BAHRIA UNIVERSITY, ISLAMABAD
Department of Computer Science

CEN 444
Digital Image Processing
Lab Journal 2

Student Name: _M MUNEEB AHMED KIANI___________________________

Enrolment No.: __01-135212-063______________________________

Title: Basic Image Processing Operations.

Objectives: To introduce you to the basic functions in the Opencv. To be able to read images from the disk display them, write them back to the disk and perform conversions between different image classes.

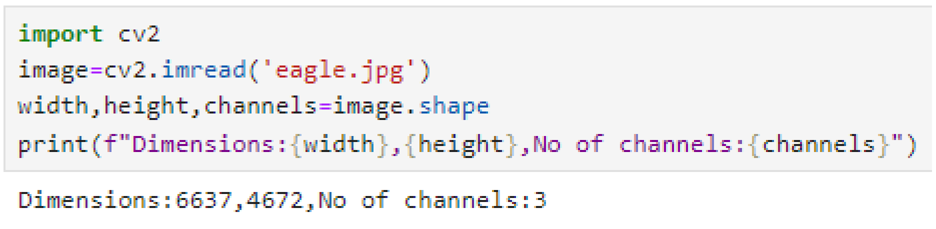
Tools Used: Jupyter Nootebook

Instructions for Submission:

- **Create a main repository** for the entire course on GitHub.
- **Within this main repository**, create separate folders for each week's content.
- **Upload your Jupyter notebooks** into the appropriate folder for each week.
- For the final submission, simply **upload a PDF file** that contains the GitHub link to your repository.

Task 1:

a) Load any image, find its dimensions and number of channels and display it.



```
import cv2
image=cv2.imread('eagle.jpg')
width,height,channels=image.shape
print(f"Dimensions:{width},{height},No of channels:{channels}")
```

```
Dimensions:6637,4672,No of channels:3
```

b) Find the size, date, coding method, bit depth, height and width.

```
import cv2
import os
import datetime
image=cv2.imread('eagle.jpg')
width,height,channels=image.shape
filesize=os.path.getsize('eagle.jpg')
modified_time=os.path.getmtime('eagle.jpg')
modified_date=datetime.datetime.fromtimestamp(modified_time)
print(f"Dimensions:{width},{height},No of channels:{channels}")
print("Image Size:",filesize)
print("Last modified time:",modified_time)
print("Last modified date:",modified_date)
print(f"Codng Method:JPEG ")
print(f"Bits Depth:8 bits")
```

```
Dimensions:6637,4672,No of channels:3
Image Size: 4364301
Last modified time: 1727527603.5262449
Last modified date: 2024-09-28 17:46:43.526245
Codng Method:JPEG
Bits Depth:8 bits
```

c) What happens if you convert a double having values outside the range [0 255] to an uint8?

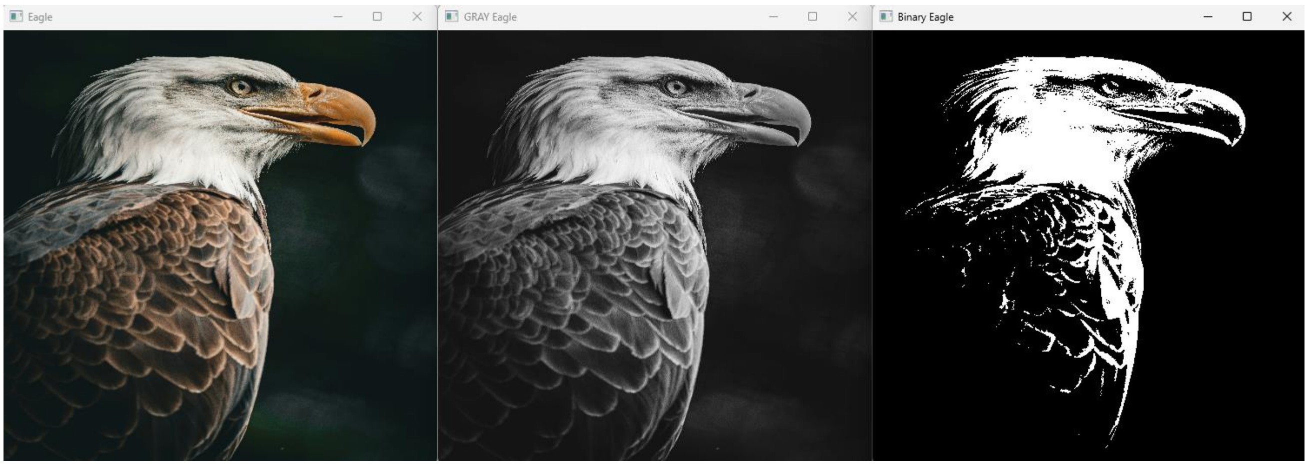
```
import numpy as np
double_array=np.array=[1.1,2.2,3.3,4.4,5.5]
uint8_array=np.clip(double_array,0,255).astype(np.uint8)
print("double array:",double_array)
print("uint_8 array:",uint8_array)
```

```
double array: [1.1, 2.2, 3.3, 4.4, 5.5]
uint_8 array: [1 2 3 4 5]
```

