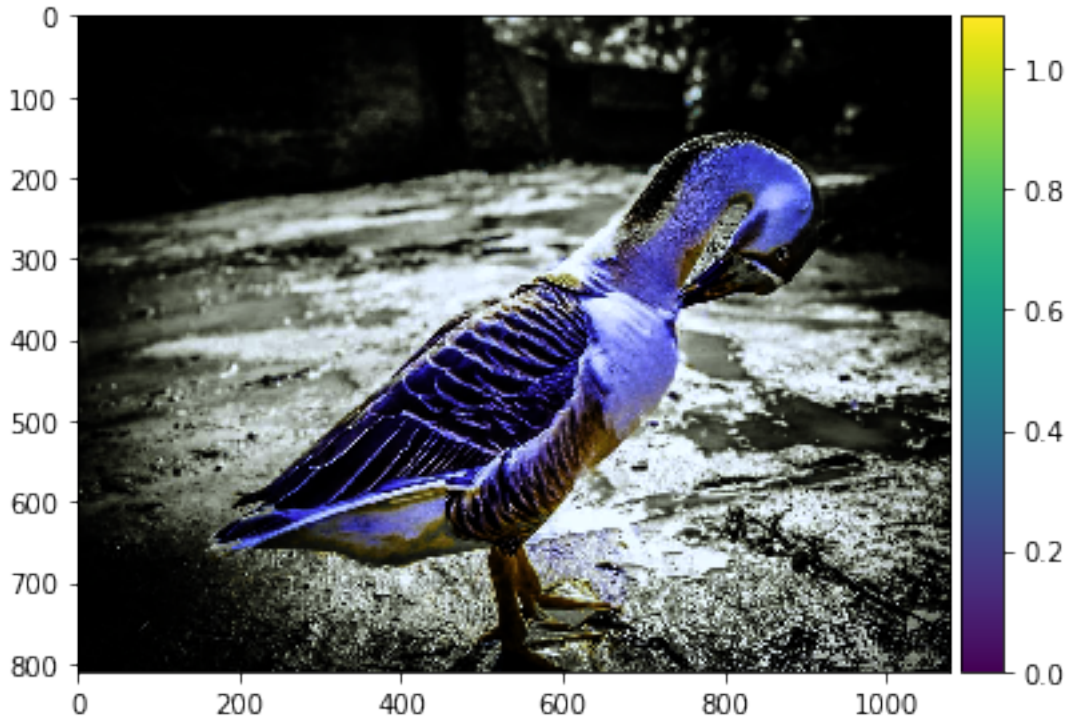
10. Result/Output/Writing Summary of the concept behind the experiment:

