**Digital Image Processing**

**(Lab-05)**

**Name:** Ahmad Amjad Mughal

**Reg No:** 121672

**Class:** BSCS6C

**Task**

**Linear Contrast Sketching**

**Code**

#Ahmad Amjad Mughal

#121672

#Necessary libraries

import PIL

from PIL import Image

#required function for Normalizing Red Band

def RedNormalization(intensity):

minImage = 90

maxImage = 200

output = (intensity - minImage)\*(((255 - 0)/(maxImage - minImage)) + 0)

return output

#required function for Normalizing Green Band

def GreenNormalization(intensity):

minImage = 95

maxImage = 205

output = (intensity - minImage)\*(((255 - 0)/(maxImage - minImage)) + 0)

return output

#required function for Normalizing Blue Band

def BlueNormalization(intensity):

minImage = 101

maxImage = 220

output = (intensity - minImage)\*(((255 - 0)/(maxImage - minImage)) + 0)

return output

#Open an Image and convert it into RGB

image = Image.open('myImage.jpg').convert('RGB')

#Split the image into respective Colour bands

splitImage = image.split()

#Getting the improved intensity value of reespective color band and update intensiy level

redBand = splitImage[0].point(RedNormalization)

greenBand = splitImage[1].point(GreenNormalization)

blueBand = splitImage[2].point(BlueNormalization)

#merging the splitted image into one RGB Image

mergedImage = Image.merge('RGB',(redBand,greenBand,blueBand))

mergedImage.show()

mergedImage.save('NewContrastedImage.jpg')

**Screen-shot**

**Original Image**

****

**Contrasted Image Using Linear Contrast**

****

**Spatial Filtering**

**Code**

from PIL import Image

from scipy import ndimage, misc

import numpy as np

#Read the

img = misc.imread('myImage.jpg').astype(np.float)

#Define a kernel mask of 3\*3

kernel = np.array([0, -1, 0, -1, 4.75, -1, 0, -1, 0]).reshape((3, 3, 1))

# We Perform Convolution with the kernel and select the mode nearest

imgsharp = ndimage.convolve(img, kernel, mode='nearest')

# then we clip (0 to 255) and convert to unsigned int

imgsharp = np.clip(imgsharp, 0, 255).astype(np.uint8)

#Display the resulted image

Image.fromarray(imgsharp).show()

#Save the resulted image

Image.fromarray(imgsharp).save('mySharpenImage.jpg')

**Screenshot**

****

**Image after using Spatial Filtering**

****