Task 2:

Load any image, binarize it, using the function ‘cv2.threshold’ with a threshold of 127 and display both original and binarized images.

```
import cv2
image=cv2.imread('eagle.jpg')
img=cv2.resize(image,(500,500))
gray_scale=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
_,binary_image=cv2.threshold(gray_scale,127,255,cv2.THRESH_BINARY)
cv2.imshow('Eagle',img)
cv2.imshow('GRAY Eagle',gray_scale)
cv2.imshow('Binary Eagle',binary_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



Task 3:

The function rgb2gray() can be used to convert three channel-colored images into single channel gray scale images.

Write program which reads any image in the. Find the size of the image and the number of channels in it. Use if else. If it is a three-channel image, then:

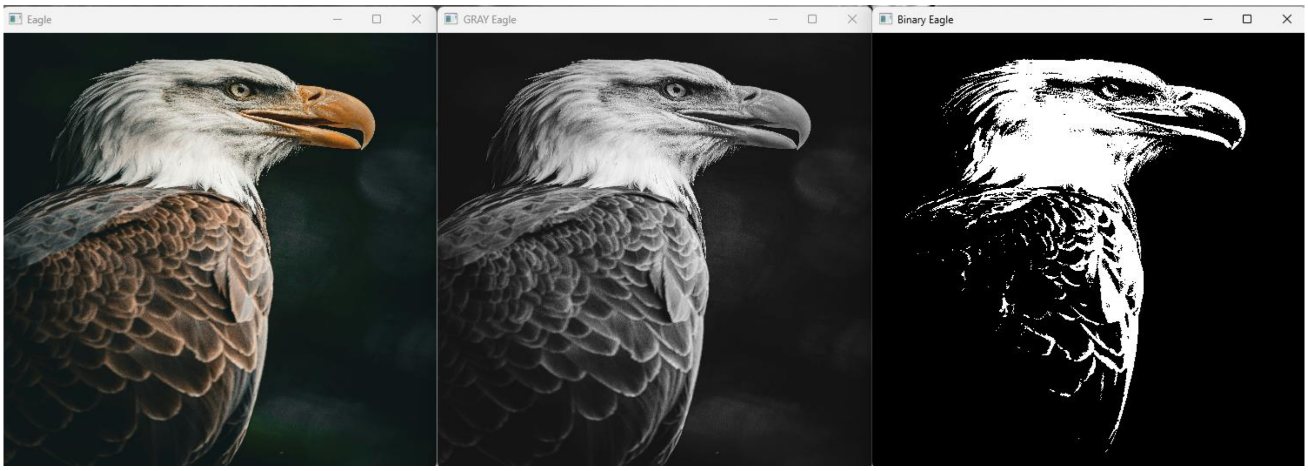
- display that it is a three-channel image
- then convert it to grayscale.
- Now binarize the gray image.
- Finally, display it.

If it is a one-channel image, then:

- display that it is a one-channel image
- Just display image.