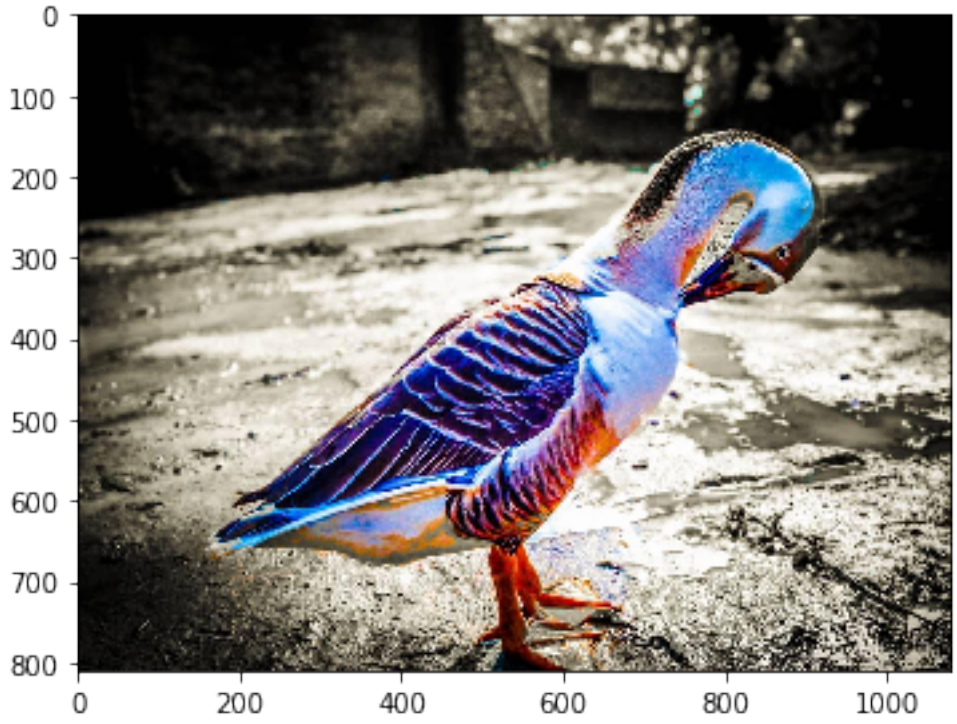

0
1
2
3
4
5
6
7
0
2
4
6

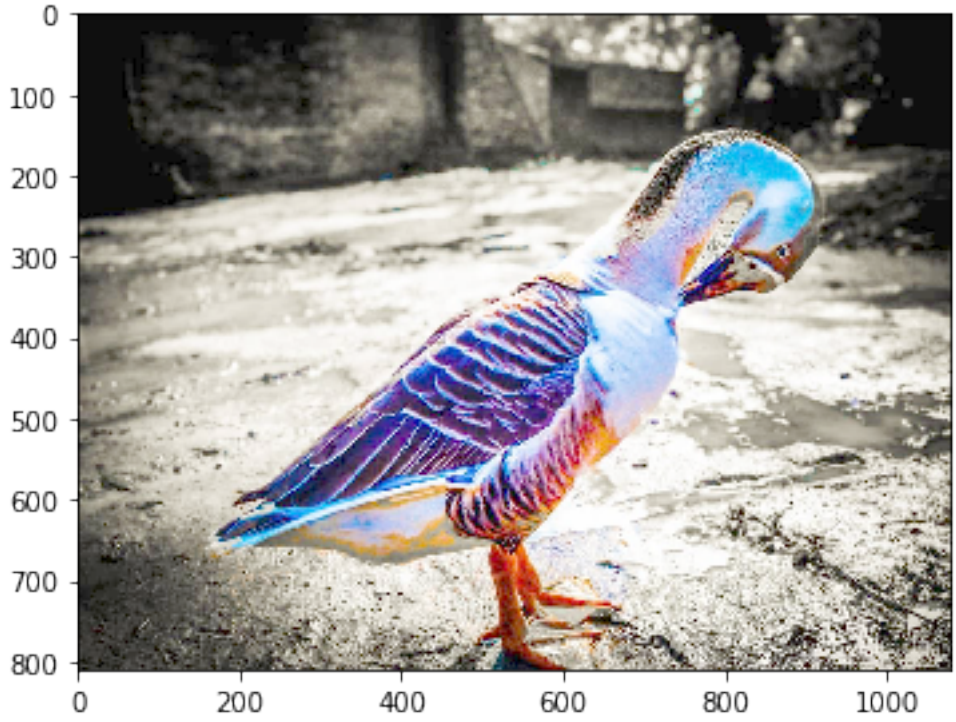

0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000

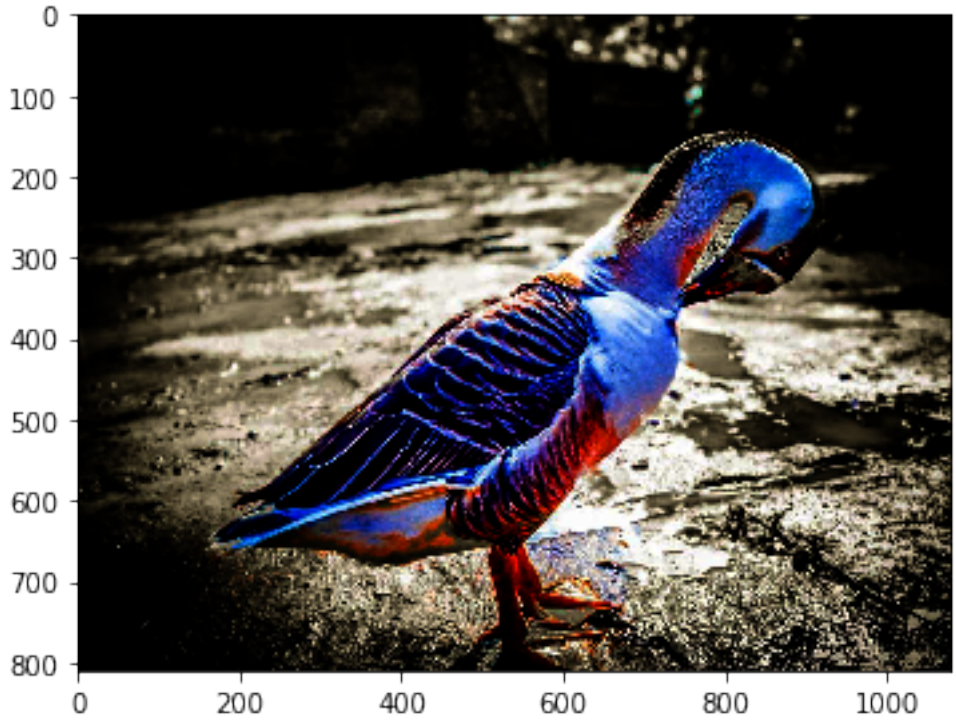

0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000

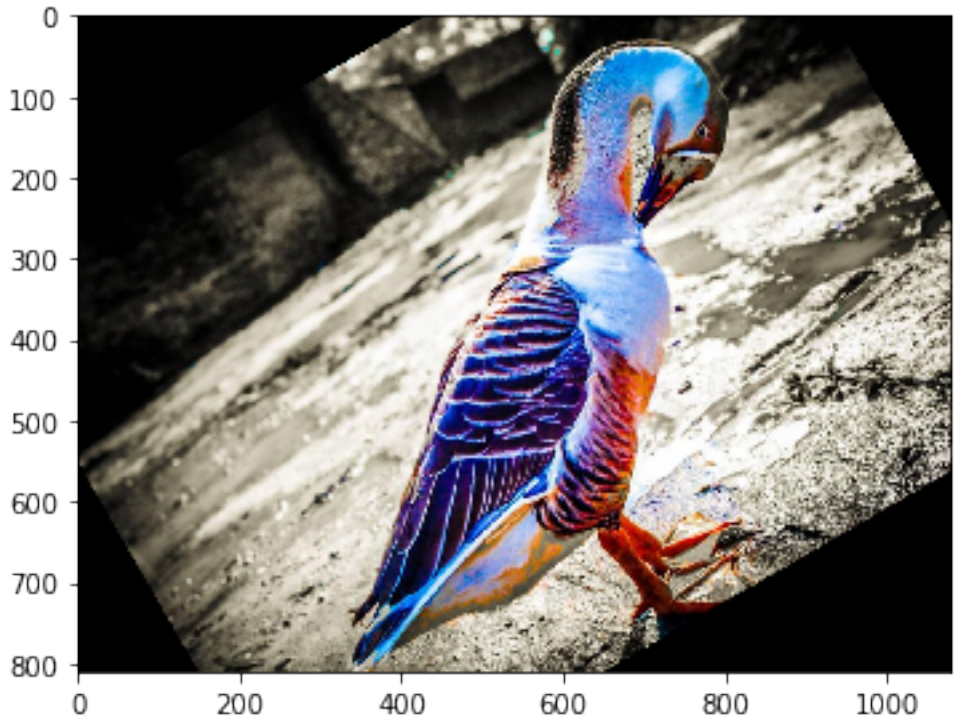

0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000

