Abdul basit

Digital image processing

**My code:**

Task1:

#import cv2 as cv

import numpy as np

import PIL

from PIL import Image

def rgbToNegative(im):

numrows=im1.shape[0]

numcols=im1.shape[1]

new\_img=np.zeros((numrows,numcols,3),dtype=np.uint8)

for i in range(numrows):

for j in range(numcols):

new\_img[i][j][0]=255-im1[i][j][0]

new\_img[i][j][1]=255-im1[i][j][1]

new\_img[i][j][2]=255-im1[i][j][2]

return new\_img

def grayToNegative(im1):

numrows=im1.shape[0]

numcols=im1.shape[1]

new\_img=np.zeros((numrows,numcols),dtype=np.uint8)

for i in range(numrows):

for j in range(numcols):

new\_img[i][j]=255-im1[i][j]

return new\_img

def toBinary(im\_2):

numrows=im\_2.shape[0]

numcols=im\_2.shape[1]

new\_img1=np.zeros((numrows,numcols),dtype=np.uint8)

for i in range(numrows):

for j in range(numcols):

if(im\_2[i][j]>=130):

new\_img1[i][j]=0

else:

new\_img1[i][j]=255

return new\_img1

#open the image

#img = cv.imread('lena\_color.jpg',0)

img = Image.open('lena\_color.jpg')

#binary\_negative=np.invert(im1)

option=input("1.RGB inversion\n2.grayscale\n3.binary")

if(option==1):

im1=np.asarray(img)

rgb\_negative=rgbToNegative(im1)

rgb=Image.fromarray(rgb\_negative)

rgb.show()

rgb.save("rgb.jpg")

elif(option==2):

img=img.convert('L')

im1=np.asarray(img)

gray\_negative=grayToNegative(im1)

gray=Image.fromarray(gray\_negative)

gray.show()

gray.save("graya.png")

else:

im1=img.convert('L')

im2=np.asarray(im1)

bn=toBinary(im2)

bnp=Image.fromarray(bn)

bnp.show()

bnp.save("Binary.jpg")



Task2:

#name: ABDUL BASIT

#SECTION: BSCS5A

# REG NO: 146120

import numpy as np

import PIL

from PIL import Image

def cal\_gradiant(im):

numrows=im.shape[0]

numcols=im.shape[1]

new\_img=np.zeros((numrows+2,numcols+2),dtype=np.uint8)

for i in range(numrows):

for j in range(numcols):

if(i+1<(numrows-2)):

new\_img[i][j]=abs(im[i+1][j]-im[i][j])

gradd=Image.fromarray(new\_img)

gradd.show()

gradd.save("gradd.jpg")

return 0

img = Image.open('lena\_color.jpg').convert('L')

im=np.asarray(img)

print(im)

cal\_gradiant(im)