```python
import cv2
image=cv2.imread('eagle.jpg')
img=cv2.resize(image,(500,500))
width,height,channels=img.shape
print(f"Dimensions:{width},{height},No of channels:{channels}")
if channels==3:
    print("This is a three channel image.")
    gray_scale=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    _,binary_image=cv2.threshold(gray_scale,127,255,cv2.THRESH_BINARY)
    cv2.imshow('Eagle',img)
    cv2.imshow('GRAY Eagle',gray_scale)
    cv2.imshow('Binary Eagle',binary_image)
elif channel==1:
    print("This is single channel image")
    cv2.imshow('Eagle',img)
else:
    print("Channels are unexpected.")
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
Dimensions:500,500,No of channels:3
This is a three channel image.
```

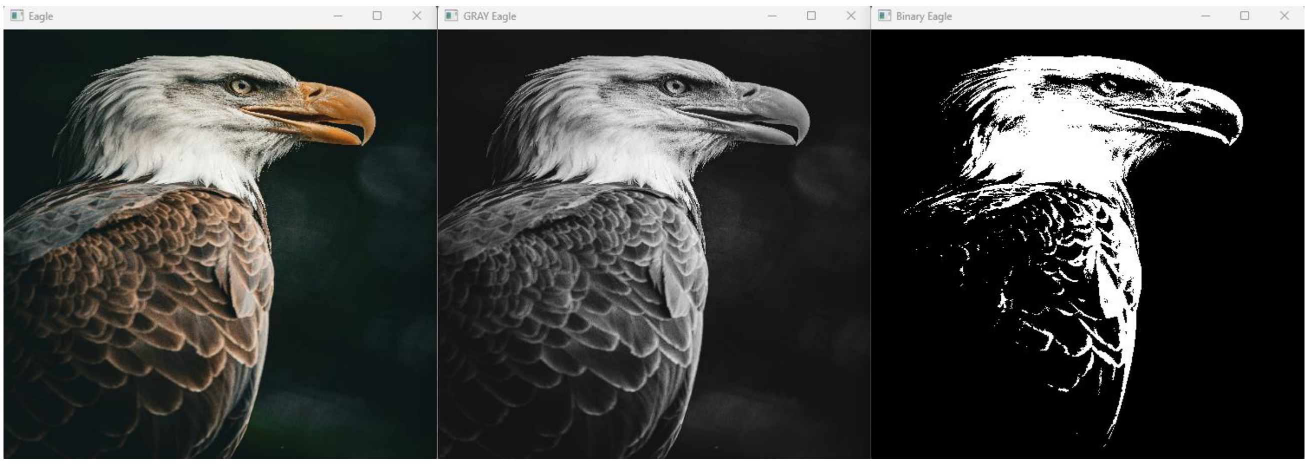


Task 4:

Write which reads the image. Binarize the image using your own implementation rather than the functions. Use two loops (nested), compare each value with a given threshold and generate the output image.

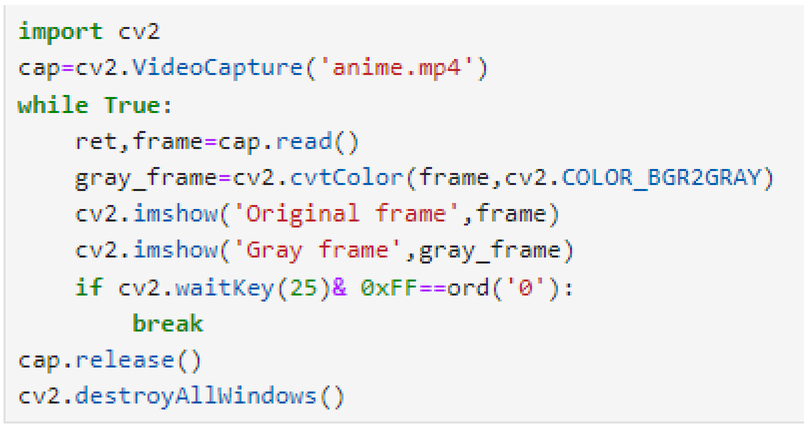
```python
import cv2
import numpy as np
image=cv2.imread('eagle.jpg')
img=cv2.resize(image,(500,500))
width,height,channels=img.shape
print(f"Dimensions:{width},{height},No of channels:{channels}")
if channels==3:
    print("This is a three channel image.")
    gray_scale=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
else:
    print("This is single channel image")
    gray_scale=img
binary_image=np.zeros((height,width),dtype=np.uint8)
threshold=127
for i in range(height):
    for j in range(width):
        if gray_scale[i,j]>threshold:
            binary_image[i,j]=255
        else:
            binary_image[i,j]=0
cv2.imshow('Eagle',img)
cv2.imshow('GRAY Eagle',gray_scale)
cv2.imshow('Binary Eagle',binary_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
Dimensions:500,500,No of channels:3
This is a three channel image.
```



Task 5:

Open webcam and perform gray scaling on the video frames, you can use opencv functions for gray scale.



```
import cv2
cap=cv2.VideoCapture('anime.mp4')
while True:
    ret,frame=cap.read()
    gray_frame=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    cv2.imshow('Original frame',frame)
    cv2.imshow('Gray frame',gray_frame)
    if cv2.waitKey(25)& 0xFF==ord('0'):
        break
cap.release()
cv2.destroyAllWindows()
```

rgb image.mp4

gray scale video.mp4

Submission Date: **Signature:**