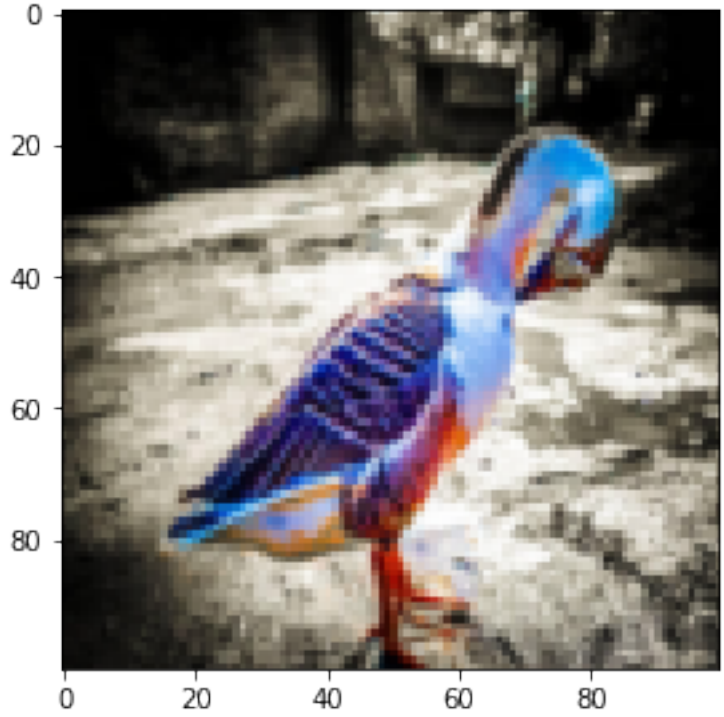

0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000
1.0
0.8
0.6
0.4
0.2
0.0


0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000


0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000


0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000





11. Graphs (If Any): Image /Soft copy of graph paper to be attached here